USE OF WEATHER FORECAST DATA TO PREDICT THE ICE REQUIREMENTS FOR A THERMAL STORAGE SYSTEM

By

JAMES CRAIG TUCKER

Bachelor of Science in Mechanical Engineering

Oklahoma State University

Stillwater, Oklahoma

1985

Submitted to the Faculty of the Graduate College
of the Oklahoma State University
in partial fulfillment of the requirements
for the Degree of
MASTER OF SCIENCE
December, 1987

USE OF WEATHER FORECAST DATA TO PREDICT

THE ICE REQUIREMENTS FOR A

THERMAL STORAGE SYSTEM

Thesis Approved:

Gary E. Young
Thesis Adviser

Jerald D. Parker

Lacy? McLuister

Norman N. Durham
Dean of the Graduate College

PREFACE

A study was conducted to investigate the possibility of using weather forecast data to predict the ice requirements for the next day for a thermal storage system. An energy model was developed which estimated the energy consumption of a thermal storage system and a conventional chiller system. A random number generator was incorporated to simulate a prediction error. The performances of the two systems were compared. The results are promising and further research in this area is encouraged.

I would like to express my sincere appreciation to my major adviser, Dr. Gary E. Young, for his patience and guidance throughout my stay at Oklahoma State University. I am also thankful to my other committee members, Dr. Faye C. McQuistion and Dr. Jerald D. Parker, for their advisement during the course of my work. Special thanks are due to Mr. Steven M. Harris for his advice and assistance with the IBM AT.

I extend my deepest appreciation to my wife, Julia, for her patience, encouragement, and sacrifices during this project.

Finally, thanks to Ms. Charlene Fries for typing this manuscript.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

1.1. Problem Background

The use of thermal storage first became popular 30 to 50 years ago. Churches were the first to use ice storage for cooling. Approximately 2000 churches installed ice storage plates because they were less expensive to operate than large air conditioners. As electric rates decreased, nonstorage air conditioners became popular [1].

The conventional chiller air conditioning systems remained popular until the 1980's because of the abundance of low cost electricity. The United States is now faced with a serious problem. Electricity is becoming very expensive as power plants reach their maximum potentials during peak demand periods. New power plants, especially nuclear plants, are costly to build and operate.

The cooling of some 40 billion square feet of commercial floor space in the United States is the largest single contributor to the summer peak demand. Air conditioning the buildings constructed in 1986 alone added an estimated 1,600 megawatts to the summer peak demand which is equivalent to one and one-half large nuclear power plants [2].

With this onslaught of summer peak demand, utilities are faced with the option of building new power plants or leveling their loads over the course of a day. The peak output required from a power plant always occurs during the hottest summer days. At all other periods, such as night time and during the spring, fall, and winter months, the power plants are operating at only a fraction of their maximum peak capacity. Since existing plants operate on such a cyclical path, utilities would obviously prefer to use the economically lucrative potential of the off-peak periods occurring in the summer, at night, and on weekends, rather than build new costly power plants and prolong the problem. As an incentive for customers to use this off-peak energy, utilities are offering rate structures that encourage off-peak consumption and discourage on-peak consumption. Thermal storage is the means for the consumer to reduce his utility charges and the means of the utility company to level its load and postpone building expensive new power plants.

Thermal storage is classified into two divisions: ice storage and chilled water storage. Ice storage is superior in that more cooling potential can be stored in a smaller volume than chilled water. The negative aspect of ice storage is that the compressor works more inefficiently making ice than chilled water. The basic principle behind thermal storage is to shift the energy consumption of the building's peak cooling demand to an off-peak rate period. Ice can be produced at night for use during the next day. By taking advantage of the rate structures offered by the utilities, a building's cooling costs can be reduced by as much as 50%.

There are numerous existing computer programs that aid in the sizing and design of a thermal storage system for a building. Several can be found advertised in the ASHRAE Journal. The proper design and operation of a thermal storage system is very important for successful results. Thermal systems that are too small are not able to produce enough ice to handle the entire cooling load, causing the conventional chillers to operate during the on-peak period. Poor control of ice inventories can create ice shortages requiring the conventional chillers to operate during the on-peak period. Both of these scenarios cause a rise in the cost of cooling. New peaks are set when chillers have to operate during the on-peak periods. Higher peak demands bring higher costs of electricity from the utilities.

There are three different control strategies used with ice storage systems: chiller-priority, ice-priority, and constant proportion. Chiller-priority control is a strategy where the chiller is allowed to run at full capacity during the day and ice is melted only when the building load exceeds the chiller's full capacity. Ice-priority control is a strategy where as much ice as possible is melted during the day. The chiller would only operate when the melting ice could not properly cool the building. Constant proportion control is a strategy where the ice tanks and the chillers each handle a specified percentage of the building load. Each of these control strategies have their advantages and disadvantages [3]. This study will only examine ice-priority control.

The problem with ice storage is matching the ice produced and stored the night before to the cooling load the next day. Too little ice would cause the chillers to be used during the on-peak period causing higher demand charges. Too much ice would mean unnecessary thermal losses to the environment from storage causing higher than necessary electric bills.

A computer program will be used to help predict the amount of ice that needs to be produced and stored for a building's on-peak cooling load for the next day, thereby minimizing the building's energy usage. The program will use weather forecast data and building data to help predict the cooling load for the next day.

1.2 Literature Survey

A survey of the literature reveals that there is no direct work in utilizing weather forecast data to control the ice production in a thermal system.

Several sources have proposed using weather forecast data to help control ice production.

Rawlings [4] has suggested using weather forecast data and computer controlled equipment to decide when and how much chiller capacity to bring on for a day to help the thermal storage system handle a particularly hot day. Rawlings also proposed a control scheme that if the outdoor temperature at a predetermined time of day exceeded a predetermined outdoor temperature, then a portion of the chiller capacity could be used to help the ice storage last throughout the on-peak period without setting new peak demands.

Tamblyn [5] suggests the thermal storage operator keep records of weather conditions versus ton-hour requirements from the refrigerating plant and storage to help predict the ice requirements for the next day. It is suggested the calculation should be refined so that only a 10% storage margin is required.

Load forecasting for power companies parallels the thermal storage forecasting problem. Utilities also want to predict how much power will be needed for the day. This knowledge allows them to more economically operate their equipment and keep their costs down.

Nakamura [6] proposed one-day ahead load forecasting for power companies by the least-squares method using weather data and load curve properties. The accuracy of this model was very high with a standard deviation of about 3% for the load forecast.

Gupta and Yamada [7] developed a computer-oriented procedure for probabilistic forecasting of hourly power-system loads using historical load data and weather forecast information. The forecasting models are adaptive in that the model parameters are automatically corrected to keep track of changing load conditions.

CHAPTER II

MODEL DESCRIPTION

2.1 Model Selection and Description

Not all buildings are good candidates for thermal storage. The cooling load profiles of each building being considered for thermal storage should be observed so that only buildings with high afternoon peak loads and low night and morning loads would be fitted with thermal storage cooling systems. Office buildings produce excellent results with thermal storage because they fit this profile perfectly [8].

The building being modeled in this study is a public school service center located in Oklahoma City, Oklahoma. It is a medium weight, two-zone commercial building possessing 242,167 square feet of air-conditioned floor area. The walls have an overall heat transfer coefficient of 0.425 Btu/hr-ft^2-F. The roof has an average overall heat transfer coefficient of 0.240 Btu/hr-ft^2-F. The building also has 1270.8 square feet of window area.

The building occupancy of 85 begins at 0800 hours and ends at 1700 hours. The thermostat is set at 78°F from 0600 to 2200 hours to ensure proper comfort for the occupants. It is set back to 85°F at all other times during the summer months. There are 175.1 kW of fluorescent lighting and 203,200 Btu/hr of internal sensible heat gain during the day.

This has been a brief physical description of the building being modeled. Appendix A contains the input file which more fully describes the building being modeled. Figure 1 displays a typical daily load profile for this building. The high peak daily plateau and low night time cooling requirements make this building an ideal thermal storage candidate.

Each utility company will have its own rate structure. The one used with this model has the on-peak period from 0900 to 2200 hours at a rate of $.05510 per kWh of energy used. The off-peak period occurs from 2300 to 0800 hours and all day on weekends at a rate of $.02700 per kWh. These rates will represent the only charges examined in this study. The thermal storage was designed to handle the entire cooling load during the on-peak periods. Off-peak periods will be cooled with chilled water.

The thermal storage system is compared with a conventional chiller system. The efficiencies used for both systems examined are the same as those used in a comparison by McNeil and Mathey [9]. The ice storage system has a compressor efficiency of 1.03 kW/ton. The auxiliary loads have an efficiency of 0.19 kW/ton. The conventional chiller system has a compressor efficiency of 0.77 kW/ton and an auxiliary load efficiency of 0.48 kW/ton. It is assumed that the ice storage compressors can be used to produce chilled water for cooling during off-peak periods with an efficiency of 0.77 kW/ton and with an auxiliary load efficiency of 0.19 kW/ton. The efficiencies allow for the calculation of the energy consumption of the air conditioning equipment with knowledge of the heat extraction rate.

A schematic of the proposed compressor/ice storage arrangement is shown in Figure 2. During off-peak periods, ice storage system No. 4 will produce chilled water for the off-peak cooling needs. The other ice storage systems can produce and store ice during the off-peak periods along with system No. 4 when it is not needed for off-peak cooling. During the on-peak

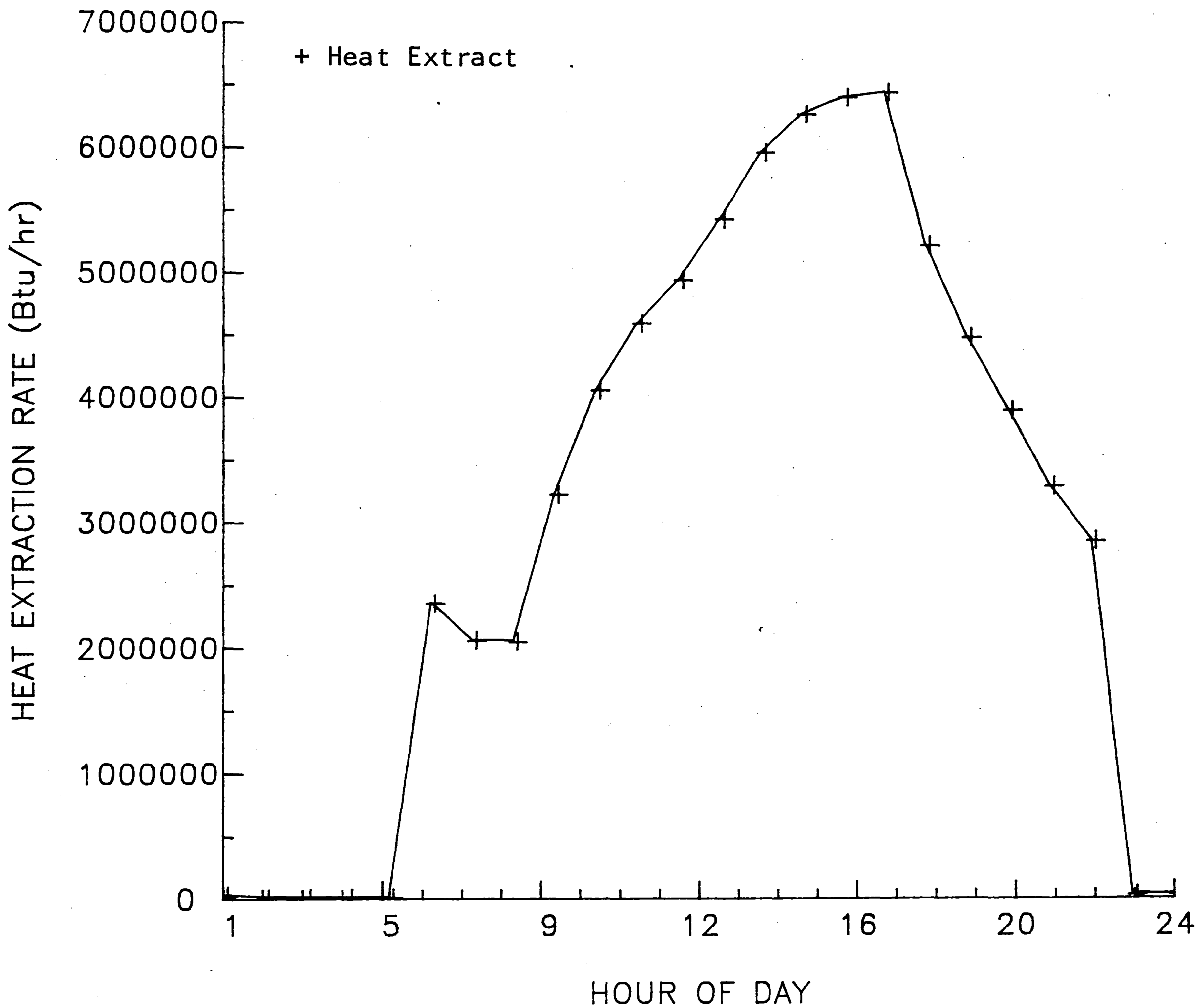

Figure 1. Daily Load Profile for Building Model

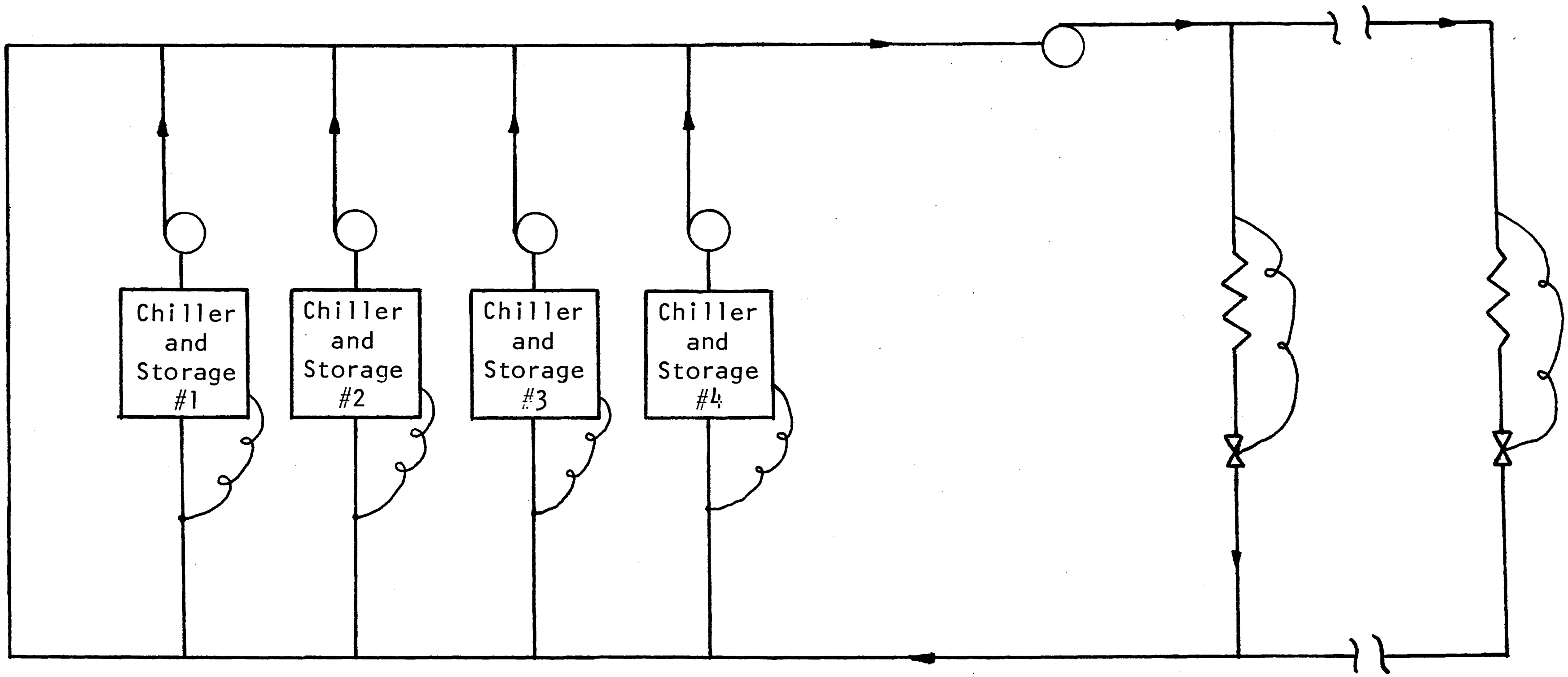

Figure 2. Schematic of Proposed Chiller/Ice Storage Configuration

periods when ice storage is depleted, the ice storage systems can produce chilled water at the higher efficiencies to cool the building.

There are thermal losses associated with the ice storage tanks. A loss of 5% per day is assumed for the stored ice. This is based on a rough estimate of the heat loss by the storage tank to the surroundings.

2.2 Computer Program Methodology

The computer program used to model this building is the FORTRAN program BLDSYM developed by Dr. Faye C. McQuiston at Oklahoma State University for the IBM 3081. It was designed to simulate the thermal performance of residential and commercial buildings. BLDSYM uses the transfer function thermal response method found in the 1981 ASHRAE Handbook of Fundamentals [10]. BLDSYM was modified to run on the IBM AT using the Microsoft FORTRAN (3.13) compiler.

BLDSYM estimates the instantaneous rate of heat gain, cooling load, and heat extraction rate on an hourly basis using the transfer function method mentioned above. The program uses hourly weather data and building parameters supplied by the user. Appendix A displays an input file which contains the input of the building model. There is an option at the beginning of the program that allows the user to store the input data in a file for future runs. This file can be edited and rerun, saving the user the time and trouble of inputting the building data for each run.

The space heat gain is classified in two categories: sensible and latent. The sensible heat gain is further divided into four areas:

1. Solar radiation through windows
2. Heat conduction through exterior walls

3. Heat generated by occupants, lights, and appliances
4. Energy transfer by ventilation and infiltration.

The latent heat gain is the heat gain of moisture added to the space. This can occur by ventilation, infiltration, occupants, plants, and some appliances.

The dry bulb and wet bulb temperatures, wind velocity, and solar radiation are input to the program from an ASHRAE WYEC weather file available on a floppy disk. All building data are input by the user in an interactive manner to the program.

The amount of solar radiation is calculated for the surfaces using the data from the WYEC weather file. The insolation on each surface is found using the SOLAR subroutine. After the insolation on a surface is found, the sol-air temperature for each surface is calculated. The transfer function method is then used to predict the instantaneous heat gain through each surface. The transfer function coefficients depend upon the construction of each surface and are responsible for accounting for the time delay of the energy absorbed by the building mass at previous hours.

The instantaneous heat gain from windows is found by use of the solar heat gain factor and shading coefficient. Heat loss is accounted for through the concrete slab floor. The internal loads, people, lights, and equipment are specified by the user. Infiltration is calculated using a mathematical model based on wind velocity, indoor temperature, and outdoor temperature. There is both a sensible and latent load associated with infiltration, people, and some appliances.

Once the instantaneous heat gains are found for each component discussed above, the transfer function method is used to convert the heat gains into cooling loads. There are four different classifications that heat gains are divided into:

1. Solar heat gain through glass and radiation from people and equipment
2. Conduction through outside surfaces
3. Radiation from lights
4. Convection from people, lights, equipment, and infiltration.

Each heat gain classification has its own set of transfer functions.

The heat extraction rate is the rate the air conditioning equipment must remove heat to maintain the desired building conditions. The heat extraction rate is converted from the cooling load by the transfer function method also. The heat extraction rate is the variable used to calculate the energy consumption by the building's air conditioning system in the ENERGY subroutine.

For a detailed description of the methodology used by BLDSYM, refer to a report by Hildebrand [11]. The program discussed by Hildebrand is a later version of BLDSYM and differs from the program discussed here only in format. The methodology and principles are the same. A program listing of BLDSYM can be found in Appendix B.

An energy model is used to calculate the energy consumption for a building and is contained in the ENERGY subroutine. Each building has its own energy model which is supplied by the program user. The ENERGY subroutine uses the heat extraction rate calculated by the subroutine HEATX to find the energy consumed by the air conditioning equipment. The lighting and equipment energy usage is also calculated. Previous versions of BLDSYM have had energy models that calculate the natural gas usage by the building for heating hot water and for heating the air space. Since this is a study of ice storage, the natural gas consumption was not examined.

The energy consumption of the cooling equipment is found by multiplying the heat extraction rate by the system efficiencies of the particular system being examined on an hour-by-hour basis. The efficiencies were discussed earlier and are summarized briefly here:

System efficiency for ice storage = 1.22 kW/ton

System efficiency for conventional chiller = 1.25 kW/ton

The ventilation air fan power consumption is also calculated on an hourly basis. The fan horsepower is simply converted to kilowatts by the conversion 0.746 kW/hp. The fluorescent lights' wattage is multiplied by the ballast factor 1.20 to find the energy used. There are wattage values for both day and night usage. The equipment power usage is simply the rating of the internal appliances and also has values for both day and night.

There are four cases examined by the energy model:

1. Ice storage model with perfect prediction
2. Conventional chiller model corresponding to 1
3. Ice storage model with imperfect prediction
4. Conventional chiller model corresponding to 3.

The ice storage model with perfect prediction uses the WYEC weather file data as its weather forecast data. Five percent is added to the subsequent heat extraction rate to account for the thermal losses encountered with the ice storage. Only the heat extraction rates from 0900 to 2200 hours are used to predict the amount of ice needed for cooling the next day. During the off-peak hours, the building is cooled with chilled water produced by the ice chillers at a higher efficiency. No ice is made on Friday or Saturday night because the weekend is an off-peak period and is cooled with chilled water. A system efficiency of 1.22 kW/ton is used for ice production and an efficiency of 0.96 kW/ton is used for the chilled water production.

Case 2, the conventional chiller model, uses the same heat extraction rates found with the WYEC weather file for Case 1 so a comparison can be made to the Case 1 thermal storage model. A system efficiency of 1.25 kW/ton is used.

Case 3, the ice storage model with imperfect prediction, uses a psuedo-random number generator to represent an error in the ice prediction. The random error in the ice prediction is used to simulate unseen changes in the weather which would affect the estimation of the heat extraction rate. The error is allowed to range from -20% to 20%. The prediction then is between 0.80 and 1.20 of the heat extraction rate. The thermal losses and system efficiencies are the same as those for Case 1. The weather forecast data are assumed to be the same as the weather data used in Case 1. Therefore, Case 1 is the prediction of ice production for the next day. When the ice prediction is too small, the ice chillers have to cool the building during the day. The number of hours that the chillers would have to operate is estimated by dividing the cooling required by two million. Two million was chosen because it is approximately the average hourly heat extraction rate over the later hours of the thermal storage cooling period--the hours when the chillers would have to operate. All off-peak loads are still cooled with the ice chillers producing chilled water at the higher efficiency. When the prediction is high, the excess ice is stored until the next day with a 5% thermal loss assumed. Therefore, the amount of ice predicted for the next night is partially met by the previous day's excess ice. Ice leftover on Friday is held until the following Monday with a 5% loss per day.

Case 4 is the conventional chiller model which uses the heat extraction rates used in Case 3. This allows a comparison of costs and energy usages between Cases 3 and 4.

CHAPTER III

RESULTS AND DISCUSSION

3.1 Conversion to IBM AT

Much work was expended converting BLDSYM to Microsoft FORTRAN and the IBM AT. When BLDSYM was working on the IBM AT, the output was compared to the output from the mainframe version for the same building. The results were nearly exact. The IBT AT version was proven to agree with the mainframe version.

3.2 Results

BLDSYM was run for both July and August to obtain a broad range of data to examine. Appendix C contains a sample of the output. The output data for July and August are summarized in Tables 1 and 2, respectively. Figures 3 through 10 graphically present the output data for both months.

Figure 3 compares the energy consumption of the perfect prediction ice storage system and the conventional chiller system for July. Because the system efficiencies are nearly identical, the curves are very similar. They mirror each other by an apparently constant amount. Notice that weekends are the "valleys" and weekdays are the "peaks."

Figure 4 compares the costs of the above systems, Case 1 and Case 2, for July. The thermal storage system takes advantage of the rate structure, so it is much more economical to operate than the conventional chiller system.

TABLE 1

ENERGY CONSUMPTION DATA FOR JULY

	Case 1		Case 2		Case 3		Case 4	
DAY	KWH	COST	KWH	COST	KWH	COST	KWH	COST
182	9400.	330.	9600.	480.	11400.	440.	10500.	530.
183	10100.	350.	10400.	530.	10100.	350.	10200.	520.
184	10700.	370.	11000.	560.	10600.	370.	10500.	540.
185	4600.	120.	5200.	140.	4700.	130.	5600.	150.
186	5200.	140.	6000.	160.	4700.	130.	4900.	130.
187	12000.	400.	12400.	620.	11700.	400.	11700.	590.
188	11000.	380.	11300.	580.	11000.	390.	11700.	600.
189	10800.	370.	11100.	560.	10800.	370.	10900.	550.
190	10800.	370.	11100.	560.	10700.	370.	10900.	550.
191	10900.	370.	11200.	570.	10800.	370.	11200.	570.
192	4800.	130.	5500.	150.	5100.	140.	6400.	170.
193	5600.	150.	6700.	180.	5400.	140.	6000.	160.
194	12200.	410.	12600.	630.	12200.	410.	12800.	640.
195	11700.	400.	12100.	610.	11700.	400.	11000.	560.
196	9600.	340.	9900.	490.	8700.	320.	8900.	440.
197	9000.	320.	9200.	460.	8600.	320.	9500.	480.
198	9400.	330.	9600.	490.	13100.	530.	10800.	550.
199	3900.	100.	4300.	120.	3700.	100.	3900.	100.
200	4900.	130.	5700.	150.	5400.	140.	7000.	190.
201	11600.	390.	12000.	590.	11400.	390.	10500.	520.
202	10300.	360.	10500.	530.	9200.	330.	10500.	530.
203	10900.	380.	11200.	570.	11200.	390.	11600.	590.
204	9600.	340.	9900.	490.	9500.	340.	9000.	450.
205	7500.	280.	7600.	380.	6800.	260.	7600.	380.
206	2500.	70.	2500.	70.	2500.	70.	2500.	70.
207	3600.	100.	4000.	110.	3400.	90.	3500.	90.
208	9300.	330.	9600.	470.	11400.	450.	10622.	520.
209	7300.	280.	7500.	370.	7300.	280.	7400.	360.
210	8300.	310.	8500.	430.	8200.	300.	7900.	400.
211	9300.	330.	9500.	480.	11600.	470.	10500.	530.
212	8500.	310.	8700.	440.	8500.	310.	8500.	420.
	265300.	9090.	276400.	12700.	271400.	9500.	274500.	12800.

TABLE 2

ENERGY CONSUMPTION DATA FOR AUGUST

	Case 1		Case 2		Case 3		Case 4	
DAY	KWH	COST	KWH	COST	KWH	COST	KWH	COST
213	3600.	100.	4000.	110.	3400.	90.	3500.	90.
214	5000.	130.	5800.	160.	4600.	120.	4800.	130.
215	11000.	380.	11400.	560.	11900.	430.	12200.	600.
216	9000.	320.	9200.	460.	9800.	370.	9800.	490.
217	7300.	280.	7400.	370.	7300.	280.	7300.	370.
218	7000.	270.	7100.	360.	7000.	270.	7300.	370.
219	6900.	270.	7000.	350.	7000.	270.	7100.	360.
220	2700.	70.	2800.	80.	2700.	70.	2900.	80.
221	4000.	110.	4500.	120.	3900.	110.	4300.	120.
222	10000.	350.	10300.	510.	10200.	360.	10600.	530.
223	9700.	340.	9900.	500.	9700.	340.	9500.	480.
224	9800.	340.	10000.	510.	9400.	340.	9800.	500.
225	10100.	350.	10400.	530.	10800.	400.	11000.	560.
226	8900.	320.	9100.	450.	9400.	350.	9600.	450.
227	3000.	80.	3200.	90.	2900.	80.	2900.	80.
228	4200.	110.	4800.	130.	4300.	120.	5100.	140.
229	8500.	310.	8800.	430.	8500.	310.	8700.	420.
230	6700.	260.	6800.	340.	6700.	260.	6900.	340.
231	6900.	270.	6900.	350.	7200.	280.	7200.	370.
232	7500.	280.	7600.	380.	7600.	290.	7800.	390.
233	7700.	290.	7800.	390.	8700.	340.	8400.	430.
234	3000.	80.	3200.	90.	2900.	80.	2900.	80.
235	3500.	100.	3900.	110.	3400.	90.	3500.	90.
236	8600.	310.	8900.	430.	8600.	310.	9000.	440.
237	7500.	280.	7600.	380.	7800.	300.	7900.	400.
238	8500.	310.	8600.	440.	8500.	310.	8600.	440.
239	9200.	330.	9400.	480.	9100.	330.	8400.	420.
240	8700.	320.	8900.	450.	8300.	320.	9300.	470.
241	2900.	80.	3100.	80.	2900.	80.	2900.	80.
242	3600.	100.	4000.	110.	3500.	90.	3700.	100.
243	9500.	340.	9800.	480.	9400.	330.	9200.	460.
	214500.	7480.	222200.	10300.	217400.	7730.	222100.	10280.

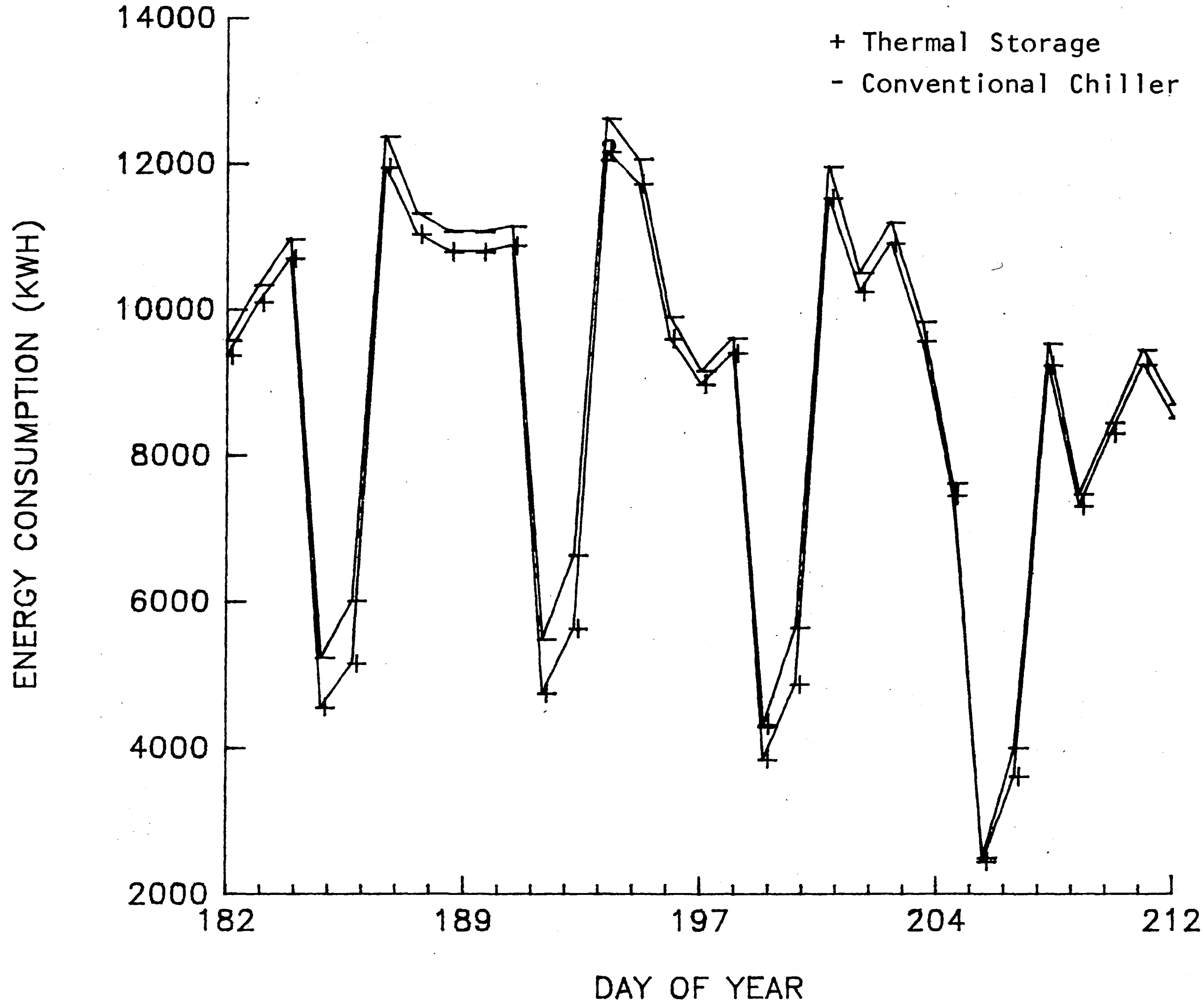

Figure 3. Comparison of Energy Usage for Case 1 and Case 2 for July

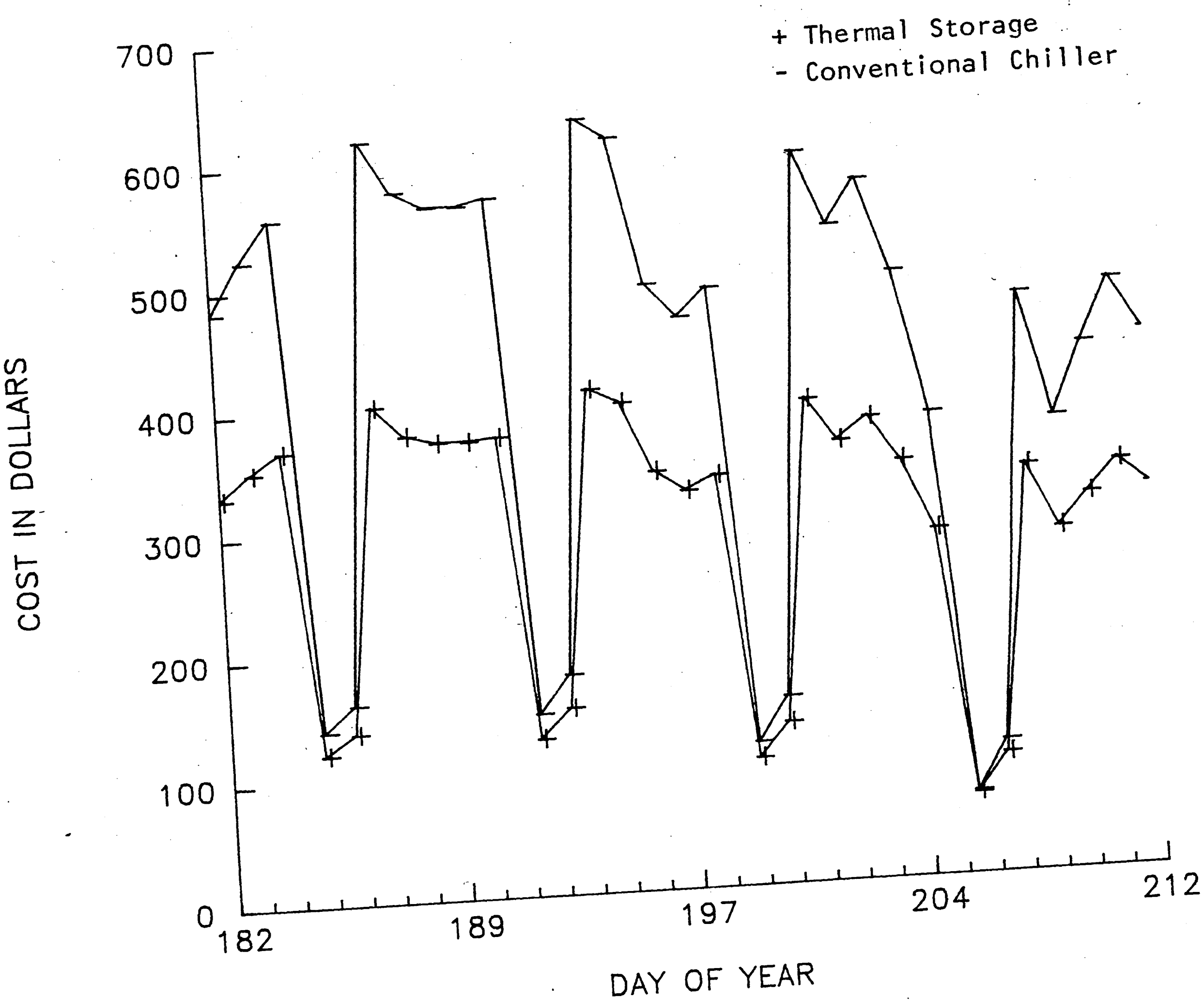

Figure 4. Comparison of Cost for Case 1 and Case 2 for July

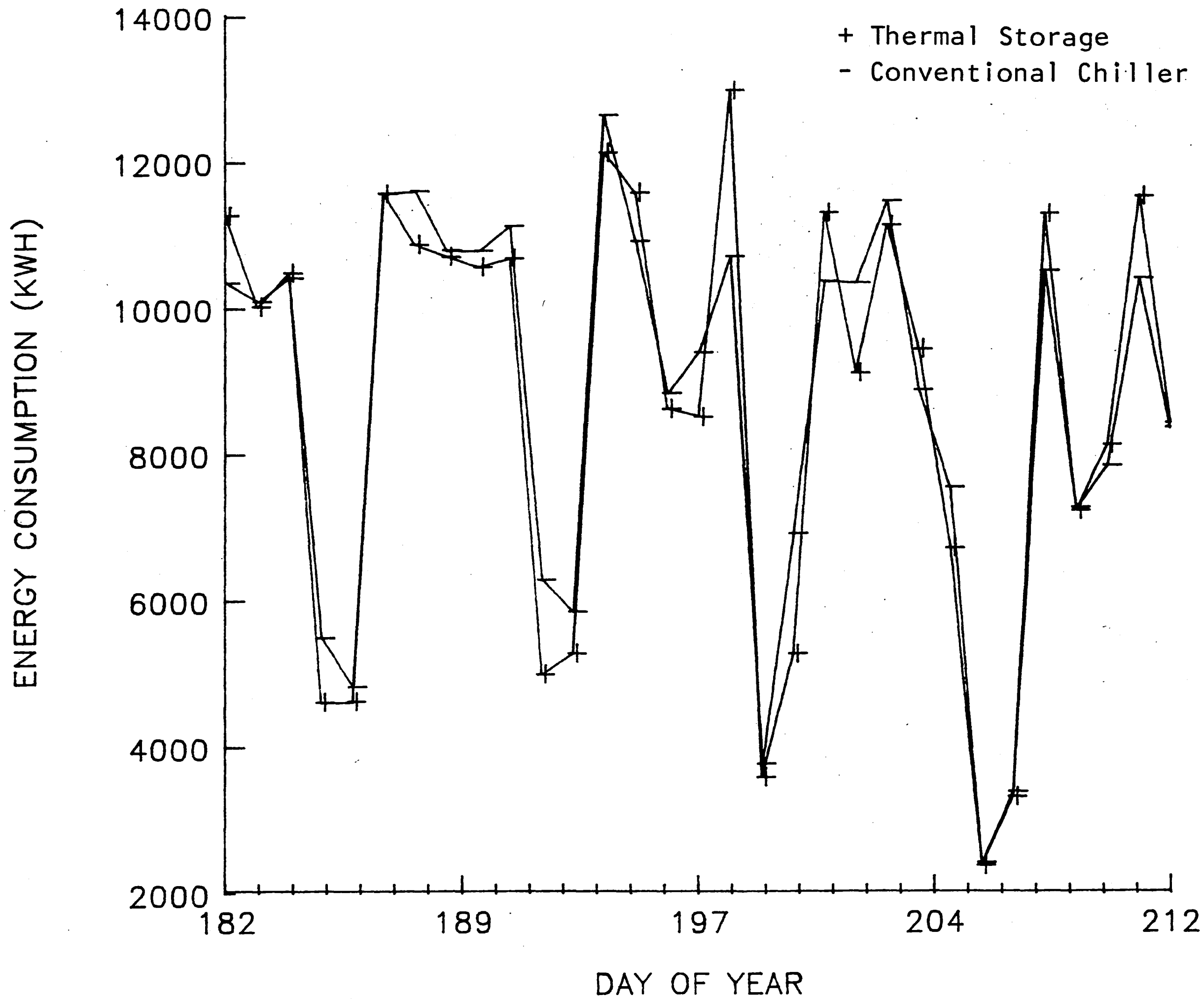

Figure 5. Comparison of Energy Usage for Case 3 and Case 4 for July

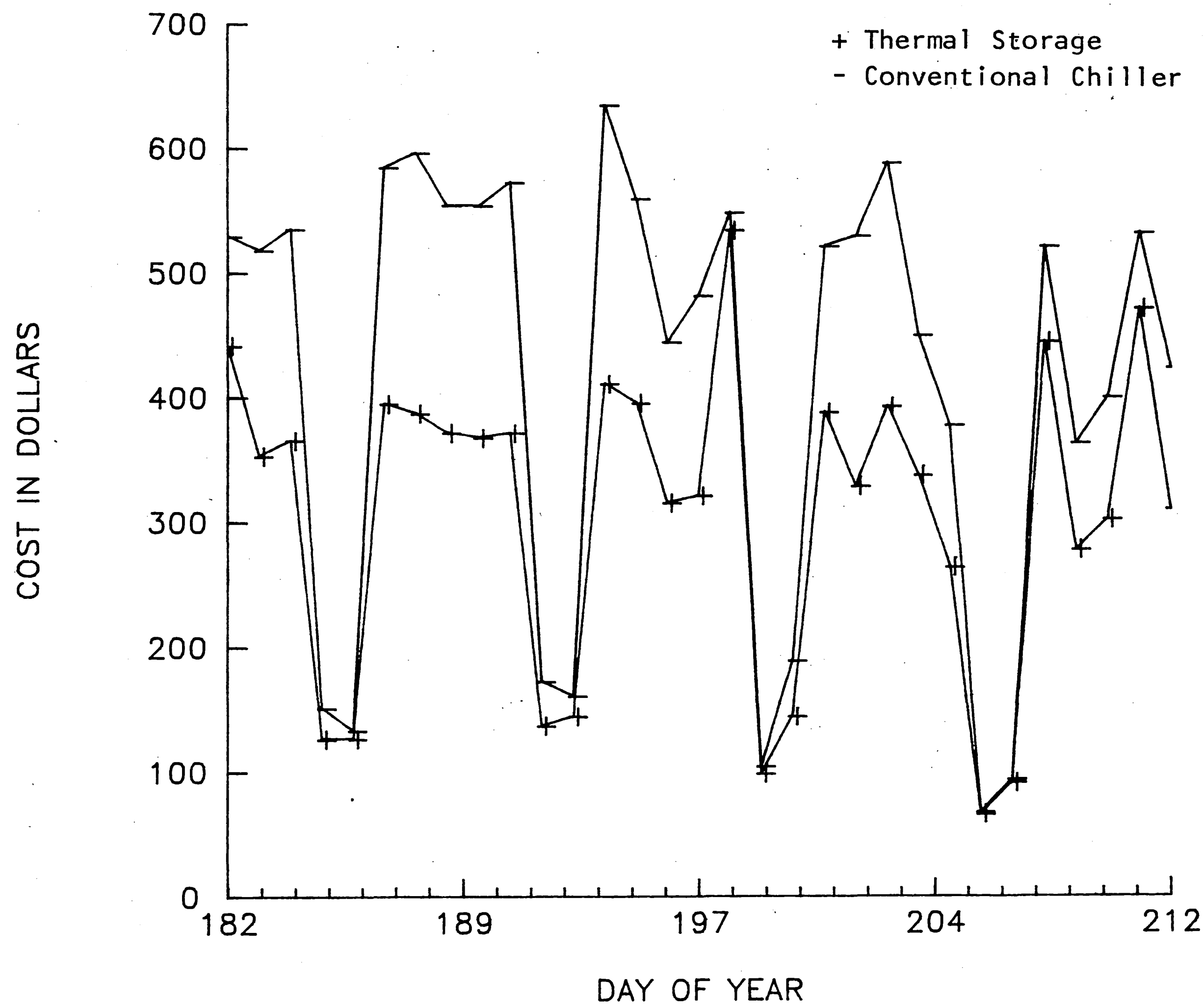

Figure 6. Comparison of Cost for Case 3
and Case 4 for July

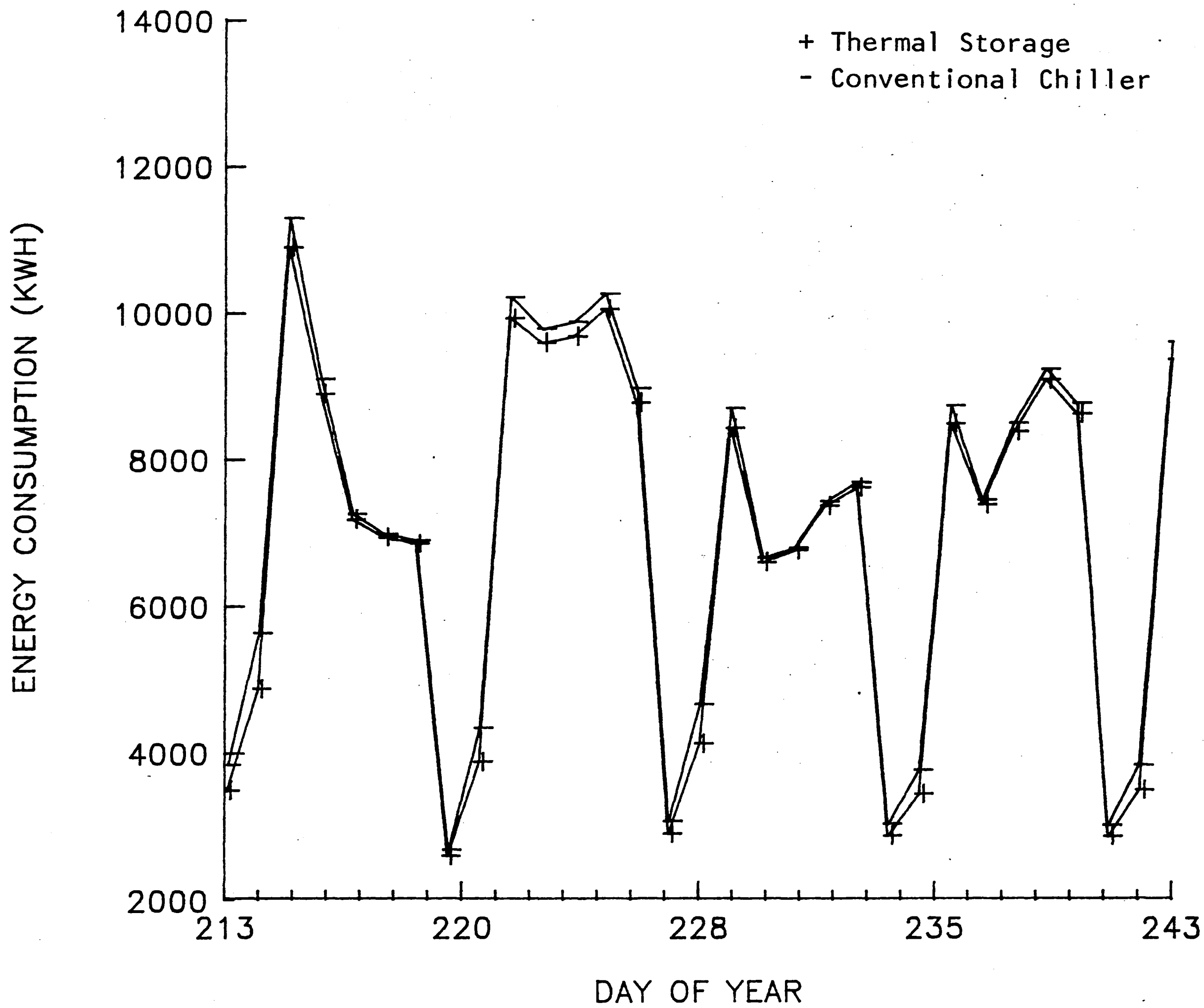

Figure 7. Comparison of Energy Usage for Case 1 and Case 2 for August

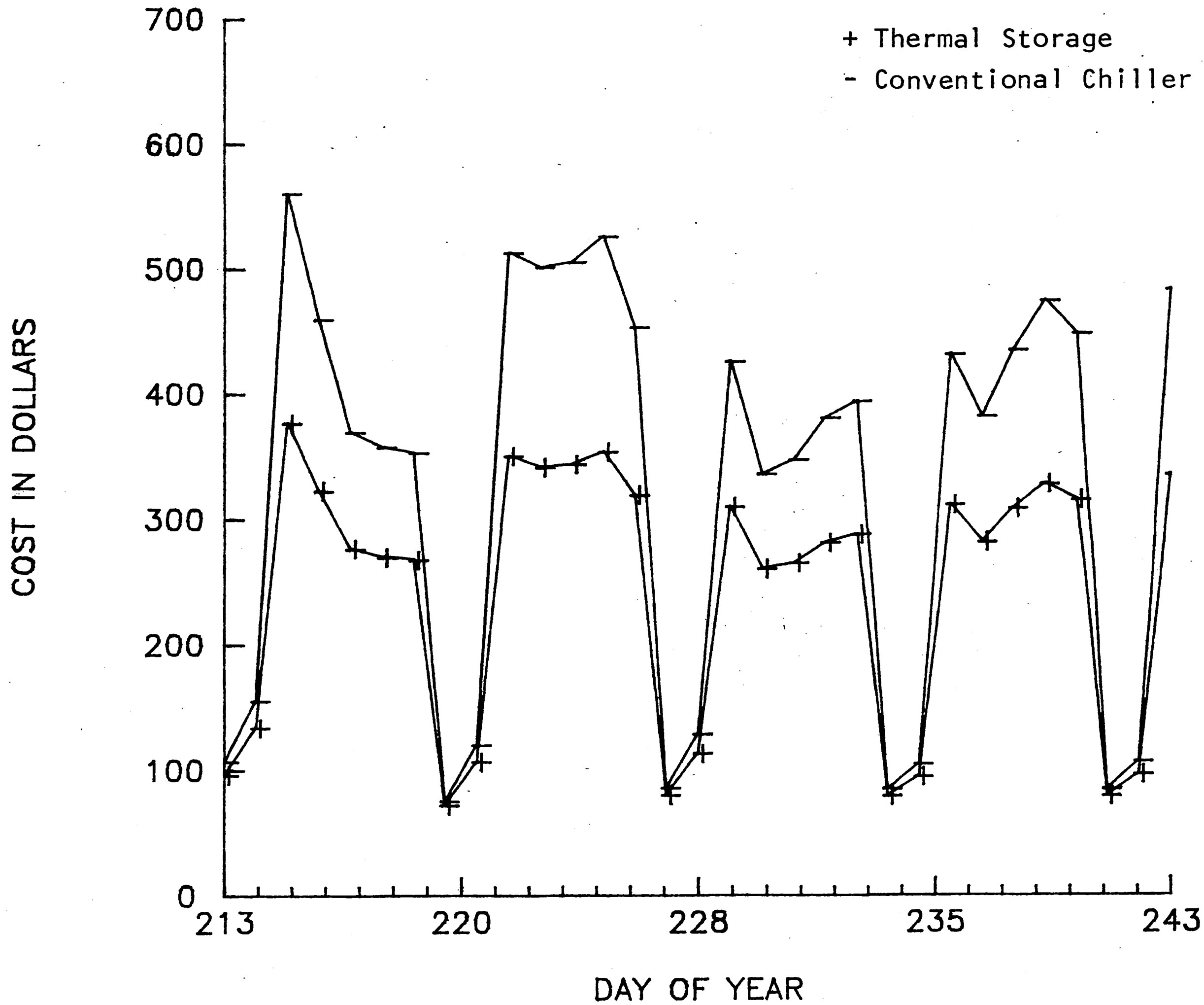

Figure 8. Comparison of Cost for Case 1
and Case 2 for August

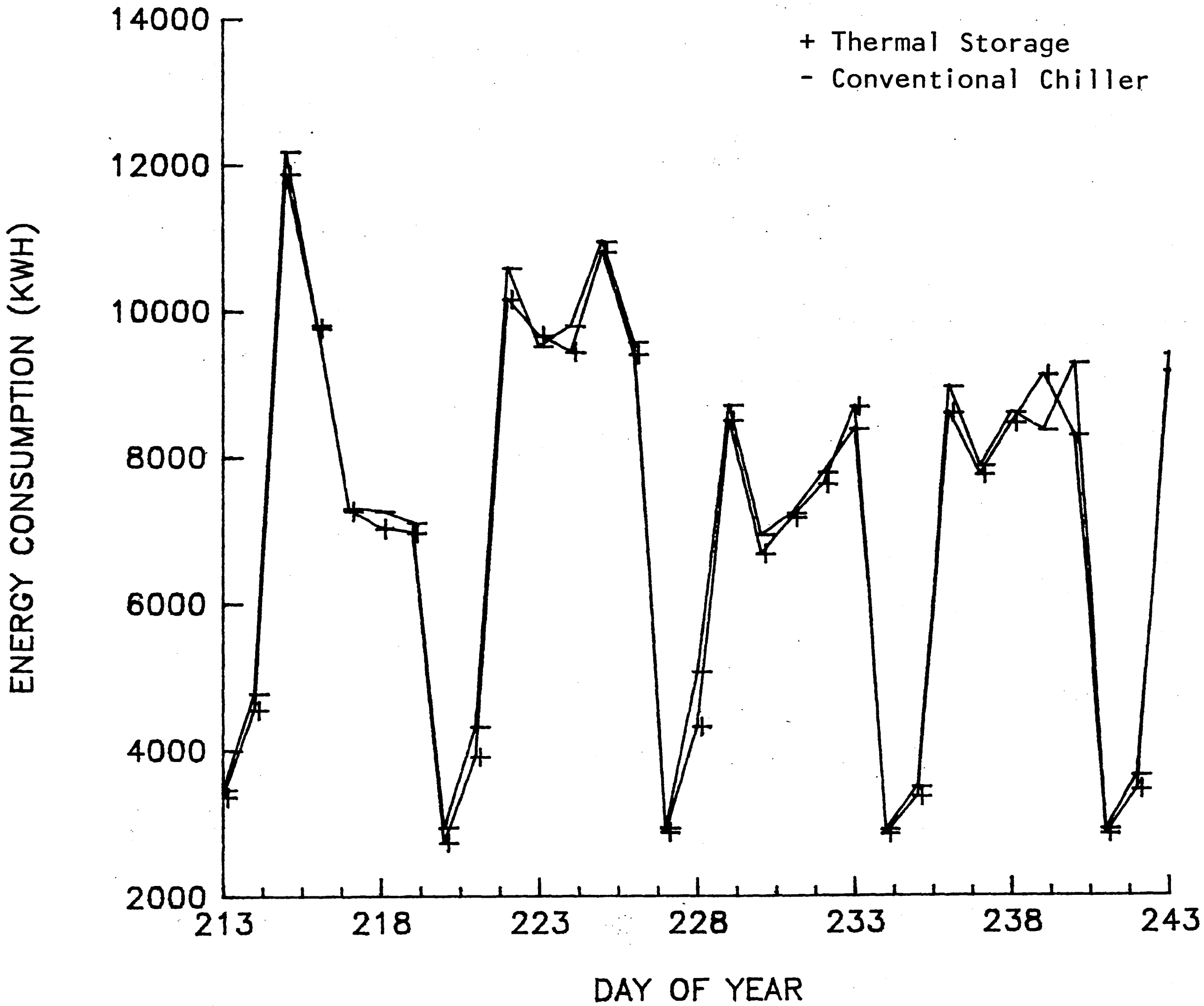

Figure 9. Comparison of Energy Usage for Case 3 and Case 4 for August

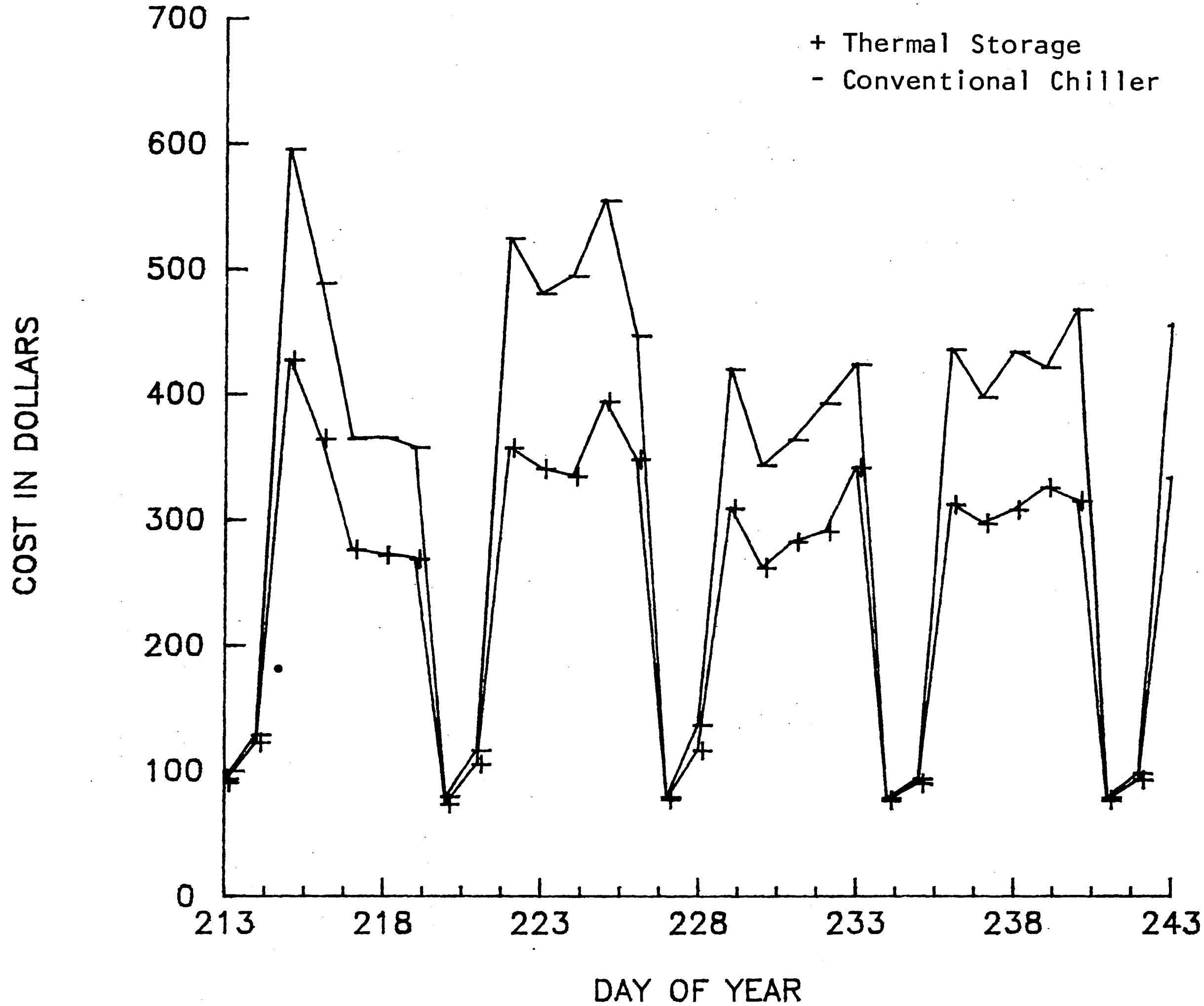

Figure 10. Comparison of Cost for Case 3 and
Case 4 for August

Notice that weekends are similar in cost. This is because both systems are operating during the off-peak period. Since Figure 4 assumes perfect prediction for the ice storage system, the area between the two curves represents the maximum savings attainable by the ice storage system for the weather conditions used.

Figure 5 compares the energy consumption between Cases 3 and 4 for July. Here the psuedo-random number generator has assigned an error between -20% and +20% for the ice prediction. This is compared to the conventional chiller performance for the same weather data. On a monthly basis the curves display approximately equal energy consumption for the two systems. Comparing Figure 5 to Figure 3, the curves do not parallel each other in Figure 5 as they do in Figure 3. This is because of the ice prediction scheme. When ice is leftover, the energy consumption is higher for that day than the conventional chiller but that ice is used the following day. Therefore, the next day's energy consumption will be less than the psuedo-random error prediction because of the leftover ice. Over the entire month, the total energy consumption for the psuedo-random error prediction system was about the same as the total energy consumption for the perfect prediction system.

Figures 7 through 10 are the results for August. These figures correspond to Figures 3 through 6 for the month of July. Notice that the general trends for July also hold for August.

Table 1 shows for the month of July the perfect prediction ice system consumed 265300 kWh compared to 276400 for the conventional system. The random error ice system consumed 271400 kWh compared to 274500 for the conventional system. The random error system more closely approached the conventional system in energy consumption because of the error of the ice predictions. August compares favorably with July. Table 2 shows perfect prediction consumed 214500 kWh compared to 222200 for the conventional system. The random error ice system consumed 217400 kWh compared to 222100 for the conventional chiller system.

Table 1 shows that for July the perfect prediction ice storage system cost $9090 to operate compared to $12700 for the conventional system. This represents a savings of 28%. This would be the maximum savings of an ice storage system over a conventional chiller system for these weather conditions. The random error prediction system cost $9500 to operate compared to $12800 for the conventional system. This represents a savings of 26%--still a significant savings.

Table 2 shows that for August the perfect prediction system cost $7480 to operate compared to $10300 for the conventional system. This would be a savings of 26%. The random error ice prediction system cost $7730 to operate compared to $10280 for the conventional system, representing a savings of 25%. It must be remembered that no penalties for setting new peak demand charges were assumed which would not be the real case. The real savings could be considerably less depending upon the penalties levied for using extra on-peak energy and the cost of amortizing the thermal storage equipment.

August savings were lower because the random number generator had 17 days "hotter" than expected in August as compared to 14 days "hotter" for July. The chillers run more in August during the on-peak period. The savings would definitely depend upon how accurate the weather prediction was for the month.

CHAPTER IV

CONCLUSIONS AND RECOMMENDATIONS

4.1 Conclusions

The energy model for the four cases was examined over a two-month period using ice priority control for the ice storage system. A psuedo-random number generator was used to simulate a prediction error of -20% to +20% of the heat extraction rate. The savings over a conventional chiller system for the month of July was 28% for perfect prediction and 26% for random error prediction. There was a savings of 27% for perfect prediction and 25% for random error prediction for August. It can be seen that, for a rate structure without demand charges, the random error prediction does not severely penalize the economic savings of the ice storage system.

Using weather forecast data for predicting the ice storage requirements for the next day has definite economic potential for the utility companies and the consumer. An accurate weather forecast can help trim the operating costs of the thermal storage operation.

4.2 Recommendations

It is recommended that an accurate weather forecasting model be developed and tested on an actual building with a thermal storage system. The building data should be refined to obtain as accurate a model as possible. The prediction model could be tested to analyze the reasonable maximum savings of an ice storage system over a conventional system. Penalties for setting new peak demands should also be examined as well as their effect on the economics of the thermal storage operation.

The three control strategies discussed--chiller-priority, ice-priority, and constant proportion--should be analyzed to find which control scheme is the most economical. Different rate structures which more accurately predict the real costs per month should also be used to test the economics of using ice storage systems. The amortized costs of the equipment for the three different control strategies should be compared in an overall economic analysis.

Since the peak cooling load does not necessarily exist during the summer months, the analysis should also be expanded to include the entire year.

It is also recommended that a new version of Microsoft FORTRAN be used because of the difficulty encountered with Version 3.13. When using Version 3.13, the FORTRAN program would compile without errors, but the program would terminate with a run-time error. When using WRITE statements to find where the error occurred, the program would run with good output. The reason for this run-time error was never found but was believed to be due to the large program size.

REFERENCES

[1] MacCracken, C. D. "Off-peak Market is Utility Driven." ASHRAE Journal, Vol. 29, No. 5 (May, 1987), p. 21.

[2] Wendland, R. D. "Storage to Become Rule not Exception." ASHRAE Journal, Vol. 29, No. 5 (May, 1987), p. 22.

[3] Rawlings, L. K. "Ice Storage System Optimization and Control Strategies." ASHRAE Transactions 1985, Vol. 91, Part 1B, p. 12.

[4] Rawlings, L. K. "Ice Storage System Optimization and Control Strategies." ASHRAE Transactions 1985, Vol. 91, Part 1B, p. 12.

[5] Tamblyn, R. T. "Control Concepts for Thermal Storage." ASRAE Transactions 1985, Vol. 91, Part 1B, p. 5.

[6] Nakamura, M. "Short Term Load Forecasting Using Daily Updated Load Models." Automatica 1985, Vol. 21, p. 729.

[7] Gupta, P. C., and K. Yamada. "Adaptive Short-Term Forecasting of Hourly Loads Using Weather Information." IEEE Winter Meeting, New York, NY, January 30-February 4, 1982, p. 2085.

[8] Gatley, D. P., and J. J. Riticher. "Successful Thermal Storage." ASHRAE Transactions 1985, Vol. 91, Part 1B, p. 843.

[9] McNeil, W. P., and J. D. Mathey. "Review of an Operating Ice Storage System Performance." ASHRAE Technical Data Bulletin, "Thermal Storage." "A collection of papers presented during the ASHRAE Winter meeting, January, 1985, at Chicago, Illinois," p. 124.

[10] ASHRAE Handbook of Fundamentals, 1981. Chapter 26, p. 261.

[11] Hildebrand, V. A. "Modification of the Energy Simulation Program--BLDSYM." Master's Report, Oklahoma State University, December, 1987.

APPENDIX A

INPUT FILE

```
 INPUT FILE NAME
COOL.DAT
 INPUT OUTPUT TYPE
1
 INPUT INWRIT
1
 ENTER NUMBER OF ZONES
2
 ENTER THE NUMBER OF SURFACES FOR THIS ZONE
3
 ENTER THE NUMBER OF SURFACES FOR THIS ZONE
5
 ARE THE ZONES AND THE CORRESPONDING NUMBER OF WALLS CORRECT? (Y=0/N=1)

 INPUT LOCAL LATITUDE - DEG
35.0
 INPUT LOCAL LONGITUDE - DEG
97.0
 INPUT STANDARD LONGITUDE - DEG (75,90,105,120,ETC)
90.0
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 INPUT CONSTRUCTION TYPE
2
 INPUT LIGHT FIXTURE TYPE
2
 INPUT FLOOR TYPE
2
 INPUT CIRCULATION TYPE
2
 INPUT OUTSIDE PERIPHERAL LENGTH OF ZONE, FT
282.
 INPUT MINIMUM ZONE RELATIVE HUMIDITY
.2
 INPUT MAXIMUM ZONE RELATIVE HUMIDITY
.3
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 ENTER SURFACE DESCRIPTION - DIR., ROOF, ETC
SOUTH WALL
 INPUT WALL HGT OR ROOF LGTH - FT
201.
 INPUT WALL WDTH OR ROOF WDTH - FT
8.8
 INPUT AREA OF THE DOORS - SQFT
342.6
 INPUT THE AREA OF THE WINDOWS - SQFT
86.4
 INPUT TILT OF THE SURFACE FROM THE HORIZONTAL - HORZ=0.0, VERT=90.0 - DEG
90.0
 INPUT DIR SURFACE FACES - DEG CNTR-CLKWISE FROM SOUTH - ENTER 0.0 FOR HORZ.
0.0
```

```
 INPUT THE OUTSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
3.0
 INPUT THE INSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
1.46
 INPUT THE REFLECTANCE FROM SURROUNDINGS
.2
 INPUT ABSORBANCE OF THE SURFACE
.8
 INPUT EMITTANCE OF THE SURFACE
.9
 INPUT SHADING COEFFICIENT OF GLASS
.25
 INPUT ACTUAL OVERALL COND. OF WALL OR ROOF - BTU/HR-SQFT-F
.425
 INPUT OVERALL CONDUCTANCE OF WINDOWS - BTU/HR-SQFT-F
.95
 INPUT OVERALL CONDUCTANCE OF DOORS - BTU/HR-SQFT-F
.6
 PRESS RETURN TO CONTINUE LIST

 ENTER SURFACE NUMBER OR 0
19
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 ENTER SURFACE DESCRIPTION - DIR., ROOF, ETC
EAST WALL
 INPUT WALL HGT OR ROOF LGTH - FT
81.0
 INPUT WALL WDTH OR ROOF WDTH - FT
8.8
 INPUT AREA OF THE DOORS - SQFT
21.
 INPUT THE AREA OF THE WINDOWS - SQFT
197.9
 INPUT TILT OF THE SURFACE FROM THE HORIZONTAL - HORZ=0.0, VERT=90.0 - DEG
90.0
 INPUT DIR SURFACE FACES - DEG CNTR-CLKWISE FROM SOUTH - ENTER 0.0 FOR HORZ.
90.
 INPUT THE OUTSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
3.0
 INPUT THE INSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
1.46
 INPUT THE REFLECTANCE FROM SURROUNDINGS
.2
 INPUT ABSORBANCE OF THE SURFACE
.8
 INPUT EMITTANCE OF THE SURFACE
.9
 INPUT SHADING COEFFICIENT OF GLASS
.96
 INPUT ACTUAL OVERALL COND. OF WALL OR ROOF - BTU/HR-SQFT-F
.425
```

```
 INPUT OVERALL CONDUCTANCE OF WINDOWS - BTU/HR-SQFT-F
.95
 INPUT OVERALL CONDUCTANCE OF DOORS - BTU/HR-SQFT-F
.6
 PRESS RETURN TO CONTINUE LIST

 ENTER SURFACE NUMBER OR 0
19
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 ENTER SURFACE DESCRIPTION - DIR., ROOF, ETC
ROOF
 INPUT WALL HGT OR ROOF LGTH - FT
81.0
 INPUT WALL WDTH OR ROOF WDTH - FT
201.0
 INPUT AREA OF THE DOORS - SQFT
0.0
 INPUT THE AREA OF THE WINDOWS - SQFT
0.0
 INPUT TILT OF THE SURFACE FROM THE HORIZONTAL - HORZ=0.0, VERT=90.0 - DEG
0.0
 INPUT DIR SURFACE FACES - DEG CNTR-CLKWISE FROM SOUTH - ENTER 0.0 FOR HORZ.
0.0
 INPUT THE OUTSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
3.0
 INPUT THE INSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
1.46
 INPUT THE REFLECTANCE FROM SURROUNDINGS
.2
 INPUT ABSORBANCE OF THE SURFACE
.8
 INPUT EMITTANCE OF THE SURFACE
.9
 INPUT SHADING COEFFICIENT OF GLASS
0.0
 INPUT ACTUAL OVERALL COND. OF WALL OR ROOF - BTU/HR-SQFT-F
.150
 INPUT OVERALL CONDUCTANCE OF WINDOWS - BTU/HR-SQFT-F
0.0
 INPUT OVERALL CONDUCTANCE OF DOORS - BTU/HR-SQFT-F
0.0
 ENTER SURFACE NUMBER OR 0
14
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 INPUT CONSTRUCTION TYPE
2
 INPUT LIGHT FIXTURE TYPE
2
 INPUT FLOOR TYPE
2
```

```
 INPUT CIRCULATION TYPE
2
 INPUT OUTSIDE PERIPHERAL LENGTH OF ZONE, FT
1718.
 INPUT MINIMUM ZONE RELATIVE HUMIDITY
.2
 INPUT MAXIMUM ZONE RELATIVE HUMIDITY
.3
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 ENTER SURFACE DESCRIPTION - DIR., ROOF, ETC
NORTH WALL
 INPUT WALL HGT OR ROOF LGTH - FT
384.0
 INPUT WALL WDTH OR ROOF WDTH - FT
20.0
 INPUT AREA OF THE DOORS - SQFT
342.
 INPUT THE AREA OF THE WINDOWS - SQFT
590.4
 INPUT TILT OF THE SURFACE FROM THE HORIZONTAL - HORZ=0.0, VERT=90.0 - DEG
90.0
 INPUT DIR SURFACE FACES - DEG CNTR-CLKWISE FROM SOUTH - ENTER 0.0 FOR HORZ.
180.
 INPUT THE OUTSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
3.0
 INPUT THE INSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
1.46
 INPUT THE REFLECTANCE FROM SURROUNDINGS
.2
 INPUT ABSORBANCE OF THE SURFACE
.8
 INPUT EMITTANCE OF THE SURFACE
.9
 INPUT SHADING COEFFICIENT OF GLASS
.96
 INPUT ACTUAL OVERALL COND. OF WALL OR ROOF - BTU/HR-SQFT-F
.425
 INPUT OVERALL CONDUCTANCE OF WINDOWS - BTU/HR-SQFT-F
.95
 INPUT OVERALL CONDUCTANCE OF DOORS - BTU/HR-SQFT-F
.6
 PRESS RETURN TO CONTINUE LIST

 ENTER SURFACE NUMBER OR 0
19
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 ENTER SURFACE DESCRIPTION - DIR., ROOF, ETC
SOUTH WALL
 INPUT WALL HGT OR ROOF LGTH - FT
394.1
```

```
 INPUT WALL WDTH OR ROOF WDTH - FT
15.0
 INPUT AREA OF THE DOORS - SQFT
492.6
 INPUT THE AREA OF THE WINDOWS - SQFT
0.0
 INPUT TILT OF THE SURFACE FROM THE HORIZONTAL - HORZ=0.0, VERT=90.0 - DEG
90.0
 INPUT DIR SURFACE FACES - DEG CNTR-CLKWISE FROM SOUTH - ENTER 0.0 FOR HORZ.
0.0
 INPUT THE OUTSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
3.0
 INPUT THE INSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
1.46
 INPUT THE REFLECTANCE FROM SURROUNDINGS
.2
 INPUT ABSORBANCE OF THE SURFACE
.8
 INPUT EMITTANCE OF THE SURFACE
.9
 INPUT SHADING COEFFICIENT OF GLASS
0.0
 INPUT ACTUAL OVERALL COND. OF WALL OR ROOF - BTU/HR-SQFT-F
.425
 INPUT OVERALL CONDUCTANCE OF WINDOWS - BTU/HR-SQFT-F
0.0
 INPUT OVERALL CONDUCTANCE OF DOORS - BTU/HR-SQFT-F
.6
 PRESS RETURN TO CONTINUE LIST

 ENTER SURFACE NUMBER OR 0
19
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 ENTER SURFACE DESCRIPTION - DIR., ROOF, ETC
EAST WALL
 INPUT WALL HGT OR ROOF LGTH - FT
748.4
 INPUT WALL WDTH OR ROOF WDTH - FT
20.0
 INPUT AREA OF THE DOORS - SQFT
1638.
 INPUT THE AREA OF THE WINDOWS - SQFT
396.1
 INPUT TILT OF THE SURFACE FROM THE HORIZONTAL - HORZ=0.0, VERT=90.0 - DEG
90.0
 INPUT DIR SURFACE FACES - DEG CNTR-CLKWISE FROM SOUTH - ENTER 0.0 FOR HORZ.
90.
 INPUT THE OUTSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
3.0
 INPUT THE INSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
1.46
```

```
 INPUT THE REFLECTANCE FROM SURROUNDINGS
.2
 INPUT ABSORBANCE OF THE SURFACE
.8
 INPUT EMITTANCE OF THE SURFACE
.9
 INPUT SHADING COEFFICIENT OF GLASS
.96
 INPUT ACTUAL OVERALL COND. OF WALL OR ROOF - BTU/HR-SQFT-F
.425
 INPUT OVERALL CONDUCTANCE OF WINDOWS - BTU/HR-SQFT-F
.95
 INPUT OVERALL CONDUCTANCE OF DOORS - BTU/HR-SQFT-F
.6
 PRESS RETURN TO CONTINUE LIST

 ENTER SURFACE NUMBER OR 0
19
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 ENTER SURFACE DESCRIPTION - DIR., ROOF, ETC
WEST WALL
 INPUT WALL HGT OR ROOF LGTH - FT
284.0
 INPUT WALL WDTH OR ROOF WDTH - FT
20.0
 INPUT AREA OF THE DOORS - SQFT
931.8
 INPUT THE AREA OF THE WINDOWS - SQFT
0.0
 INPUT TILT OF THE SURFACE FROM THE HORIZONTAL - HORZ=0.0, VERT=90.0 - DEG
90.0
 INPUT DIR SURFACE FACES - DEG CNTR-CLKWISE FROM SOUTH - ENTER 0.0 FOR HORZ.
270.
 INPUT THE OUTSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
3.0
 INPUT THE INSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
1.46
 INPUT THE REFLECTANCE FROM SURROUNDINGS
.2
 INPUT ABSORBANCE OF THE SURFACE
.8
 INPUT EMITTANCE OF THE SURFACE
.9
 INPUT SHADING COEFFICIENT OF GLASS
0.0
 INPUT ACTUAL OVERALL COND. OF WALL OR ROOF - BTU/HR-SQFT-F
.425
 INPUT OVERALL CONDUCTANCE OF WINDOWS - BTU/HR-SQFT-F
0.0
 INPUT OVERALL CONDUCTANCE OF DOORS - BTU/HR-SQFT-F
.6
```

```
 PRESS RETURN TO CONTINUE LIST

 ENTER SURFACE NUMBER OR 0
19
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 ENTER SURFACE DESCRIPTION - DIR., ROOF, ETC
ROOF
 INPUT WALL HGT OR ROOF LGTH - FT
617.
 INPUT WALL WDTH OR ROOF WDTH - FT
207.
 INPUT AREA OF THE DOORS - SQFT
0.0
 INPUT THE AREA OF THE WINDOWS - SQFT
0.0
 INPUT TILT OF THE SURFACE FROM THE HORIZONTAL - HORZ=0.0, VERT=90.0 - DEG
0.0
 INPUT DIR SURFACE FACES - DEG CNTR-CLKWISE FROM SOUTH - ENTER 0.0 FOR HORZ.
0.0
 INPUT THE OUTSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
3.0
 INPUT THE INSIDE HEAT TRANSFER COEFFICIENT - BTU/HR-SQFT-F
1.46
 INPUT THE REFLECTANCE FROM SURROUNDINGS
.2
 INPUT ABSORBANCE OF THE SURFACE
.8
 INPUT EMITTANCE OF THE SURFACE
.9
 INPUT SHADING COEFFICIENT OF GLASS
0.0
 INPUT ACTUAL OVERALL COND. OF WALL OR ROOF - BTU/HR-SQFT-F
.246
 INPUT OVERALL CONDUCTANCE OF WINDOWS - BTU/HR-SQFT-F
0.0
 INPUT OVERALL CONDUCTANCE OF DOORS - BTU/HR-SQFT-F
0.0
 ENTER SURFACE NUMBER OR 0
15
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 INPUT FIRST SHIFT START TIME - HRS
8.0
 INPUT SECOND SHIFT START TIME - HRS
17.0
 INPUT NUMBER OF 1ST SHIFT OCCUPANTS
30
 INPUT NUMBER OF 2ND SHIFT OCCUPANTS
3
 INPUT INFILTRATION CLASSIFICATION
2
```

```
 INPUT CEILING HEIGHT - FT
8.8
 INPUT FL. LIGHT POWER FOR 1ST SHIFT - KW
16.28
 INPUT FL. LIGHT POWER FOR 2ND SHIFT - KW
1.63
 INPUT INCND. LIGHTS FOR 1ST SHIFT - KW
0.0
 INPUT INCND. LIGHTS FOR 2ND SHIFT - KW
0.0
 INPUT INTERNAL SENSIBLE HEAT GAIN FOR 1ST SHIFT - BTU/HR
79400.
 INPUT INTERNAL SENSIBLE HEAT GAIN FOR 2ND SHIFT - BTU/HR
43400.
 INPUT INTERNAL LATENT HEAT GAIN FOR 1ST SHIFT - BTU/HR
0.0
 INPUT INTERNAL LATENT HEAT GAIN FOR 2ND SHIFT - BTU/HR
0.0
 INPUT ZONE OPERATING TEMPERATURE
72.
 INPUT 1 TO RUN HEATX SUBROUTINE
1
 INPUT 1 TO RUN HEATING AND COOLING EQUIPMENT MODEL
1
 INPUT THE NUMBER OF HOLIDAYS TO BE CONSIDERED
1
 INPUT HOLIDAY (REFER TO INSTRUCTIONS)
0704
 ARE THE HOLIDAYS CORRECT? (Y=0/N=1)

 INPUT 0 TO TREAT WEEKENDS AS WEEKDAYS
 1
  INPUT MAXIMUM CAPACITY OF THE AIR HANDLER - BTU/HR
 750000.
  INPUT MINIMUM CAPACITY OF THE AIR HANDLER - BTU/HR
 0.0
  INPUT TOTAL FLOOR AREA OF ZONE - SQFT
 16281.
  INPUT THERMOSTAT RANGE - THE DEAD BAND
 3.0
  INPUT THERMOSTAT SET TEMPERATURE DURING DAY TIME - F
 78.
  INPUT THERMOSTAT SET TEMPERATURE DURING NIGHT TIME - F
 85.
  INPUT THERMOSTAT SET TIME FOR 1ST SHIFT
 6.
  INPUT THERMOSTAT SET TIME FOR 2ND SHIFT
 22.
  ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

  INPUT FIRST SHIFT START TIME - HRS
 8.0
```

```
 INPUT SECOND SHIFT START TIME - HRS
17.0
 INPUT NUMBER OF 1ST SHIFT OCCUPANTS
55
 INPUT NUMBER OF 2ND SHIFT OCCUPANTS
6
 INPUT INFILTRATION CLASSIFICATION
2
 INPUT CEILING HEIGHT - FT
20.0
 INPUT FL. LIGHT POWER FOR 1ST SHIFT - KW
158.8
 INPUT FL. LIGHT POWER FOR 2ND SHIFT - KW
15.9
 INPUT INCND. LIGHTS FOR 1ST SHIFT - KW
0.0
 INPUT INCND. LIGHTS FOR 2ND SHIFT - KW
0.0
 INPUT INTERNAL SENSIBLE HEAT GAIN FOR 1ST SHIFT - BTU/HR
123800.
 INPUT INTERNAL SENSIBLE HEAT GAIN FOR 2ND SHIFT - BTU/HR
67400.
 INPUT INTERNAL LATENT HEAT GAIN FOR 1ST SHIFT - BTU/HR
0.0
 INPUT INTERNAL LATENT HEAT GAIN FOR 2ND SHIFT - BTU/HR
0.0
 INPUT ZONE OPERATING TEMPERATURE
78.
 INPUT 1 TO RUN HEATX SUBROUTINE
1
 INPUT 1 TO RUN HEATING AND COOLING EQUIPMENT MODEL
1
 INPUT THE NUMBER OF HOLIDAYS TO BE CONSIDERED
1
 INPUT HOLIDAY (REFER TO INSTRUCTIONS)
0704
 ARE THE HOLIDAYS CORRECT? (Y=0/N=1)

 INPUT 0 TO TREAT WEEKENDS AS WEEKDAYS
1
 INPUT MAXIMUM CAPACITY OF THE AIR HANDLER - BTU/HR
7000000.
 INPUT MINIMUM CAPACITY OF THE AIR HANDLER - BTU/HR
0.0
 INPUT TOTAL FLOOR AREA OF ZONE - SQFT
225886.
 INPUT THERMOSTAT RANGE - THE DEAD BAND
3.0
 INPUT THERMOSTAT SET TEMPERATURE DURING DAY TIME - F
78.
 INPUT THERMOSTAT SET TEMPERATURE DURING NIGHT TIME - F
85.
```

```
 INPUT THERMOSTAT SET TIME FOR 1ST SHIFT
6.0
 INPUT THERMOSTAT SET TIME FOR 2ND SHIFT
22.0
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 INPUT MONTH WHEN SIMULATION IS TO START
6
 INPUT THE DAY OF THE MONTH WHEN SIMULATION IS TO START
28
 INPUT MONTH WHEN SIMULATION IS TO END
8
 INPUT THE DAY OF THE MONTH WHEN SIMULATION IS TO END
31
 INPUT THE DAY OF THE WEEK FOR STARTING DAY
1
 ARE THESE PARAMETERS CORRECT? (Y=0/N=1)

 INPUT THE WEATHER FORECAST FILE
FILE4.DAT
 INPUT 1 FOR ANOTHER RUN FOR THIS BUILDING
0
```

APPENDIX B

PROGRAM LISTING

```
C     BUILDING THERMAL SYSTEM SIMULATION PROGRAM
C
C
C     UPDATED TO MULTIZONE BY MCQUISTON & HARRIS  DEC 1984
C       OUTPUT UPDATED BY MCQUISTON FEB 1985
C     THIS PROGRAM SIMULATES THE THERMAL PERFORMANCE OF
C     A BUILDING USING HOURLY WEATHER DATA AND PREDICTS THE
C     COOLING LOAD, HEAT EXTRACTION RATE AND ENERGY
C     REQUIRED WHEN AN EQUIPMENT MODEL IS ADDED.
C
C
      CHARACTER*80 ORIENT,ZORNT
      CHARACTER*80 FNAME
C
      COMMON /SOL1/ ATDB(24),SOLH(24),XID(24),XIDHV(24),XIT(24)
      COMMON /SOL2/ XLATR,EPR,PSIR,DLONG,IDAY1
      COMMON /SOL3/ SH(24),CH(24),CZ(24),CT(24),CE,SD,CD,SL,CL
      COMMON /BLOCK1/IIN,IOT,MC,NPRT,INWRIT,ZTROOM(4),ZOFST(4),INM2,
     1ZOFCT(4),ZCFMD(4),XKTZ(4),XLF(4),INITZ(4),INIHX,MONTH,IDAYM,
     &DAYTYP
      COMMON /BLOCK2/ QTOTAL(24),ER(4,48)
C
      COMMON /BLOCK3/MONTH1,MONTH2,MDAY1,MDAY2
      COMMON /BLOCK4/ZERMAX(4),ZERMIN(4),ZFLARA(4),ZTRANG(4),
     1ZTSETD(4),ZTSETN(4),ZTTIMN(4),ZTTIMD(4)
      COMMON /BLOCK6/ NZ,ERB(24)
C
      INTEGER HOLDAY(50),NHLDAY,DAYWK,DAYTYP
      COMMON /DUMMY1/ ZQTLD(4),ZQTLN(4),ZQFLD(4),ZQFLN(4)
      COMMON /DUMMY2/  DATR(4,24),VWIND(24)
      COMMON /DUMMY3/ PD(4),PS1(4),PS2(4),WSP1(4),WSP2(4)
      COMMON /DUMMY9/ FC1(4),ZQOTLD(4),QSLB(24),NWAR(4)
      COMMON /DUMY10/ ZNPD(4),ZNPN(4),ZQOTSN(4),ZQOTSD(4),ZQOTLN(4)
      COMMON /DUMY11/ QSSW(4,48),QSWR(24),Q34(48)
      COMMON /DUMY15/ SGV(4,3),CWRV(4,3),HGLV(4,3),RTFW(4,3),
     +HGEPRV(4,3),IBLDZ(4)
      COMMON /DUMY17/ ZWRL(4,10),ZWRW(4,10)
      COMMON /DUMY18/ ZEPR(4,10),ZPSIR(4,10),ZHO(4,10),ZHI(4,10)
      COMMON /DUMY19/ ZROG(4,10),ZEPSWR(4,10),ZSCG(4,10)
      COMMON /DUMY20/ ZUWRA(4,10),ZUW(4,10),ZUD(4,10)
      COMMON /DUMY21/ ZBT(4,10,7),ZUWRT(4,10),ZDT(4,10,7)
      COMMON /DUMY22/ ZXNI(4,10),ZCNS(4,10),ZCNST(4,10)
C
      COMMON /BLCK23/ BINF(3),CINF(3),SGV1(3),CWRV1(3),HGLV1(3)
      COMMON /BLCK24/ RTFW1(12),CRATFG(12),CRATFP(12)
      COMMON /BLCK25/ ACOEF(4),ALPAJ(8),TAUJ(8)
      COMMON /BLCK26/ NSTOP,NHTX,IENERG,NWKND,ML,MF,MA,PHI1,PHI2
      COMMON /BLCK27/ IFLAGD,XLAT,ACLONG,STLONG,PB,NHLDAY
C
```

```
      COMMON /DUMY30/ DUM,ALPAD,TAUD,SALPAJ,STAUJ,XJ,SHGTC,SHGAC
      COMMON /DUMY31/ ALPAW,EMEW,CFMD,CFM,XI,XNP,QOTS,QOTL
      COMMON /DUMY32/ QFLE,QTLE,QSFWT,QFWRT,QFFLT,QF34T,UWRT
C
      COMMON /BLCK35/ WRL,WRW,AD,AW,PSI,EPSILN,HO
      COMMON /BLCK36/ HI,ROG,ALPHWR,EPSWR,SCG,UWRA,UW,UD,BT(7),DT(7)
      COMMON /BLCK37/ NPN,NPD,IBLD,CLHGT,QOTSN,QOTSD,QOTLN,QOTLD,QFLN
      COMMON /BLCK38/ QFLD,QTLN,QTLD,OFST,OFCT,TROOM,ERMIN,ERMAX
      COMMON /BLCK39/ FLAREA,THRANG,THSETD,THSETN,THTIMN,THTIMD
      COMMON /BLCK40/ ACH(4,24),QTSHZ(24),QTSHL(24)
      COMMON /BLCK41/ QTOTAZ(24),QTOFHZ,QTOFCZ,Q30Z(24)
      COMMON /BLCK42/ QSFWZ(24),Q37Z(24),Q38Z(24),Q39Z(24),Q31Z(24)
      COMMON /BLCK43/ QOTHSZ(24),QOTHLZ(24),QIVSZ(24),QIVLZ(24)
      COMMON /BLCK44/ QPPLZ(24),QPPSZ(24),QFLSZ(24),QTLSZ(24)
      COMMON /BLCK45/ QSSWRZ(24),Q35Z(24)
      COMMON /BLCK46/ ZALP(4,10),ZORNT(4,10),TSWR(4,10,48)
      COMMON /BLCK47/ ZAW(4,10),SHGF(24),TSW(24),ZAWR(4,10)
      COMMON /BLCK48/ QEWR(4,10,48),QEW(24),ZAD(4,10),QED(24)
      COMMON /BLCK49/ QFLS(48),QTLS(24),QPPS(24),QOTHS(24),QIVS(24)
      COMMON /BLCK50/ QSSWR(4,48),Q35(24),QPPL(24),QOTHL(24),QIVL(24)
      COMMON /BLCK51/ Q30(24),Q31(24),Q37(4,48),Q38(4,48),Q39(4,48)
      COMMON /BLCK52/ QSFW(4,48),ATWB(24)
      COMMON /BLCK53/ DW(4,24)
      COMMON /BLCK54/ QTOTFC,QTOTFH
C
      DATA NAT,NSCG/6,0/
      DATA RTD/57.29578/
C
C
C  BEGINNING OF PROGRAM
C
      CALL SETINP
C
C  INITIALIZING PARAMETERS
C
      CALL INTIAL(DAYWK,HOLDAY,NZON,NWAR)
C
      FNAME='PRN'
      WRITE(*,1603)
 1603 FORMAT(' INPUT A FILE NAME FOR OUTPUT WHERE:',
     +/,8X,'CON = SCREEN OUTPUT',/,8X,
     +'PRN = PRINTER OUTPUT',/,8X,'??? = ANY OUTPUT FILE NAME',
     +' YOU CHOOSE',//)
      CALL INPUTT(FNAME,'INPUT FILE NAME $')
      OPEN(6,FILE=FNAME,STATUS='NEW')
C
 2001 CONTINUE
      WRITE(*,1604)
 1604 FORMAT('  INPUT 0,1, OR 2 TO SPECIFIY TYPE OF OUTPUT DE',
     +'SIRED WHERE:',/,'     (0)=TOTAL LOAD FOR EACH DAY, AND',
     +' FOR THE WHOLE PERIOD MONTH BY MONTH',/,'     (1)=(0) +',
     +' HOUR BY HOUR LOAD FOR EACH DAY',/,'     (2)=(1) + AT',
```

```
     +'MOSPHERIC DRY BULB (F), WET BULB (F), HUMIDITY DIFFERE',
     +'NCE (LBM OF             WATER / LBM OF AIR), SENSIBLE,',
     +' LATENT, AND TOTAL LOADS (BTU/HR)',/,'    (3)=(2) +',
     +' INDIVIDUAL CONTRIBUTIONS',/,'    (4)=(3) + SOL-AIR',
     +' TEMPERATURE, HEAT GAIN THROUGH EACH SURFACE - HOUR',
     +' BY HOUR                 - EVERY DAY',//)
      CALL INPUTI(NPRT,'INPUT OUTPUT TYPE $')
C
      WRITE(*,1605)
 1605 FORMAT(//,' INWRIT- TO WRITE INPUT DATA ENTER (1)',/,
     +8X,' OTHERWISE ENTER (0)',//)
      CALL INPUTI(INWRIT,'INPUT INWRIT $')
C
 1632 CALL INPUTI(NZON,'ENTER NUMBER OF ZONES $')
      DO 1606 I=1,NZON
      WRITE(*,1607)I
 1607 FORMAT(9(/),' INPUT NUMBER OF SURFACES FOR ZONE-',I2,'----> ',//)
      CALL INPUTI(NWAR(I),'ENTER THE NUMBER OF SURFACES FOR THIS ZONE $'
     +)
 1606 CONTINUE
C
      NCOR=0
      DO 1630 LL=1,NZON
      WRITE(*,1631)LL,NWAR(LL)
 1631 FORMAT(/,' ZONE ',I1,' NWAR=',I2,/)
 1630 CONTINUE
      CALL INPUTI(NCOR,'ARE THE ZONES AND THE CORRESPONDING NUMBER OF WA
     +LLS CORRECT? (Y=0/N=1) $')
      IF(NCOR.EQ.1)GO TO 1632
C
      CALL INPUTR(XLAT,'INPUT LOCAL LATITUDE - DEG $')
      CALL INPUTR(ACLONG,'INPUT LOCAL LONGITUDE - DEG $')
      CALL INPUTR(STLONG,'INPUT STANDARD LONGITUDE - DEG (75,90,105,120,
     +ETC) $')
      DLONG=(STLONG-ACLONG)/15.0
C
      IF (INWRIT.LE.0) GO TO 10
      WRITE (IOT,520)
C
      WRITE (*,1707)NPRT,INWRIT,XLAT,ACLONG,STLONG
 1707 FORMAT(/,' NPRT=',I1,' INWRIT=',I1,' XLAT=',F5.1,
     +' ACLONG=',F5.1,' STLONG=',F5.1,//)
      NCOR=0
      CALL INPUTI(NCOR,'ARE THESE PARAMETERS CORRECT? (Y=0/N=1) $')
      IF(NCOR.EQ.1) GO TO 2001
C
 10   XLATR=XLAT/RTD
      IF(NHLDAY.LT.50)HOLDAY(NHLDAY+1)=0
      DO 11 N=1,NZON
      IF (NWAR(N).LE.10) GO TO 18
 11   WRITE (*,12) NWAR(N)
 12   FORMAT('0*** ERROR *** NWAR = ',I6,' EXCEEDS THE MAXIMUM ALLOWABLE
```

```
           1 VALUE OF 10')
            GO TO 490
      18    DO 60 NZ=1,NZON
1     2002  CONTINUE
1     C
1           WRITE(*,1608)NZ
1     1608  FORMAT(//,' *** YOU ARE NOW ENTERING DATA FOR ZONE ',I2,'. ***',
1          +//)
1           WRITE(*,1609)
1     1609  FORMAT('  TYPES OF CONSTRUCTION ARE (1=LIGHTWEIGHT,',
1          +' 2=MEDIUM WEIGHT, 3=HEAVYWEIGHT)',//)
1           CALL INPUTI(MC,'INPUT CONSTRUCTION TYPE $')
1           CALL INPUTI(ML,'INPUT LIGHT FIXTURE TYPE $')
1           CALL INPUTI(MF,'INPUT FLOOR TYPE $')
1           CALL INPUTI(MA,'INPUT CIRCULATION TYPE $')
1     C
1           CALL INPUTR(XLF(NZ),'INPUT OUTSIDE PERIPHERAL LENGTH OF ZONE, FT $
1          +')
1           CALL INPUTR(PHI1,'INPUT MINIMUM ZONE RELATIVE HUMIDITY $')
1           CALL INPUTR(PHI2,'INPUT MAXIMUM ZONE RELATIVE HUMIDITY $')
1     C
1           WRITE(*,1706)MC,ML,MF,MA,XLF(NZ),PHI1,PHI2
1     1706  FORMAT(/,' MC=',I1,' ML=',I1,' MF=',I1,' MA=',I1,/,
1          +' XLF=',F7.1,' PHI1=',F6.4,' PHI2=',F6.4,//)
1           NCOR=0
1           CALL INPUTI(NCOR,'ARE THESE PARAMETERS CORRECT? (Y=0/N=1) $')
1           IF(NCOR.EQ.1) GO TO 2002
1           XKT=0.0
1           XKTZ(NZ)=0.0
1           DO 50 NS=1,NWAR(NZ)
2     C
2           CALL TWINT
2     2004  CONTINUE
2           WRITE(*,2003) NZ,NS
2     2003  FORMAT(/,1X,'YOU ARE NOW ENTERING DATA FOR ZONE ',I2,'  SURFACE '
2          +,I2)
2     C
2           CALL INPUTT(ORIENT,'ENTER SURFACE DESCRIPTION - DIR., ROOF, ETC $'
2          +)
2           CALL INPUTR(WRL,'INPUT WALL HGT OR ROOF LGTH - FT $')
2           CALL INPUTR(WRW,'INPUT WALL WDTH OR ROOF WDTH - FT $')
2           CALL INPUTR(AD,'INPUT AREA OF THE DOORS - SQFT $')
2           CALL INPUTR(AW,'INPUT THE AREA OF THE WINDOWS - SQFT $')
2           CALL INPUTR(EPSILN,'INPUT TILT OF THE SURFACE FROM THE HORIZONTAL
2          +- HORZ=0.0, VERT=90.0 - DEG $')
2           CALL INPUTR(PSI,'INPUT DIR SURFACE FACES - DEG CNTR-CLKWISE FROM S
2          +OUTH - ENTER 0.0 FOR HORZ. $')
2           CALL INPUTR(HO,'INPUT THE OUTSIDE HEAT TRANSFER COEFFICIENT - BTU/
2          +HR-SQFT-F $')
2           CALL INPUTR(HI,'INPUT THE INSIDE HEAT TRANSFER COEFFICIENT - BTU/H
2          +R-SQFT-F $')
2           CALL INPUTR(ROG,'INPUT THE REFLECTANCE FROM SURROUNDINGS $')
```

```
      CALL INPUTR(ALPHWR,'INPUT ABSORBANCE OF THE SURFACE $')
      CALL INPUTR(EPSWR,'INPUT EMITTANCE OF THE SURFACE $')
      CALL INPUTR(SCG,'INPUT SHADING COEFFICIENT OF GLASS $')
      CALL INPUTR(UWRA,'INPUT ACTUAL OVERALL COND. OF WALL OR ROOF - BTU
     +/HR-SQFT-F $')
      CALL INPUTR(UW,'INPUT OVERALL CONDUCTANCE OF WINDOWS - BTU/HR-SQFT
     +-F $')
      CALL INPUTR(UD,'INPUT OVERALL CONDUCTANCE OF DOORS - BTU/HR-SQFT-F
     + $')
C
      UWRT=UWRA
C
      CALL TRANSF(BT,DT,UWRT,EPSILN,MC)
C
      IF (UWRT.GT.0.0.AND.UWRA.EQ.0.0) UWRA=UWRT
      IF (INWRIT.LE.0) GO TO 30
      WRITE (IOT,510)
C
      WRITE (*,1705)WRL,WRW,AD,AW,EPSILN,PSI,HO,HI,ROG,
     +ALPHWR,EPSWR,SCG,UWRA,UW,UD,ORIENT
 1705 FORMAT(/,' WRL=',F6.1,' WRW=',F6.1,' AD=',F7.1,' AW=',F7.1,/,
     +' EPSILN=',F5.1,' PSI=',F5.1,' HO=',F7.1,' HI=',F7.1,/,
     +' ROG=',F3.1,' ALPHWR=',F3.1,' EPSWR=',F3.1,' SCG=',F4.2,/,
     +' UWRA=',F6.1,' UW=',F6.1,' UD=',F6.1,' ORIENT=',A80,//)
      NCOR=0
      CALL INPUTI(NCOR,'ARE THESE PARAMETERS CORRECT? (Y=0/N=1) $')
      IF(NCOR.EQ.1) GO TO 2004
C
      WRITE (IOT,510)
C
C     WRITE (IOT,1704)(BT(I),DT(I),I=1,7),UWRT
C1704  FORMAT(7(/,' BT=',E13.6,' DT=',E13.6),//,' UWRT=',F9.6,//)
C
 30   ZWRL(NZ,NS)=WRL
      ZWRW(NZ,NS)=WRW
      ZAD(NZ,NS)=AD
      ZAW(NZ,NS)=AW
      ZEPR(NZ,NS)=EPSILN/RTD
      ZPSIR(NZ,NS)=PSI/RTD
      ZHO(NZ,NS)=HO
      ZHI(NZ,NS)=HI
      ZROG(NZ,NS)=ROG
      ZALP(NZ,NS)=ALPHWR
      ZEPSWR(NZ,NS)=EPSWR
      ZSCG(NZ,NS)=SCG
      ZUWRA(NZ,NS)=UWRA
      ZUW(NZ,NS)=UW
      ZUD(NZ,NS)=UD
      ZORNT(NZ,NS)=ORIENT
      ZAWR(NZ,NS)=(WRL*WRW)-AD-AW
      XKT=XKT+(ZAWR(NZ,NS)*UWRA)+(AD*UD)+(AW*UW)
      UR=UWRA/UWRT
```

```
2          ZXNI(NZ,NS)=HI/(HI+HO)
2          ZCNS(NZ,NS)=0.0
2          DO 40 I=1,7
3          ZBT(NZ,NS,I)=BT(I)*UR
3          ZDT(NZ,NS,I)=DT(I)
3          ZCNS(NZ,NS)=ZCNS(NZ,NS)+ZBT(NZ,NS,I)
3    40    CONTINUE
2          ZDT(NZ,NS,1)=0.0
2    50    CONTINUE
1          XKTZ(NZ)=XKT/XLF(NZ)
1    C     FC1(NZ)=1.0-(0.02*XKTZ(NZ))
1          FC1(NZ)=1.0
1          M=((4*MA)-3)+(MF-1)
1          RTFW(NZ,1)=0.0
1          RTFW(NZ,2)=RTFW1(M)
1          RTFW(NZ,3)=0.0
1          SGV(NZ,1)=SGV1(MC)*FC1(NZ)
1          SGV(NZ,2)=FC1(NZ)*(1.+RTFW1(M)-SGV1(MC))
1          SGV(NZ,3)=0.0
1          CWRV(NZ,1)=CWRV1(MC)*FC1(NZ)
1          CWRV(NZ,2)=FC1(NZ)*(1.+RTFW1(M)-CWRV1(MC))
1          CWRV(NZ,3)=0.0
1          HGLV(NZ,1)=HGLV1(MC)*FC1(NZ)
1          HGLV(NZ,2)=FC1(NZ)*ACOEF(ML)
1          HGLV(NZ,3)=FC1(NZ)*(1.+RTFW1(M)-ACOEF(ML))
1          HGEPRV(NZ,1)=SGV(NZ,1)
1          HGEPRV(NZ,2)=SGV(NZ,2)
1          HGEPRV(NZ,3)=0.0
1    60    CONTINUE
     61    CONTINUE
           DO 95 NZ=1,NZON
1          IF(NZ.NE.1)GO TO 70
1    C
1          CALL THRINT
1    70    CONTINUE
1    2006  CONTINUE
1          WRITE(*,2005)NZ
1    2005  FORMAT(/,1X,'YOU ARE NOW ENTERING DATA FOR ZONE ',I2,/)
1    C
1          CALL INPUTR(OFST,'INPUT FIRST SHIFT START TIME - HRS $')
1          CALL INPUTR(OFCT,'INPUT SECOND SHIFT START TIME - HRS $')
1          CALL INPUTI(NPD,'INPUT NUMBER OF 1ST SHIFT OCCUPANTS $')
1          CALL INPUTI(NPN,'INPUT NUMBER OF 2ND SHIFT OCCUPANTS $')
1          CALL INPUTI(IBLD,'INPUT INFILTRATION CLASSIFICATION $')
1          CALL INPUTR(CLHGT,'INPUT CEILING HEIGHT - FT $')
1          CALL INPUTR(QFLD,'INPUT FL. LIGHT POWER FOR 1ST SHIFT - KW $')
1          CALL INPUTR(QFLN,'INPUT FL. LIGHT POWER FOR 2ND SHIFT - KW $')
1          CALL INPUTR(QTLD,'INPUT INCND. LIGHTS FOR 1ST SHIFT - KW $')
1          CALL INPUTR(QTLN,'INPUT INCND. LIGHTS FOR 2ND SHIFT - KW $')
1          CALL INPUTR(QOTSD,'INPUT INTERNAL SENSIBLE HEAT GAIN FOR 1ST SHIFT
1         + - BTU/HR $')
1          CALL INPUTR(QOTSN,'INPUT INTERNAL SENSIBLE HEAT GAIN FOR 2ND SHIFT
```

```
1       + - BTU/HR $')
1        CALL INPUTR(QOTLD,'INPUT INTERNAL LATENT HEAT GAIN FOR 1ST SHIFT -
1       + BTU/HR $')
1        CALL INPUTR(QOTLN,'INPUT INTERNAL LATENT HEAT GAIN FOR 2ND SHIFT -
1       + BTU/HR $')
1        CALL INPUTR(TROOM,'INPUT ZONE OPERATING TEMPERATURE $')
1        CALL INPUTI(NHTX,'INPUT 1 TO RUN HEATX SUBROUTINE $')
1        CALL INPUTI(IENERG,'INPUT 1 TO RUN HEATING AND COOLING EQUIPMENT M
1       +ODEL $')
1  1635  CALL INPUTI(NHLDAY,'INPUT THE NUMBER OF HOLIDAYS TO BE CONSIDERED
1       +$')
1        DO 1610 J=1,NHLDAY
2  1610  CALL INPUTI(HOLDAY(J),'INPUT HOLIDAY (REFER TO INSTRUCTIONS) $')
1  C
1        NCOR=0
1        DO 1633 LL=1,NHLDAY
2        WRITE(*,1634)LL,HOLDAY(LL)
2  1634  FORMAT(/,' NHLDAY=',I2,' HOLDAY=',I5)
2  1633  CONTINUE
1        CALL INPUTI(NCOR,'ARE THE HOLIDAYS CORRECT? (Y=0/N=1) $')
1        IF(NCOR.EQ.1)GO TO 1635
1  C
1        CALL INPUTI(NWKND,'INPUT 0 TO TREAT WEEKENDS AS WEEKDAYS $')
1  C
1        IF(NHTX.EQ.0) GO TO 1611
1        CALL INPUTR(ERMAX,'INPUT MAXIMUM CAPACITY OF THE AIR HANDLER - BTU
1       +/HR $')
1        CALL INPUTR(ERMIN,'INPUT MINIMUM CAPACITY OF THE AIR HANDLER - BTU
1       +/HR $')
1        CALL INPUTR(FLAREA,'INPUT TOTAL FLOOR AREA OF ZONE - SQFT $')
1        CALL INPUTR(THRANG,'INPUT THERMOSTAT RANGE - THE DEAD BAND $')
1        CALL INPUTR(THSETD,'INPUT THERMOSTAT SET TEMPERATURE DURING DAY TI
1       +ME - F $')
1        CALL INPUTR(THSETN,'INPUT THERMOSTAT SET TEMPERATURE DURING NIGHT
1       +TIME - F $')
1        CALL INPUTR(THTIMD,'INPUT THERMOSTAT SET TIME FOR 1ST SHIFT $')
1        CALL INPUTR(THTIMN,'INPUT THERMOSTAT SET TIME FOR 2ND SHIFT $')
1  C
1  1611  ZNPN(NZ)=NPN
1        ZNPD(NZ)=NPD
1        IBLDZ(NZ)=IBLD
1        ZQOTSN(NZ)=QOTSN
1        ZQOTSD(NZ)=QOTSD
1        ZQOTLN(NZ)=QOTLN
1        ZQOTLD(NZ)=QOTLD
1        ZQFLN(NZ)=QFLN
1        ZQFLD(NZ)=QFLD
1        ZQTLN(NZ)=QTLN
1        ZQTLD(NZ)=QTLD
1        ZOFST(NZ)=OFST
1        ZOFCT(NZ)=OFCT
1        ZTROOM(NZ)=TROOM
1        ZERMIN(NZ)=ERMIN
```

```
1          ZERMAX(NZ)=ERMAX
1          ZFLARA(NZ)=FLAREA
1          ZTRANG(NZ)=THRANG
1          ZTSETD(NZ)=THSETD
1          ZTSETN(NZ)=THSETN
1          ZTTIMN(NZ)=THTIMN
1          ZTTIMD(NZ)=THTIMD
1          IF (NHLDAY.LT.50) HOLDAY(NHLDAY+1)=0
1          PD(NZ)=PSL(TROOM)
1          PS1(NZ)=PHI1*PD(NZ)
1          PS2(NZ)=PHI2*PD(NZ)
1          WSP1(NZ)=0.622*PS1(NZ)/(PB-PS1(NZ))
1          WSP2(NZ)=0.622*PS2(NZ)/(PB-PS2(NZ))
1          DO 80 NS=1,NWAR(NZ)
2          ZCNST(NZ,NS)=ZCNS(NZ,NS)*TROOM
2    80    CONTINUE
1          IF (INWRIT.LE.0) GO TO 95
1          WRITE (IOT,510)
1          WRITE (*,1703)NPD,NPN,QOTSD,QOTSN,QOTLD,QOTLN,QFLD,QFLN,
1         +QTLD,QTLN,OFST,OFCT,TROOM,IENERG,NHLDAY,
1         +NWKND,NHTX,IBLD,CLHGT
1    1703  FORMAT(/,' NPD=',I3,' NPN=',I3,' QOTSD=',F8.1,' QOTSN=',F9.1,
1         +/,' QOTLD=',F9.1,' QOTLN=',F9.1,' QFLD=',F9.1,' QFLN=',F9.1,/,
1         +' QTLD=',F9.1,' QTLN=',F9.1,' OFST=',F4.1,' OFCT=',F4.1,/,
1         +' TROOM=',F5.1,' IENERG=',I1,' NHLDAY=',I2,/,
1         +' NWKND=',I1,' NHTX=',I1,
1         +' IBLD=',I1,' CLHGT=',F4.1,//)
1    C
1    C
1          IF(NHTX.NE.0)  WRITE (*,1701)ERMAX,ERMIN,FLAREA,THRANG,THSETD,
1         +THSETN,THTIMN,THTIMD
1    1701  FORMAT(/,' ERMAX=',F11.1,' ERMIN=',F11.1,' FLAREA=',F9.1,
1         +' THRANG=',F9.1,/,' THSETD=',F9.1,' THSETN=',F9.1,' THTIMN=',
1         +F9.1,' THTIMD=',F9.1,//)
1          NCOR=0
1          CALL INPUTI(NCOR,'ARE THESE PARAMETERS CORRECT? (Y=0/N=1) $')
1          IF(NCOR.EQ.1) GO TO 2006
1    95    CONTINUE
     2007  CONTINUE
           WRITE(*,2008)
     2008  FORMAT(/////////,5X,'*****  BUILDING SIMULATION DATA  *****',//)
           CALL INPUTI(MONTH1,'INPUT MONTH WHEN SIMULATION IS TO START $')
           CALL INPUTI(MDAY1,'INPUT THE DAY OF THE MONTH WHEN SIMULATION IS T
          +O START $')
           CALL INPUTI(MONTH2,'INPUT MONTH WHEN SIMULATION IS TO END $')
           CALL INPUTI(MDAY2,'INPUT THE DAY OF THE MONTH WHEN SIMULATION IS T
          +O END $')
           CALL INPUTI(DAYWK,'INPUT THE DAY OF THE WEEK FOR STARTING DAY $')
           WRITE(*,1702) MONTH1,MDAY1,MONTH2,MDAY2,DAYWK
     1702  FORMAT(/,' MONTH1=',I2,' MDAY1=',I2,' MONTH2=',I2,' MDAY2=',I2,
          +' DAYWK=',I1,//)
           NCOR=0
           CALL INPUTI(NCOR,'ARE THESE PARAMETERS CORRECT? (Y=0/N=1) $')
```

```
          IF(NCOR.EQ.1) GO TO 2007
          INIHX=1
          INIT=1
          CALL JULIAN (MONTH1,MDAY1,MSTART)
          CALL JULIAN (MONTH2,MDAY2,MEND)
          IF (MEND.LT.MSTART) MEND=MEND+365
          DAYWK=DAYWK-1
          FNAME='FILE4.DAT'
          CALL INPUTT(FNAME,'INPUT THE WEATHER FORECAST FILE $')
     C
          DO 480 IDAYY=MSTART,MEND
1         CALL MONDAY (MONTH,IDAYM,IDAYY)
1         CALL WETHER (IDAYY,ATDB,ATWB,SOLH,VWIND,FNAME)
1         WRITE(*,*)' '
1         DAYWK=DAYWK+1
1         IF(DAYWK.EQ.8) DAYWK=1
1         DAYTYP=1
1         IF(NWKND.EQ.0) GO TO 105
1         IF(DAYWK.EQ.1.OR.DAYWK.EQ.7) DAYTYP=2
1         IF (DAYTYP.EQ.1.AND.NHLDAY.GT.0)
1        1 CALL HDAY(MONTH,IDAYM,HOLDAY,NHLDAY,DAYTYP)
1     105 CONTINUE
1         DO 110 I=1,24
2         DO 111 NZ=1,NZON
3         DATR(NZ,I)=ATDB(I)-ZTROOM(NZ)
3         WSPA=0.0
3         PSA=0.0
3         DW(NZ,I)=0.0
3         IF (ATDB(I).LT.10.0.OR.ATWB(I).LT.10.0) GO TO 100
3         WRITE(*,*)' '
3         PW=PSL(ATWB(I))
3         WW=0.622*PW/(PB-PW)
3         WSPA=(((1093.0-0.556*ATWB(I))*WW)-0.24*(ATDB(I)-ATWB(I)))/(1093.0+
3        1(0.444*ATDB(I))-ATWB(I))
3         IF (WSPA.LT.0.0) WSPA=0.0
3         PSA=WSPA*PB/(0.622+WSPA)
3         IF (PSA.GT.PHI2) DW(NZ,I)=WSPA-WSP2(NZ)
3         IF (PSA.LT.PHI1) DW(NZ,I)=WSPA-WSP1(NZ)
3     100 CONTINUE
3         QSSWR(NZ,I+24)=0.0
3         QSSW(NZ,I+24)=0.0
3         WRITE(*,*)' '
3     111 CONTINUE
2         ERB(I)=0.0
2     110 CONTINUE
1         SERMAX=0.0
1         QTOFCZ=0.0
1         QTOFHZ=0.0
1         DO 475 NZ=1,NZON
2         QTOTFC=0.0
2         QTOTFH=0.0
2    C
2         CALL FORINT(NZ,BINF,IBLDZ,VWIND,CINF,DATR,QTOTAL)
```

```
2     C
2           WRITE(*,*)' '
2           IDAY1=0
2           DO 290 NS=1,NWAR(NZ)
3           EPR=ZEPR(NZ,NS)
3           PSIR=ZPSIR(NZ,NS)
3           CALL SOLAR (IDAYY)
3           WRITE(*,*)' '
3           DO 160 IH=25,48
4           I=IH-24
4           QEW(I)=0.0
4           IF (ZAW(NZ,NS).LE.0.0) GO TO 150
4           ALPAD=ALPAJ(1)
4           TAUD=TAUJ(1)
4           DUM=1.0
4           DO 120 J=2,NAT
5           DUM=DUM*CT(I)
5           ALPAD=ALPAD+(ALPAJ(J)*DUM)
5           TAUD=TAUD+(TAUJ(J)*DUM)
5           WRITE(*,*)' '
5     120   CONTINUE
4           SALPAJ=0.0
4           STAUJ=0.0
4           DO 130 J=1,NAT
5           XJ=1.0/(J+1)
5           SALPAJ=SALPAJ+(ALPAJ(J)*XJ)
5           STAUJ=STAUJ+(TAUJ(J)*XJ)
5     130   CONTINUE
4           SHGTC=(XID(I)*TAUD)+(XIDHV(I)*2.0*STAUJ)
4           SHGAC=(XID(I)*ALPAD)+(XIDHV(I)*2.0*SALPAJ)
4           ALPAW=0.0
4           WRITE(*,*)' '
4           IF (SHGAC.LE.0.0.OR.XIT(I).LE.0.0) GO TO 140
4           ALPAW=SHGAC/XIT(I)
4     140   EMEW=ALPAW
4           SHGF(I)=SHGTC+ZXNI(NZ,NS)*SHGAC
4           QEW(I)=ZAW(NZ,NS)*(ZUW(NZ,NS)*DATR(NZ,I)+ZSCG(NZ,NS)*SHGF(I))
4     C
4     C     SOL-AIR TEMPERATURE CALCULATIONS
4     C
4           WRITE(*,*)' '
4           TSW(I)=ATDB(I)+(ALPAW*XIT(I)/ZHO(NZ,NS))-(EMEW*20.0*CE/ZHO(NZ,NS))
4     150   TSWR(NZ,NS,IH)=ATDB(I)+(ZALP(NZ,NS)*XIT(I)/ZHO(NZ,NS))-(ZEPSWR(NZ,
4          1NS)*20.0*CE/ZHO(NZ,NS))
4     160   CONTINUE
3           IF (INITZ(NZ).EQ.0) GO TO 180
3           DO 170 I=1,24
4           WRITE(*,*)' '
4           IH=I+24
4           QEWR(NZ,NS,I)=ZUWRA(NZ,NS)*(TSWR(NZ,NS,IH)-ZTROOM(NZ))
4           TSWR(NZ,NS,I)=TSWR(NZ,NS,IH)
4     170   CONTINUE
3     180   DO 200 K=25,48
```

```
4          QEWRT=ZBT(NZ,NS,1)*TSWR(NZ,NS,K)
4          DO 190 J=2,7
5          JJ=K+1-J
5          QEWRT=(ZBT(NZ,NS,J)*TSWR(NZ,NS,JJ))-(ZDT(NZ,NS,J)*QEWR(NZ,NS,JJ))+
5         +QEWRT
5    190   CONTINUE
4          QEWR(NZ,NS,K)=QEWRT-ZCNST(NZ,NS)
4    200   CONTINUE
3          DO 210 I=1,24
4          WRITE(*,*)' '
4          IH=I+24
4          QEWR(NZ,NS,I)=ZAWR(NZ,NS)*QEWR(NZ,NS,IH)
4          QED(I)=ZAD(NZ,NS)*ZUD(NZ,NS)*(TSWR(NZ,NS,IH)-ZTROOM(NZ))
4          QLTH=0.833*ATDB(I)-50.0
4          IF(QLTH.GT.0.0)QLTH=0.0
4          QSLB(I)=QLTH*XLF(NZ)
4          IF(NS.GT.1)QSLB(I)=0.0
4          QSWR(I)=QEWR(NZ,NS,I)+QED(I)+QEW(I)+QSLB(I)
4          QSSWR(NZ,IH)=QSSWR(NZ,IH)+QSWR(I)
4          TSWR(NZ,NS,I)=TSWR(NZ,NS,IH)
4    210   CONTINUE
3          WRITE(*,*)' '
3          IF (ZAW(NZ,NS).LE.0.0.OR.ZSCG(NZ,NS).LT.0.8) GO TO 230
3          NSCG=1
3          DO 220 I=1,24
4          IH=I+24
4          QSSW(NZ,IH)=QSSW(NZ,IH)+QEW(I)
4    220   CONTINUE
3          WRITE(*,*)' '
3    230   CONTINUE
3        C
3          CALL WRITE1(NZ,NS,NPRT,IOT,IH,XIT)
3        C
3          WRITE(*,*)' '
3    290   CONTINUE
2        C
2        C CALCULATION OF HEAT GAIN DUE TO PEOPLE, LIGHTS, OTHER EQUIPMENT,
2        C VENTILATION AND INFILTRATION
2        C
2          CFMD=0.0
2          DO 310 I=1,24
3          IH=I+24
3          XI=I
3          CIVL=4840.0*DW(NZ,I)
3          CFM=(ACH(NZ,I)-0.25)*(ZFLARA(NZ)*CLHGT/60.0)
3          XNP=ZNPN(NZ)
3          QOTS=ZQOTSN(NZ)
3          QOTL=ZQOTLN(NZ)
3          QFLE=ZQFLN(NZ)
3          QTLE=ZQTLN(NZ)
3          IF((XI.LE.ZOFST(NZ).OR.XI.GT.ZOFCT(NZ)).OR.(DAYTYP.EQ.2))GO TO 300
3        C
3        C OFFICE HOURS
```

```
3     C
3           WRITE(*,*)' '
3           CFM=ACH(NZ,I)*ZFLARA(NZ)*CLHGT/60.0
3           XNP=ZNPD(NZ)
3           QOTS=ZQOTSD(NZ)
3           QOTL=ZQOTLD(NZ)
3           QFLE=ZQFLD(NZ)
3           QTLE=ZQTLD(NZ)
3     C
3     C     OFF OFFICE HOURS
3     C
3       300 QFLS(IH)=4095.6*QFLE
3           QTLS(I)=3413.0*QTLE
3           QTLSZ(I)=QTLSZ(I)+QTLS(I)
3           QPPS(I)=250.0*XNP
3           QPPL(I)=200.0*XNP
3           QPPSZ(I)=QPPSZ(I)+QPPS(I)
3           QPPLZ(I)=QPPLZ(I)+QPPL(I)
3           QOTHS(I)=QOTS
3           QOTHL(I)=QOTL
3           QOTHSZ(I)=QOTHSZ(I)+QOTHS(I)
3           QOTHLZ(I)=QOTHLZ(I)+QOTHL(I)
3           QIVS(I)=CFM*1.08*DATR(NZ,I)
3           QIVL(I)=CFM*CIVL
3           Q30(I)=QIVL(I)+QPPL(I)+QOTHL(I)
3           QIVSZ(I)=QIVSZ(I)+QIVS(I)
3           QIVLZ(I)=QIVLZ(I)+QIVL(I)
3           Q31(I)=QTLS(I)+QIVS(I)+(0.5*QPPS(I))+QOTHS(I)
3           Q34(IH)=0.5*QPPS(I)
3           Q35(I)=Q31(I)+QFLS(IH)+QSSWR(NZ,IH)+Q34(IH)
3           Q30Z(I)=Q30Z(I)+Q30(I)
3           Q31Z(I)=Q31Z(I)+Q31(I)
3           Q35Z(I)=Q35Z(I)+Q35(I)
3           CFMD=CFMD+CFM
3           WRITE(*,*)' '
3       310 CONTINUE
2           ZCFMD(NZ)=CFMD/24.0
2     C
2     C     SENSIBLE COOLING LOAD DUE TO UNSHADED WINDOWS
2     C
2           WRITE(*,*)' '
2           IF (INITZ(NZ).EQ.0) GO TO 330
2           DO 320 I=1,24
3           IH=I+24
3           WRITE(*,*)' '
3           QFLS(I)=QFLS(IH)
3           QSSWR(NZ,I)=QSSWR(NZ,IH)
3           QSSW(NZ,I)=QSSW(NZ,IH)
3           Q34(I)=Q34(IH)
3           QSFW(NZ,I)=QSSW(NZ,IH)
3           Q37(NZ,I)=QFLS(IH)
3           Q38(NZ,I)=QSSWR(NZ,IH)
3           Q39(NZ,I)=Q34(IH)
```

```
3          QSFW(NZ,IH)=0.0
3      320 CONTINUE
2      330 IF (NSCG.NE.1) GO TO 370
2          DO 340 I=25,48
3          QSSWR(NZ,I)=QSSWR(NZ,I)-QSSW(NZ,I)
3      340 CONTINUE
2          DO 360 K=25,48
3          WRITE(*,*)' '
3          QSFWT=0.0
3          DO 350 J=1,3
4          JJ=K+1-J
4          QSFWT=QSFWT+SGV(NZ,J)*QSSW(NZ,JJ)-RTFW(NZ,J)*QSFW(NZ,JJ)
4      350 CONTINUE
3          QSFW(NZ,K)=QSFWT
3          WRITE(*,*)' '
3      360 CONTINUE
2        C
2        C         SENSIBLE COOLING LOAD DUE TO LIGHTS,SURFACES,AND RAD.FRAC. OF PEOP
2        C
2          WRITE(*,*)' '
2      370 DO 390 K=25,48
3          QFWRT=0.0
3          QFFLT=0.0
3          QF34T=0.0
3          DO 380 J=1,3
4          JJ=K+1-J
4          QFFLT=QFFLT+HGLV(NZ,J)*QFLS(JJ)-RTFW(NZ,J)*Q37(NZ,JJ)
4          QFWRT=QFWRT+CWRV(NZ,J)*QSSWR(NZ,JJ)-RTFW(NZ,J)*Q38(NZ,JJ)
4          QF34T=QF34T+HGEPRV(NZ,J)*Q34(JJ)-RTFW(NZ,J)*Q39(NZ,JJ)
4      380 CONTINUE
3          Q37(NZ,K)=QFFLT
3          Q38(NZ,K)=QFWRT
3          Q39(NZ,K)=QF34T
3          WRITE(*,*)' '
3      390 CONTINUE
2          DO 400 I=1,24
3          IH=I+24
3          QSSW(NZ,I)=QSSW(NZ,IH)
3          QSSWR(NZ,I)=QSSWR(NZ,IH)
3          QSSWRZ(I)=QSSWRZ(I)+QSSWR(NZ,I)
3          Q37(NZ,I)=Q37(NZ,IH)
3          Q38(NZ,I)=Q38(NZ,IH)
3          Q39(NZ,I)=Q39(NZ,IH)
3          Q37Z(I)=Q37Z(I)+Q37(NZ,I)
3          Q38Z(I)=Q38Z(I)+Q38(NZ,I)
3          Q39Z(I)=Q39Z(I)+Q39(NZ,I)
3          QSFW(NZ,I)=QSFW(NZ,IH)
3          QSFWZ(I)=QSFWZ(I)+QSFW(NZ,I)
3          QFLS(I)=QFLS(IH)
3          QFLSZ(I)=QFLSZ(I)+QFLS(I)
3          Q34(I)=Q34(IH)
3      400 CONTINUE
2          WRITE(*,*)' '
```

```
2            QTOTFC=0.0
2            QTOTFH=0.0
2            DO 410 I=1,24
3            QTSHL(I)=QSFW(NZ,I)+Q37(NZ,I)+Q38(NZ,I)+Q39(NZ,I)+Q31(I)
3            QTOTAL(I)=QTSHL(I)+Q30(I)
3            QTSHZ(I)=QTSHZ(I)+QTSHL(I)
3            QTOTAZ(I)=QTOTAZ(I)+QTOTAL(I)
3            IF (QTOTAL(I).GT.0.0) QTOTFC=QTOTFC+QTOTAL(I)
3            IF (QTOTAL(I).LT.0.0) QTOTFH=QTOTFH+QTOTAL(I)
3       410  CONTINUE
2            WRITE(*,*)' '
2            QTOFCZ=QTOFCZ+QTOTFC
2            QTOTFH=-QTOTFH
2            QTOFHZ=QTOFHZ+QTOTFH
2       C
2            CALL TWRITE(NPRT,IOT,NZ,ATDB,SOLH,QTOTAL,ZTROOM,VWIND,QTSHL,PHI1,
2           +PHI2)
2       C
2            WRITE(*,*)' '
2            SERMAX=SERMAX+ZERMAX(NZ)
2            IF(NHTX.EQ.1) CALL HEATX
2            INITZ(NZ)=0
2       475  CONTINUE
1            WRITE(*,*)' '
1            CALL ENERGY(ATDB,QTOTAL,ERB,INIHX,IDAYY,MEND,DAYWK)
1            INIHX=0
1       480  CONTINUE
        C
             WRITE(*,*)' '
             CALL INPUTI(NSTOP,'INPUT 1 FOR ANOTHER RUN FOR THIS BUILDING $')
        C
             IF(NSTOP.EQ.0)GO TO 490
             GO TO 61
        490  CONTINUE
        C
             STOP
        C
        510  FORMAT (1H )
        520  FORMAT (1H1)
        C
             END
```

10-29-87
11:19:27

D Line# 1 7 Microsoft FORTRAN77 V3.13 8/05/83

```
C THIS FILE CONTAINS INITIALIZATION OF VARIABLES AND WRITE STATEMENTS
C
      SUBROUTINE INTIAL(DAYWK,HOLDAY,NZON,NWAR)
      COMMON /BLCK23/ BINF(3),CINF(3),SGV1(3),CWRV1(3),HGLV1(3)
      COMMON /BLCK24/ RTFW1(12),CRATFG(12),CRATFP(12)
      COMMON /BLCK25/ ACOEF(4),ALPAJ(8),TAUJ(8)
      COMMON /BLCK26/ NSTOP,NHTX,IENERG,NWKND,ML,MF,MA,PHI1,PHI2
      COMMON /BLCK27/ IFLAGD,XLAT,ACLONG,STLONG,PB,NHLDAY
      COMMON /BLOCK1/  IIN,IOT,MC,NPRT,INWRIT,ZTROOM(4),ZOFST(4),INM2,
     +ZOFCT(4),ZCFMD(4),XKTZ(4),XLF(4),INITZ(4),INIHX,MONTH,IDAYM,DAYTYP
      COMMON /BLOCK3/  MONTH1,MONTH2,MDAY1,MDAY2
C
      INTEGER DAYWK,HOLDAY(50),DAYTYP
      DIMENSION NWAR(4)
C
      SGV1(1)=0.224
      SGV1(2)=0.197
      SGV1(3)=0.187
C
      BINF(1)=0.002
      BINF(2)=0.004
      BINF(3)=0.006
C
      CINF(1)=0.003
      CINF(2)=0.0096
      CINF(3)=0.0233
C
      CWRV1(1)=0.703
      CWRV1(2)=0.681
      CWRV1(3)=0.676
C
      HGLV1(1)=0.0
      HGLV1(2)=0.0
      HGLV1(3)=0.0
C
      RTFW1(1)=-0.88
      RTFW1(2)=-0.92
      RTFW1(3)=-0.95
      RTFW1(4)=-0.97
      RTFW1(5)=-0.84
      RTFW1(6)=-0.90
      RTFW1(7)=-0.94
      RTFW1(8)=-0.96
      RTFW1(9)=-0.81
      RTFW1(10)=-0.88
      RTFW1(11)=-0.93
      RTFW1(12)=-0.95
C
      ACOEF(1)=0.45
      ACOEF(2)=0.55
```

```
      ACOEF(3)=0.65
      ACOEF(4)=0.75
C
C     CRATFG(1)=1.73
C     CRATFG(2)=-3.50
C     CRATFG(3)=2.22
C     CRATFG(4)=-0.45
C     CRATFG(5)=1.88
C     CRATFG(6)=-4.22
C     CRATFG(7)=3.08
C     CRATFG(8)=-0.74
C     CRATFG(9)=1.89
C     CRATFG(10)=-4.55
C     CRATFG(11)=3.61
C     CRATFG(12)=-0.95
C
C     CRATFP(1)=1.000
C     CRATFP(2)=-1.8260
C     CRATFP(3)=1.0697
C     CRATFP(4)=-0.2005
C     CRATFP(5)=1.000
C     CRATFP(6)=-2.1092
C     CRATFP(7)=1.4606
C     CRATFP(8)=-0.3331
C     CRATFP(9)=1.000
C     CRATFP(10)=-2.2908
C     CRATFP(11)=1.7252
C     CRATFP(12)=-0.4277
C
C     COEFFICIENTS FOR REGULAR DS SHEET GLASS
C
      ALPAJ(1)=0.01154
      ALPAJ(2)=0.77674
      ALPAJ(3)=-3.94657
      ALPAJ(4)=8.57881
      ALPAJ(5)=-8.38135
      ALPAJ(6)=3.01188
      ALPAJ(7)=0.0
      ALPAJ(8)=0.0
C
      TAUJ(1)=-0.00885
      TAUJ(2)=2.71235
      TAUJ(3)=-0.62062
      TAUJ(4)=-7.07329
      TAUJ(5)=9.75995
      TAUJ(6)=-3.89922
      TAUJ(7)=0.0
      TAUJ(8)=0.0
C
C
C
      NSTOP=0
```

```
      NHTX=1
      IENERG=1
    4 NWKND=1
      NHLDAY=0
      DAYWK=5
      DO 5 I=1,50
    5 HOLDAY(I)=0
C READ UNIT NUMBER
      IIN=5
C WRITE UNIT NUMBER
      IOT=6
      DO 1 NZ=1,4
      NWAR(NZ)=1
      XLF(NZ)=1000.0
    1 INITZ(NZ)=1
      NPRT=1
      MC=2
      ML=2
      MF=3
      MA=2
      PHI1=0.2
      PHI2=0.3
      MONTH1=1
      MONTH2=1
      MDAY1=1
      MDAY2=7
      IFLAGD=0
      INWRIT=1
      XLAT=35.0
      ACLONG=97.0
      STLONG=90.0
      PB=14.7
      INM2=0
      NZON=1
C
      RETURN
      END
C
C  END OF SUBROUTINE INTIAL
C
C  START OF SUBROUTINE TWINT
C
      SUBROUTINE TWINT
      COMMON /BLCK35/ WRL,WRW,AD,AW,PSI,EPSILN,HO
      COMMON /BLCK36/ HI,ROG,ALPHWR,EPSWR,SCG,UWRA,UW,UD,BT(7),DT(7)
C
      WRL=0.0
      WRW=0.0
      AD=0.0
      AW=0.0
      PSI=0.0
      EPSILN=90.0
```

```
      HO=3.0
      HI=1.46
      ROG=0.2
      ALPHWR=0.8
      EPSWR=0.9
      SCG=0.0
      UWRA=0.0
      UW=0.0
      UD=0.0
      DO 20 I=1,7
      BT(I)=0.0
      DT(I)=0.0
   20 CONTINUE
C
      RETURN
      END
C
C  END OF SUBROUTINE TWINT
C
C  START OF SUBROUTINE THRINT
C
      SUBROUTINE THRINT
      COMMON /BLCK37/ NPN,NPD,IBLD,CLHGT,QOTSN,QOTSD,QOTLN,QOTLD,QFLN
      COMMON /BLCK38/ QFLD,QTLN,QTLD,OFST,OFCT,TROOM,ERMIN,ERMAX
      COMMON /BLCK39/ FLAREA,THRANG,THSETD,THSETN,THTIMN,THTIMD
C
      NPN=0
      NPD=0
      IBLD=2
      CLHGT=10.0
      QOTSN=0.0
      QOTSD=0.0
      QOTLN=0.0
      QOTLD=0.0
      QFLN=0.0
      QFLD=0.0
      QTLN=0.0
      QTLD=0.0
      OFST=8.0
      OFCT=17.0
      TROOM=70.0
      ERMIN=0.0
      ERMAX=0.0
      FLAREA=0.0
      THRANG=3.0
      THSETD=78.
      THSETN=78.
      THTIMN=17.
      THTIMD=8.
C
      RETURN
      END
C
```

```
C  END OF SUBROUTINE THRINT
C
C  START OF SUBROUTINE FORINT
C
      SUBROUTINE FORINT(NZ,BINF,IBLDZ,VWIND,CINF,DATR,QTOTAL)
      COMMON /BLCK40/ ACH(4,24),QTSHZ(24),QTSHL(24)
      COMMON /BLCK41/ QTOTAZ(24),QTOFHZ,QTOFCZ,Q30Z(24)
      COMMON /BLCK42/ QSFWZ(24),Q37Z(24),Q38Z(24),Q39Z(24),Q31Z(24)
      COMMON /BLCK43/ QOTHSZ(24),QOTHLZ(24),QIVSZ(24),QIVLZ(24)
      COMMON /BLCK44/ QPPLZ(24),QPPSZ(24),QFLSZ(24),QTLSZ(24)
      COMMON /BLCK45/ QSSWRZ(24),Q35Z(24)
C
      DIMENSION BINF(3),IBLDZ(4),VWIND(24),CINF(3),DATR(4,24),QTOTAL(24)
C
      DO 112 I=1,24
      ACH(NZ,I)=0.25+BINF(IBLDZ(NZ))*VWIND(I)+CINF(IBLDZ(NZ))*ABS
     +(DATR(NZ,I))
      QTSHZ(I)=0.0
      QTSHL(I)=0.0
      QTOTAL(I)=0.0
      QTOTAZ(I)=0.
      QTOFHZ=0.
      QTOFCZ=0.
      Q30Z(I)=0.
      QOTHSZ(I)=0.
      QSFWZ(I)=0.
      Q37Z(I)=0.
      Q38Z(I)=0.
      Q39Z(I)=0.
      Q31Z(I)=0.
      QOTHSZ(I)=0.
      QOTHLZ(I)=0.
      QIVSZ(I)=0.
      QIVLZ(I)=0.
      QPPLZ(I)=0.
      QPPSZ(I)=0.
      QFLSZ(I)=0.
      QTLSZ(I)=0.
      QSSWRZ(I)=0.
      Q35Z(I)=0.
  112 CONTINUE
C
      RETURN
      END
C
C  END OF SUBROUTINE FORINT
C
C  START OF SUBROUTINE WRITE1
C
      SUBROUTINE WRITE1(NZ,NS,NPRT,IOT,IH,XIT)
      COMMON /BLCK46/ ZALP(4,10),ZORNT(4,10),TSWR(4,10,48)
      COMMON /BLCK47/ ZAW(4,10),SHGF(24),TSW(24),ZAWR(4,10)
      COMMON /BLCK48/ QEWR(4,10,48),QEW(24),ZAD(4,10),QED(24)
```

```
C
      CHARACTER*80 ZORNT
      DIMENSION XIT(24)
C
      IF (NPRT.LT.4) GO TO 270
      WRITE (IOT,510)
      IF (ZALP(NZ,NS).LE.0.0) GO TO 240
      WRITE (IOT,560) ZORNT(NZ,NS)
      WRITE (IOT,550) ((I,XIT(I),I=N,24,3),N=1,3)
      WRITE (IOT,500)
  240 WRITE (IOT,570) ZORNT(NZ,NS)
      WRITE (IOT,550) ((I,TSWR(NZ,NS,I),I=N,24,3),N=1,3)
      IF (ZAW(NZ,NS).LE.0.0) GO TO 250
      WRITE (IOT,510)
      WRITE (IOT,580) ZORNT(NZ,NS)
      WRITE (IOT,550) ((I,SHGF(I),I=N,24,3),N=1,3)
      WRITE (IOT,510)
      WRITE (IOT,590) ZORNT(NZ,NS)
      WRITE (IOT,550) ((I,TSW(I),I=N,24,3),N=1,3)
  250 WRITE (IOT,510)
      WRITE (IOT,600) ZORNT(NZ,NS),ZAWR(NZ,NS)
      WRITE (IOT,610) ((I,QEWR(NZ,NS,I),I=N,24,3),N=1,3)
      IF (ZAW(NZ,NS).LE.0.0) GO TO 260
      WRITE (IOT,510)
      WRITE (IOT,620) ZORNT(NZ,NS),ZAW(NZ,NS)
      WRITE (IOT,610) ((I,QEW(I),I=N,24,3),N=1,3)
  260 IF (ZAD(NZ,NS).LE.0.0) GO TO 270
      WRITE (IOT,510)
      WRITE (IOT,630) ZORNT(NZ,NS),ZAD(NZ,NS)
      WRITE (IOT,610) ((I,QED(I),I=N,24,3),N=1,3)
  270 DO 280 I=1,24
1           IH=I+24
1           QEWR(NZ,NS,I)=QEWR(NZ,NS,IH)
1     280 CONTINUE
C
      RETURN
C
  500 FORMAT (1H )
  510 FORMAT (1H )
  550 FORMAT (3(2X,8(5X,I2,2X,F5.1),/))
  560 FORMAT (5X,32HSOLAR INCIDENT ENERGY ON SURFACE,2X,A8,2X,15HIN BTU/
     1HR-FT**2)
  570 FORMAT (5X,31HSOL-AIR TEMPERATURE FOR SURFACE,2X,A8,2X,4HIN F)
  580 FORMAT (5X,44HSOLAR HEAT GAIN FACTOR FOR WINDOW ON SURFACE,2X,A8,2
     1X,15HIN BTU/HR-FT**2)
  590 FORMAT (5X,41HSOL-AIR TEMPERATURE FOR WINDOW ON SURFACE,2X,A8,2X,4
     1HIN F)
  600 FORMAT (5X,25HHEAT GAIN THROUGH SURFACE,2X,A8,21H IN BTU/HR      AR
     1EA= ,F8.0,1X,6HSQ.FT.)
  610 FORMAT (3(4X,8(2X,I2,2X,F8.0),/))
  620 FORMAT (5X,35HHEAT GAIN THROUGH WINDOW ON SURFACE,2X,A8,21H IN BTU
     1/HR     AREA= ,F6.1,7H SQ.FT.)
```

```
630   FORMAT (5X,35HHEAT GAIN THROUGH  DOOR  ON SURFACE,2X,A8,5X,5HAREA=
     1,F6.1,5HFT**2,2X,9HIN BTU/HR)
C
      END
C
C  END OF SUBROUTINE WRITE1
C
C  START OF SUBROUTINE TWRITE
C
      SUBROUTINE TWRITE(NPRT,IOT,NZ,ATDB,SOLH,QTOTAL,ZTROOM,VWIND,
     +QTSHL,PHI1,PHI2)
      COMMON /BLCK49/ QFLS(48),QTLS(24),QPPS(24),QOTHS(24),QIVS(24)
      COMMON /BLCK50/ QSSWR(4,48),Q35(24),QPPL(24),QOTHL(24),QIVL(24)
      COMMON /BLCK51/ Q30(24),Q31(24),Q37(4,48),Q38(4,48),Q39(4,48)
      COMMON /BLCK52/ QSFW(4,48),ATWB(24)
      COMMON /BLCK53/ DW(4,24)
      COMMON /BLCK54/ QTOTFC,QTOTFH
C
      DIMENSION ATDB(24),SOLH(24),QTOTAL(24),ZTROOM(24),VWIND(24),
     +QTSHL(24)
C
      IF (NPRT.LE.1) GO TO 470
      IF (NPRT.LE.2) GO TO 450
C
C     WRITE INSTANTANEOUS SENSIBLE HEAT GAINS
C
      WRITE (IOT,520)
      WRITE (IOT,640)
      WRITE (IOT,540)
      WRITE (IOT,650)
      WRITE (IOT,510)
      DO 420 I=1,24
      WRITE (IOT,660) I,QFLS(I),QTLS(I),QPPS(I),QOTHS(I),QIVS(I),QSSWR(N
     1 Z,I),Q35(I)
      WRITE (IOT,500)
420   CONTINUE
C
C     WRITE LATENT HEAT GAINS
C
      WRITE (IOT,520)
      WRITE (IOT,670)
      WRITE (IOT,540)
      WRITE (IOT,680)
      WRITE (IOT,510)
      DO 430 I=1,24
      WRITE (IOT,690) I,QPPL(I),QOTHL(I),QIVL(I),Q30(I)
      WRITE (IOT,500)
430   CONTINUE
C
C     WRITE SENSIBLE COOLING LOAD DUE TO VARIOUS HEAT GAINS
C
      WRITE (IOT,520)
      WRITE(IOT,530)NZ
```

```
      WRITE (IOT,700)
      WRITE (IOT,540)
      WRITE (IOT,710)
      WRITE (IOT,510)
      DO 440 I=1,24
      WRITE (IOT,720) I,Q31(I),Q37(NZ,I),Q38(NZ,I),Q39(NZ,I),QSFW(NZ,I),
     +QTSHL(I)
      WRITE (IOT,500)
  440 CONTINUE
  450 CONTINUE
      WRITE (IOT,520)
      WRITE(IOT,530)NZ
      WRITE (IOT,730)
      WRITE (IOT,540)
      WRITE (IOT,740)
      WRITE (IOT,510)
      DO 460 I=1,24
      WRITE (IOT,750) I,ATDB(I),ATWB(I),SOLH(I),VWIND(I),DW(NZ,I),
     1QTSHL(I),Q30(I),QTOTAL(I)
      WRITE (IOT,500)
  460 CONTINUE
      WRITE (IOT,510)
      WRITE (IOT,760) QTOTFC,QTOTFH,ZTROOM(NZ),PHI1,PHI2
      WRITE (IOT,520)
  470 CONTINUE
C
      RETURN
C
  500 FORMAT (1H )
  510 FORMAT (1H )
  520 FORMAT (1H1)
  530 FORMAT (56X,26HLOAD CALCULATIONS FOR ZONE,2X,I2,/)
  540 FORMAT (52X,25(1H*),/)
  640 FORMAT (52X,32HINST SENSIBLE HEAT GAIN (BTU/HR))
  650 FORMAT (10X,4HTIME,3X,13HON-OFF LIGHTS,4X,9HON LIGHTS,6X,6HPEOPLE,
     19X,9HEQUIPMENT,6X,9HINFL&VENT,7X,8HSURFACES,13X,5HTOTAL)
  660 FORMAT (10X,I3,1X,6E15.4,E18.4)
  670 FORMAT (52X,26HLATENT HEAT LOADS (BTU/HR))
  680 FORMAT (10X,4HTIME,5X,6HPEOPLE,9X,9HEQUIPMENT,6X,9HINFL&VENT,10X,5
     1HTOTAL)
  690 FORMAT (10X,I3,1X,3E15.4,E18.4)
  700 FORMAT (35X,67HSENSIBLE COOLING LOAD COMPONENTS DUE TO VARIOUS HEA
     1T GAINS (BTU/HR))
  710 FORMAT (10X,4HTIME,5X,7HINSTANT,5X,13HLIGHTS ON OFF,5X,8HSURFACES,
     15X,12HPEOPLE&EQUIP,5X,12HWINDOWS UNSH,7X,5HTOTAL)
  720 FORMAT (10X,I4,5E15.4,E18.4)
  730 FORMAT (52X,27HTOTAL COOLING LOAD (BTU/HR))
  740 FORMAT(5X,'TIME',3X,'DB-TEMP',3X,'WB-TEMP',3X,'SOL-FLUX',3X,
     +'VWIND',7X,'HUM-DIF',11X,'SENSIBLE',13X,'LATENT',15X,'  TOTAL ')
  750 FORMAT(5X,I3,4(5X,F5.1),2X,F13.6,3E20.5)
  760 FORMAT (10X,32HTHE TOTAL LOAD FOR THE WHOLE DAY,//,10X,8HCOOLING=
     1,G12.5,2X,8HHEATING=,G12.5,5X,10HROOM TEMP=,F5.1,5X,5HR.H1=,F5.3,2
     2X,5HR.H2=,F5.3)
      END
```

```
                                                         10-29-87
                                                         11:20:40
D Line# 1      7                  Microsoft FORTRAN77 V3.13 8/05/83
  C
  C  THIS FILE CONTAINS THE SUBROUTINES SOLAR, HEATX, WETHER, HDAY, AND JU
  C
        SUBROUTINE SOLAR (IDAY)
        COMMON /SOL1/ ATDB(24),SOLH(24),XID(24),XIDHV(24),XIT(24)
        COMMON /SOL2/ XLATR,EPR,PSIR,DLONG,IDAY1
        COMMON /SOL3/ SH(24),CH(24),CZ(24),CT(24),CE,SD,CD,SL,CL
        DIMENSION SID(24), SIDHV(24)
        SC=428.0
        IF (IDAY.EQ.IDAY1) GO TO 20
        DAY=IDAY
        EQTIME=-0.092075+1.367238*COS(0.017214*DAY)-7.044381*SIN(0.017214*
       1DAY)-1.287451*COS(0.034428*DAY)-10.175641*SIN(0.034428*DAY)-0.1112
       263*COS(0.051643*DAY)-1.084028*SIN(0.051643*DAY)-0.144821*COS(0.068
       3857*DAY)-0.616121*SIN(0.068857*DAY)+0.061813*COS(0.086071*DAY)-0.4
       428534*SIN(0.086071*DAY)-0.094842*COS(0.103285*DAY)-0.256195*SIN(0.
       5103285*DAY)
        STC=(EQTIME/60.0)+DLONG
        D=0.40928*SIN(6.28319*(284.0+IDAY)/365.0)
        SD=SIN(D)
        CD=COS(D)
        SL=SIN(XLATR)
        CL=COS(XLATR)
        CZT=SD*SL
        CDL=CD*CL
        DO 10 I=1,24
1       STLTIM=STC+I
1       H=3.1416-(0.2618*STLTIM)
1       SH(I)=SIN(H)
1       CH(I)=COS(H)
1       CZ(I)=CZT+(CDL*CH(I))
1       IF (CZ(I).LT.0.05) CZ(I)=0.05
1       XKX=SOLH(I)/(SC*CZ(I))
1       IF (XKX.GT.0.75) XKX=0.75
1       RATIO=0.5*(1.0+COS(XKX*3.14159))
1       SIDHV(I)=RATIO*SOLH(I)
1       SID(I)=SOLH(I)-SIDHV(I)
1    10 CONTINUE
     20 SE=SIN(EPR)
        CE=COS(EPR)
        SP=SIN(PSIR)
        CP=COS(PSIR)
        CTT=SD*(SL*CE-CL*SE*CP)
        DO 30 I=1,24
1       CT(I)=0.0
1       XIT(I)=0.0
1       XID(I)=0.0
1       XIDHV(I)=0.0
1       IF (SOLH(I).LE.0.0) GO TO 30
```

```
      CT(I)=CTT+(CH(I)*CD)*((CL*CE)+(SL*SE*CP))+(CD*SE*SP*SH(I))
      IF (CT(I).LT.0.0) CT(I)=0.0
      RB=CT(I)/CZ(I)
      IF (RB.GT.5.0) RB=1.0
      XIDHV(I)=SIDHV(I)*0.5*(1.0+CE)
      XID(I)=SID(I)*RB
      XIT(I)=XID(I)+XIDHV(I)
   30 CONTINUE
      IDAY1=IDAY
      RETURN
      END
C
      SUBROUTINE HEATX
      COMMON /BLOCK1/ IIN,IOT,MC,NPRT,INWRIT,ZTROOM(4),ZOFST(4),INM2,
     1ZOFCT(4),ZCFMD(4),XKTZ(4),XLF(4),INITZ(4),INIHX,MONTH,IDAYM,
     $DAYTYP
      COMMON /BLOCK2/ QTOTAL(24),ER(4,48)
C
      COMMON/BLOCK4/ZERMAX(4),ZERMIN(4),ZFLARA(4),ZTRANG(4),ZTSETD(4),
     $ZTSETN(4),ZTTIMN(4),ZTTIMD(4)
      COMMON /BLOCK6/ NZ,ERB(24)
      INTEGER DAYTYP
      DIMENSION G(4), P(4), ZG(9), ZP(9)
      DIMENSION XI(24), QTOT(4,48), T(4,48)
      DATA ZG/1.68,-1.73,0.05,1.81,-1.89,0.08,1.85,-1.95,0.10/
      DATA ZP/1.0,-0.82,0.0,1.0,-0.87,0.0,1.0,-0.93,0.0/
C
      IFLAGH=0
      I=3*MC-2
      IP3=I+2
      IJ=1
      DO 10 J=I,IP3
      P(IJ)=ZP(J)
      G(IJ)=ZG(J)
   10 IJ=IJ+1
      IF(INITZ(NZ).EQ.0) GO TO 30
      SUMP=P(1)+P(2)+P(3)
      DO 20 I=1,24
      QTOT(NZ,I)=QTOTAL(I)
   20 ER(NZ,I)=QTOTAL(I)
   30 IF(ZERMAX(NZ).LT.0.0) IFLAGH=1
      ID=ZTTIMD(NZ)
      IN=ZTTIMN(NZ)
      ED=ZERMAX(NZ)-ZERMIN(NZ)
      I=3*MC-2
      G(1)=(G(1)*ZFLARA(NZ))+(((XKTZ(NZ)*XLF(NZ))+(ZCFMD(NZ)*1.08))*P(1
     +))
      G(2)=(G(2)*ZFLARA(NZ))+(((XKTZ(NZ)*XLF(NZ))+(ZCFMD(NZ)*1.08))*P(2
     +))
      G(3)=G(3)*ZFLARA(NZ)
      SUMG=G(1)+G(2)+G(3)
      S=ABS(ED/ZTRANG(NZ))
```

```
          THDD=ZTSETD(NZ)
          WN=(ED/2.0)-S*ZTSETN(NZ)
          GT1=G(1)/(S+G(1))
          ST1=S/(S+G(1))
          IF (INITZ(NZ).EQ.0) GO TO 50
          DO 40 I=1,24
1         T(NZ,I)=ZTSETN(NZ)
1         IF (I.GE.ID.AND.I.LT.IN.AND.DAYTYP.EQ.1) T(NZ,I)=ZTSETD(NZ)
1    40   T(NZ,I+24)=T(NZ,I)
     50   WD=(ED/2.0)-S*THDD
          IF (DAYTYP.EQ.2) WD=WN
          DO 60 I=1,24
1    60   QTOT(NZ,I+24)=QTOTAL(I)
          KOUNT=1
          II=25
          IL=ID-1+24
     70   WT=WN
     80   DO 120 K=II,IL
1         XIT=0.0
1         DO 90 J=2,3
2         JJ=K+1-J
2         XIT=XIT-G(J)*T(NZ,JJ)+P(J)*QTOT(NZ,JJ)-P(J)*ER(NZ,JJ)
2    90   CONTINUE
1         KJ=K-24
1         XI(KJ)=XIT+ZTROOM(NZ)*SUMG+P(1)*QTOT(NZ,K)
1         ER(NZ,K)=(GT1*WT)+(ST1*XI(KJ))
1         IF (IFLAGH.EQ.1) GO TO 100
1         IF (ER(NZ,K).LT.ZERMIN(NZ)) ER(NZ,K)=ZERMIN(NZ)
1         IF (ER(NZ,K).GT.ZERMAX(NZ)) ER(NZ,K)=ZERMAX(NZ)
1         GO TO 110
1    100  IF (ER(NZ,K).GT.ZERMIN(NZ)) ER(NZ,K)=ZERMIN(NZ)
1         IF (ER(NZ,K).LT.ZERMAX(NZ)) ER(NZ,K)=ZERMAX(NZ)
1    110  T(NZ,K)=(XI(KJ)-ER(NZ,K))/G(1)
1    120  CONTINUE
          KOUNT=KOUNT+1
          II=IL+1
          IL=IN+24
          WT=WD
          IF (KOUNT.EQ.2) GO TO 80
          IL=48
          IF (KOUNT.EQ.3) GO TO 70
          ERTOTC=0.0
          ERTOTH=0.0
          DO 130 I=1,24
1         IP24=I+24
1         QTOT(NZ,I)=QTOT(NZ,IP24)
1         ER(NZ,I)=ER(NZ,IP24)
1         ERB(I)=ERB(I)+ER(NZ,I)
1         T(NZ,I)=T(NZ,IP24)
1    130  CONTINUE
          IF(IFLAGH.EQ.0) GO TO 171
          DO 170 I=1,24
```

```
1        XI(I)=-ER(NZ,I)
1        ERTOTH=ERTOTH+XI(I)
1    170 CONTINUE
     171 CONTINUE
         IF (NPRT.EQ.0) GO TO 190
         IF (IFLAGH.EQ.1) GO TO 160
     C    WRITE (IOT,215)
     C   WRITE (IOT,210)
     C    WRITE (IOT,230)
     C    WRITE (IOT,220)
     C    WRITE (IOT,340) MONTH,IDAYM
     C    WRITE (IOT,260) ZTSETD(NZ),ZTTIMD(NZ),ZTSETN(NZ),ZTTIMN(NZ)
     C   WRITE (IOT,200)
         DO 142 N=1,3
1    C    WRITE (IOT,250) (I,T(NZ,I),I=N,24,3)
1     142 CONTINUE
     C   WRITE (IOT,210)
     C    WRITE (IOT,270) ZERMIN(NZ),ZERMAX(NZ)
     C   WRITE (IOT,200)
         DO 144 N=1,3
1    C    WRITE (IOT,290) (I,ER(NZ,I),I=N,24,3)
1     144 CONTINUE
     C   WRITE (IOT,210)
         GO TO 150
     160 ERMAXH=-ZERMAX(NZ)
         ERMINH=ABS(ZERMIN(NZ))
     C    WRITE (IOT,215)
     C   WRITE (IOT,210)
     C    WRITE (IOT,240)
     C    WRITE (IOT,220)
     C    WRITE (IOT,340) MONTH,IDAYM
     C    WRITE (IOT,260) ZTSETD(NZ),ZTTIMD(NZ),ZTSETN(NZ),ZTTIMN(NZ)
     C   WRITE (IOT,200)
         DO 175 N=1,3
1    C    WRITE (IOT,250) (I,T(NZ,I),I=N,24,3)
1     175 CONTINUE
     C   WRITE (IOT,210)
     C    WRITE (IOT,280) ERMINH,ERMAXH
     C   WRITE (IOT,200)
         DO 177 N=1,3
1    C    WRITE (IOT,290) (I,XI(I),I=N,24,3)
1     177 CONTINUE
     C   WRITE (IOT,210)
         GO TO 180
     190 CONTINUE
     C    WRITE (IOT,340) MONTH,IDAYM
         IF (IFLAGH.EQ.1) GO TO 180
      150 DO 140 I=1,24
1    140 ERTOTC=ERTOTC+ER(NZ,I)
     C    WRITE (IOT,300) ERTOTC
     C   WRITE (IOT,210)
     C   WRITE (IOT,210)
         GO TO 195
```

```
180   WRITE (IOT,310) ERTOTH
C     WRITE (IOT,210)
C     WRITE (IOT,210)
195   RETURN
C
200   FORMAT (/)
210   FORMAT (/)
215   FORMAT (1H1)
220   FORMAT (51X,31(1H*),//)
230   FORMAT (52X,30HHEAT EXTRACTION RATES (BTU/HR))
240   FORMAT (52X,30HHEAT  ADDITION  RATES (BTU/HR))
250   FORMAT (3(2X,8(5X,I2,2X,F5.1),/))
260   FORMAT('     ROOM AIR TEMPERATURES 1-24 HRS     THERMOSTAT SETTING
     1',F5.1,1X,1HF,1X,2HAT,F4.0,3HHRS,3X,F5.1,1X,1HF,1X,2HAT,F4.0,
     23HHRS)
270   FORMAT ('     HEAT EXTRACTION RATES 1-24 HRS     ERMIN=',
     1 G13.6,'  ERMAX=',G13.6,'  BTU/HR')
280   FORMAT ('     HEAT  ADDITION  RATES 1-24 HRS     ERMIN=',
     1 G13.6,'  ERMAX=',G13.6,'  BTU/HR')
290   FORMAT (3(4X,8(2X,I2,1X,F9.0),/))
300   FORMAT (5X,47HTOTAL COOLING LOAD PROVIDED DURING THE 24 HRS =,E14.
     16,1X,7HBTU/DAY)
310   FORMAT (5X,47HTOTAL HEATING LOAD PROVIDED DURING THE 24 HRS =,E14.
     16,1X,7HBTU/DAY)
320   FORMAT(5X,53HTOTAL COOLING LOAD FROM BEGINNING (OF MONTH) TO TODAY
     1,1H=,E14.6,1X,3HBTU)
330   FORMAT(5X,53HTOTAL HEATING LOAD FROM BEGINNING (OF MONTH) TO TODAY
     1,1H=,E14.6,1X,3HBTU)
340   FORMAT (5X,8HMONTH = ,I2,5X,6HDAY = ,I3,/)
      END
      FUNCTION PSL (T)
      TK=(T-32.0)/1.8+273.16
      X=647.27-TK
      Y=X*(3.2438+(5.8683E-03+1.17024E-08*X*X)*X)/(TK*(1.0+2.18785E-03*X
     1))
      PSL=14.696*218.167/(10.0**Y)
      RETURN
      END
C
      SUBROUTINE WETHER (IDAY,ATDB,ATWB,SOLH,VWIND,FNAME)
C
      DIMENSION ATDB(24),ATWB(24),SOLH(24),VWIND(24)
      DIMENSION JV(24),JFLUX(24)
      DIMENSION CTDB(24),CTWB(24),CV(24)
      INTEGER*2 JV,JFLUX
      CHARACTER*1 CTDB,CTWB,CV
      CHARACTER*80 FNAME
C
      OPEN(1,FILE=FNAME,STATUS='OLD',ACCESS='DIRECT',
     +FORM='BINARY',RECL=120)
      READ(1,REC=IDAY) (CTDB(J),CTWB(J),CV(J),JFLUX(J),J=1,24)
      CLOSE(1)
```

```
         DO 1600 M=1,24
         ATDB(M)=FLOAT(ICHAR(CTDB(M))-128)
         ATWB(M)=FLOAT(ICHAR(CTWB(M))-128)
         JV(M)=ICHAR(CV(M))
         SOLH(M)=0.36855*FLOAT(JFLUX(M))
         VWIND(M)=1.151*FLOAT(JV(M))
         IF(JFLUX(M).EQ.9999)SOLH(M)=0.0
         IF(JV(M).EQ.999)VWIND(M)=0.0
C     WRITE(*,*) ATDB(M),ATWB(M),VWIND(M),SOLH(M)
 1600 CONTINUE
      RETURN
      END
C
C
C
      SUBROUTINE HDAY(MONTH,IDAYM,HOLDAY,NHDAY,DAYTYP)
      DIMENSION HOLDAY(NHDAY)
      INTEGER DAY,DAYTYP,HOLDAY
      DAY=100*MONTH+IDAYM
      DO 20 J=1,NHDAY
      IF(HOLDAY(J).LT.0) GO TO 20
      IF(HOLDAY(J).GT.DAY) GO TO 30
      IF(J.LT.NHDAY.AND.HOLDAY(J+1).LT.0) GO TO 10
      IF(DAY.NE.HOLDAY(J)) GO TO 20
      DAYTYP=2
      GO TO 30
   10 IF(DAY.LT.HOLDAY(J)) GO TO 20
      IF(DAY.GT.IABS(HOLDAY(J+1))) GO TO 20
      DAYTYP=2
      GO TO 30
   20 CONTINUE
   30 RETURN
      END
C
      SUBROUTINE JULIAN (MONTH,DAY,JDAY)
C
C     THIS ROUTINE CAN CONVERT MONTH AND DAY TO THE JULIAN
C     DATE JDAY OR IT CAN CONVERT JDAY TO MONTH AND DAY.
C     USE CALL JULIAN TO CONVERT TO JULIAN AND USE CALL MONDAY
C     TO CONVERT TO MONTH AND DAY.
C
      INTEGER NDAYM(13),DAY
C
      DATA NDAYM/1,32,60,91,121,152,182,213,244,274,305,335,366/
C
      JDAY=DAY+NDAYM(MONTH)-1
      RETURN
      END
C
      SUBROUTINE MONDAY(MONTH,DAY,JDAY)
      INTEGER NDAYM(13),DAY
      DATA NDAYM/1,32,60,91,121,152,182,213,244,274,305,335,366/
```

```
        IDAY=JDAY
        IF (IDAY.GT.365) IDAY=IDAY-365
        DO 10 I=2,13
1       IF (IDAY.LT.NDAYM(I)) GO TO 20
1    10 CONTINUE
      C
     20 MONTH=I-1
        DAY=IDAY-NDAYM(MONTH)+1
      C
        RETURN
        END
```

```
  C
        SUBROUTINE ENERGY(ATDB,QTOTAL,ENER,INM,IDAYY,MEND,DAYWK)
        CHARACTER*8 MNAME
        COMMON /BLOCK1/IIN,IOT,MC,NPRT,INWRIT,ZTROOM(4),ZOFST(4),INM2,
       1ZOFCT(4),ZCFMD(4),XKTZ(4),XLF(4),INITZ(4),INIHX,MONTH,IDAYM,
       &DAYTYP
        COMMON /BLOCK3/ MONTH1,MONTH2,MDAY1,MDAY2
        INTEGER DAYTYP
  C RANDOM NUMBER GENERATOR
        INTEGER X,Y,Z
        REAL RANDOM,TEMP,ZLEFT,PLEFT,CHILL,PCHILL,TOTCL
        DIMENSION ATDB(24),ENER(24),QTOTAL(24)
        DIMENSION PC(24),PSYS(24),PFAN(24),PE(24),PL(24),ZENER(24)
  C
  C       PRV=POWER ROOF VENTILATORS IN HORSEPOWER INPUT,
  C       FANP= VENT AIR FAN HP, EQPD=EQ AND APL KW INPUT - DAY,
  C       EQPN=EQ AND APL KW INPUT - NIGHT, QFLD=TOT. LIGHTS KW - DAY,
  C       QFLN=TOT. LIGHTS KW - NIGHT
  C
        DATA PRV,FANP,EQPD,EQPN/0.0,57.6,59.6,32.5/
        DATA QFLD,QFLN/175.1,17.53/
  C
        WRITE(6,9999) DAYTYP
  9999  FORMAT(' ***DAYTYP=',I2)
        WRITE(IOT,899)
  899   FORMAT(14X,'       TOTAL COOLING LOAD FOR 24 HOURS (BTU/HR)')
        WRITE(IOT,900) (I,ENER(I),I=1,24,4)
        WRITE(IOT,900) (I,ENER(I),I=2,24,4)
        WRITE(IOT,900) (I,ENER(I),I=3,24,4)
        WRITE(IOT,900) (I,ENER(I),I=4,24,4)
  900   FORMAT(6(2X,I2,1X,F8.0),/,1X)
        WRITE(*,*)' '
  C
        TOTCL=0.0
  C
  C DO LOOP FOR THE FOUR CASES TO BE ANALYZED
  C MM=1  THERMAL
  C MM=2  CONVENTIONAL
  C MM=3  THERMAL (RANDOM)
  C MM=4  CONVENTIONAL (RANDOM)
        DO 333 MM=1,4
1 C RANDOM NUMBER GENERATOR (FOR -20% TO 20%)
1       IF(MM.NE.3) GOTO 334
1       X=IDAYY
1       Y=COST
1       Z=ELD/20.0
1       X=171*MOD(X,177)-2*((X/177)-MOD(X,177))
1       IF(X.LT.0) X=X+30296
1       Y=172*MOD(Y,176)-35*((Y/176)-MOD(Y,176))
1       IF(Y.LT.0) Y=Y+30307
```

```
1          Z=170*MOD(Z,178)-63*((Z/178)-MOD(Z,178))
1          IF(Z.LT.0) Z=Z+30323
1          TEMP=X/30269.0 + Y/30307.0 + Z/30323.0
1          RANDOM=TEMP-AINT(TEMP)
1          RANDOM=(40.0*RANDOM - 20.0)/100.0
1          RANDOM=RANDOM+1.0
1          ZRANDM=RANDOM
1    C
1    334   CONTINUE
1          IF(MM.EQ.1) RANDOM=1.0
1          IF(MM.EQ.2) RANDOM=1.0
1          IF(MM.EQ.4) RANDOM=ZRANDM
1    C
1    C     INITIALIZE FOR THE DAY
1    C
1          COST=0.0
1          ELD=0.0
1    C
1    C COOLING ENERGY CALCULATION
1    C
1          CHILL=0.0
1          PCHILL=0.0
1          PLEFT=0.0
1          IF(DAYTYP.EQ.2) ZLEFT=ZLEFT*0.95
1          IF(DAYTYP.EQ.1.AND.MM.EQ.3) PLEFT=(ZLEFT*1.22)/12000.0
1          IF(DAYTYP.EQ.1.AND.MM.EQ.3) ZLEFT=0.0
1          WRITE(*,*)' '
1          DO 30 I=1,24
2          ZLOSS=1.0
2          IF(MM.EQ.1.OR.MM.EQ.3) ZLOSS=1.05
2          IF(I.LT.9.OR.I.GT.22) ZLOSS=1.00
2          IF(DAYTYP.EQ.2) ZLOSS=1.0
2          ENER(I)=ENER(I)*RANDOM*ZLOSS
2          IF(MM.EQ.1) ZENER(I)=ENER(I)
2          WRITE(*,*)' '
2          IF((MM.EQ.1).AND.(I.GT.8.AND.I.LT.23)) TOTCL=TOTCL+ENER(I)
2          ZC=1.03
2          ZS=0.19
2          IF(MM.EQ.2.OR.MM.EQ.4) ZC=0.77
2          IF(I.LT.9.OR.I.GT.22) ZC=0.77
2          IF(MM.EQ.2.OR.MM.EQ.4) ZS=0.48
2          IF(DAYTYP.EQ.2) ZC=0.77
2          IF((MM.EQ.3.AND.DAYTYP.EQ.1).AND.(I.GT.8.AND.I.LT.23))
2         +ENER(I)=ZENER(I)
2          PC(I)=(ENER(I)*ZC)/12000.0
2          PSYS(I)=(ENER(I)*ZS)/12000.0
2          PFAN(I)=0.746*FANP
2     30   CONTINUE
1          WRITE(*,*)' '
1          IF(DAYTYP.EQ.2) GOTO 178
1          IF(MM.EQ.3.AND.RANDOM.GT.1.0) CHILL=(TOTCL/1.05)*(RANDOM-1.0)
1          IF(MM.EQ.3.AND.RANDOM.LT.1.0) ZLEFT=(TOTCL*0.95)*(1.0-RANDOM)
```

```
      IF(MM.EQ.3.AND.RANDOM.GT.1.0) PCHILL=CHILL*(0.96/12000.)*
     +(CHILL/2000000.0)
      WRITE(*,*)' '
178   CONTINUE
      IF(MM.EQ.3) TOTCL=0.0
C
C     GENERAL ENERGY CAL.
C
      DO 60 I=1,24
      PL(I)=1.2*QFLD
      IF(I.GT.15.OR.I.LT.8.OR.DAYTYP.EQ.2) PL(I)=1.2*QFLN
      PE(I)=EQPD
      IF(I.GT.15.OR.I.LT.8.OR.DAYTYP.EQ.2) PE(I)=EQPN
      ELD=ELD+PC(I)+PSYS(I)+PE(I)+PL(I)+PFAN(I)
      WRITE(*,*)' '
C
C  COST OF ELECTRICITY
C
      OFFPEK=0.02700
      ONPEK=0.05510
      IF(I.LT.9.OR.I.GT.22.OR.DAYTYP.EQ.2)COST=COST+
     +(PC(I)+PSYS(I)+PL(I)+PE(I)+PFAN(I))*OFFPEK
      IF(DAYTYP.EQ.2) GOTO 111
      IF(MM.EQ.2.OR.MM.EQ.4)OFFPEK=ONPEK
      IF(I.GT.8.AND.I.LT.23)COST=COST+(PFAN(I)+PL(I)+PE(I))*ONPEK
      IF(I.GT.8.AND.I.LT.23)COST=COST+(PC(I)+PSYS(I))*OFFPEK
111   CONTINUE
 60   CONTINUE
      IF(MM.EQ.3.AND.DAYTYP.EQ.1) ELD=ELD+PCHILL-PLEFT
      IF(MM.EQ.3.AND.DAYTYP.EQ.1) COST=COST+PCHILL*ONPEK-PLEFT*OFFPEK
      IF(MM.EQ.1) THCOST=COST
      IF(MM.EQ.1) TKWH=ELD
      IF(MM.EQ.2) CCOST=COST
      IF(MM.EQ.2) CKWH=ELD
      IF(MM.EQ.3) RTCOST=COST
      IF(MM.EQ.3) RTKWH=ELD
      IF(MM.EQ.4) RCCOST=COST
      IF(MM.EQ.4) RCKWH=ELD
      IF(MM.EQ.3) QP=PLEFT
      IF(MM.EQ.3) QC=PCHILL
      WRITE(*,*)' '
 333  CONTINUE
      WRITE(6,444) IDAYY,ZRANDM
      WRITE(6,445) THCOST,TKWH
      WRITE(6,446) CCOST,CKWH
      WRITE(6,447) RTCOST,RTKWH
      WRITE(6,448) RCCOST,RCKWH
      WRITE(6,450) QP,QC
      WRITE(6,449)
444   FORMAT(' DAY= ',I3,'  RANDOM = ',F5.3)
445   FORMAT(' COST WITH THERMAL STORAGE = $',F8.2,' , KWH = ',1PE11.4)
446   FORMAT(' COST WITH CONVENTIONAL COOLING = $',F8.2,' , KWH = ',
```

```
     +1PE11.4)
447   FORMAT(' COST WITH THERMAL STORAGE (RANDOM) = $',F8.2,' , KWH = ',
     +1PE11.4)
448   FORMAT(' COST WITH CONVENTIONAL COOLING (RANDOM) = $',F8.2,
     +' , KWH = ',1PE11.4)
450   FORMAT(' KWH LEFTOVER FROM PREVIOUS DAY = ',1PE11.4,/,
     +' KWH EXTRA FOR THE CHILLER FOR THE DAY = ',1PE11.4)
449   FORMAT(//)
1001  RETURN
      END
```

```
C
      SUBROUTINE TRANSF(BT,DT,UWRT,EPSILN,MC)
C
C
C *****************************************************
C *                                                   *
C *  THIS SUBROUTINE READS THE DATA FILE DTRSF.DAT WHICH  *
C *  CONTAINS B AND D TRANSFER FUNCTION COEFFICIENTS AND  *
C *  DESCRIPTIONS OF COMPONENTS FOR A NUMBER OF WALLS,    *
C *  THE LIST IS PRINTED AND INDEXED AND THE USER IS      *
C *  THE CHOICE OF PICKING A WALL TYPE FROM THE LIST OR   *
C *  ENTERING HIS OWN DESCRIPTION AND B AND D COEFFI-     *
C *  CIENTS PLUS THE U-VALUE. IF THE USER CHOOSES TO      *
C *  ENTER HIS OWN VALUES THEY ARE ADDED TO THE DATA SET  *
C *  AND WILL BE AVAILABLE THE NEXT TIME THE SUBROUTINE   *
C *  IS CALLED. THE MAXIMUM NUMBER OF WALL DESCIPTIONS    *
C *  IS LIMITED TO 100, WHICH SHOULD BE SUFFICIENT FOR    *
C *  MOST USERS NEEDS.                                    *
C *                                                   *
C *****************************************************
C
C
      CHARACTER*64 WRTYPE
      DIMENSION WRTYPE(100),B(7,100),D(7,100),UWRTWR(100),BT(7),DT(7)
      DIMENSION BR(7,4),DR(7,4),UWRTR(4)
      DATA BR/.78785E-02,.22323E-01,.3095E-02,.11659E-04,.4686E-10,
     +.15321E-16,0.,.155969E-01,.625E-01,.57019E-02,.113507E-04,
     +.221129E-13,0.,0.,.77579E-04,.26427E-02,.44282E-02,.90555E-03,
     +.21036E-04,.31931E-07,.15873E-11,.35383E-05,.6607E-03,.34298E-02,
     +.24933E-02,.31818E-03,.6204E-05,.10343E-07/
      DATA DR/1.0,-.29813,.5651E-02,-.12572E-04,.10942E-19,0.,0.,1.0,
     +-.41266,.624691E-01,-.38269E-10,.82367E-21,0.,0.,1.0,-1.02645,
     +.20712,-.98212E-02,.49224E-04,-.38092E-08,.54842E-13,1.0,
     +-1.67313,.82703,-.1083,.44096E-02,-.47704E-04,.26864E-08/
      DATA UWRTR/0.047,0.076,0.047,0.138/
C
C  COMMERCIAL BUILDINGS ONLY
C
      IB=3
C
      CALL DEBUG('TRANSF',1)
      IF(IB.NE.3)GO TO 3000
      IF(EPSILN.LE.25.) GO TO 9500
      OPEN(3,FILE='WTRSF.TXT',STATUS='OLD')
      GO TO 9600
 9500 CONTINUE
      OPEN(3,FILE='RTRSF.TXT',STATUS='OLD')
 9600 CONTINUE
      READ(3,100)I
```

```
     100  FORMAT(I3)
          DO 10 N=1,I
1         READ(3,200) WRTYPE(N),(B(J,N),J=1,7),(D(L,N),L=1,7),UWRTWR(N)
1    200  FORMAT(A64,/,5E13.6,/,5E13.6,/,4E13.6,F9.6)
1    10   CONTINUE
          CLOSE(3)
          IF(EPSILN.LE.25.)GO TO 8500
          OPEN(4,FILE='WTRSF.TXT',STATUS='NEW')
          GO TO 8600
     8500 CONTINUE
          OPEN(4,FILE='RTRSF.TXT',STATUS='NEW')
     8600 CONTINUE
          IF(IB.NE.3)GO TO 3000
          IF(EPSILN.GT.25.)WRITE(*,300)
          IF(EPSILN.LE.25.)WRITE(*,400)
     300  FORMAT(6(/),'   WALL, PARTITION & FLOOR DESCRIPTION LIST',//)
     400  FORMAT(6(/),'   ROOF DESCRIPTION LIST',//)
          LIMIT1=1
          LIMIT2=15
          IF(I.LE.15)LIMIT2=I
     2000 DO 30 K=LIMIT1,LIMIT2
1         WRITE(*,500)K,WRTYPE(K)
1    500  FORMAT(' (',I3,'): ',A64)
1    30   CONTINUE
          IF(LIMIT2.EQ.I) WRITE(*,600)
     600  FORMAT(/,' IF A SURFACE TYPE FROM THE LIST ABOVE IS SUITABLE',
         +/,' ENTER SURFACE NUMBER .IF YOU WISH TO INPUT YOUR OWN',
         +/,' SURFACE DESCIPTION AND TRANSFER FUNCTION COEFFICIENTS',
         +/,' ENTER (0).',/)
          INDEX=10
          IF(LIMIT2.EQ.I) CALL INPUTI(INDEX,'ENTER SURFACE NUMBER OR 0$')
          CALL DEBUG('CLTDS5',1)
          IF(LIMIT2.EQ.I) GO TO 1000
          LIMIT1=LIMIT2+1
          LIMIT2=LIMIT2+15
          IF(I.LE.LIMIT2)LIMIT2=I
          ICON=0
          CALL INPUTI(ICON,'PRESS RETURN TO CONTINUE LIST $')
          CALL DEBUG('CLTDS5',2)
          IF(ICON.EQ.0)GO TO 2000
     1000 IF(INDEX.EQ.0)I=I+1
          IF(INDEX.NE.0)GO TO 3100
          WRITE(*,1100)
     1100 FORMAT(///,' INPUT SURFACE DESCRIPTION - WRTYPE:(64 CHARACTERS ',
         +'OR LESS',/)
          READ(*,'(A64)')WRTYPE(I)
          WRITE(*,1150)
     1150 FORMAT(//,' INPUT B AND D COEFFICIENT IN E-TYPE FORMAT')
          DO 1110 II=1,7
1         WRITE(*,1200)II
1    1200 FORMAT(/,' INPUT B(',I1,') ----> ')
1         READ(*,'(E20.7)')B(II,I)
```

```
1   1110  CONTINUE
          DO 1120 II=1,7
1         WRITE(*,1160)II
1   1160  FORMAT(/,' INPUT D(',I1,') ----> ')
1         READ(*,'(E20.7)')D(II,I)
1   1120  CONTINUE
          WRITE(*,1170)
    1170  FORMAT(//,' INPUT SURFACE CONDUCTANCE IN F-TYPE FORMAT ---->')
          READ(*,'(F13.9)')UWRTWR(I)
          INDEX=I
          GO TO 3100
    3000  IF((IB.NE.3).AND.(EPSILN.LE.25.))IRES=1
          IF((IB.EQ.1).AND.(EPSILN.GT.25.))IRES=2
          IF((IB.EQ.2).AND.(EPSILN.GT.25.).AND.(MC.EQ.1))IRES=2
          IF((IB.EQ.2).AND.(EPSILN.GT.25.).AND.(MC.EQ.2))IRES=3
          IF((IB.EQ.2).AND.(EPSILN.GT.25.).AND.(MC.EQ.3))IRES=4
          DO 3200 KK=1,7
1         BT(KK)=BR(KK,IRES)
1         DT(KK)=DR(KK,IRES)
1   3200  CONTINUE
          UWRT=UWRTR(IRES)
          GO TO 7800
    3100  DO 3500 JJ=1,7
1         BT(JJ)=B(JJ,INDEX)
1         DT(JJ)=D(JJ,INDEX)
1   3500  CONTINUE
          UWRT=UWRTWR(INDEX)
          WRITE(4,201)I
    201   FORMAT(I3)
          DO 7700 N=1,I
1         WRITE(4,200)WRTYPE(N),(B(J,N),J=1,7),(D(L,N),L=1,7),UWRTWR(N)
1   7700  CONTINUE
          CLOSE(4)
          CALL DEBUG('TRANSF',2)
    7800  RETURN
          END
```

```
                                                   10-29-87
                                                   11:22:45
D Line# 1      7                   Microsoft FORTRAN77 V3.13 8/05/83
               SUBROUTINE SETINP
               CHARACTER*80 RFNAME,WFNAME,YORN
               LOGICAL WRTFIL,RDFIL,TMPOFF,BUGON
               COMMON/BUGFLG/ BUGON
               COMMON/INPFIL/ WRTFIL,RDFIL,LNCOUNT
               LNCOUNT=0
               WRTFIL = .FALSE.
               RDFIL = .FALSE.
               BUGON=.FALSE.
          10   DO 15 I=1,5
1         15   WRITE(*,*)
               WRITE(*,*) ' Enter:    1 - to use interactive keyboard input only'
               WRITE(*,*) '           2 - to read input data from a file   '
               WRITE(*,*) '           3 - to write the input data to a file'
               WRITE(*,*) '           4 - to use both options  (2)  and  (3)'
               IOPT=1
               CALL INPUTI(IOPT,'$')
               IF (IOPT .LT. 0) THEN
                  BUGON=.TRUE.
                  IOPT=-IOPT
               ELSE
                  BUGON=.FALSE.
               ENDIF
               IF (IOPT .LT. 1 .OR. IOPT .GT. 4) GOTO 10
               IF (IOPT .EQ. 1) RETURN
               IF (IOPT .EQ. 3 .OR. IOPT .EQ. 4) THEN
                   WFNAME='OUTFIL.DAT'
                   CALL INPUTT(WFNAME,'ENTER NAME OF FILE TO BE USED TO SAVE INPU
              $T DATA$')
                   OPEN(106,FILE=WFNAME,STATUS='NEW')
               ENDIF
               IF (IOPT .EQ. 2 .OR. IOPT .EQ. 4) THEN
                   RFNAME='INFIL.DAT'
                   CALL INPUTT(RFNAME,'ENTER NAME OF FILE TO BE USED FOR INPUT DA
              $TA$')
                   OPEN(105,FILE=RFNAME,STATUS='OLD')
                   RDFIL=.TRUE.
               ENDIF
               IF (IOPT .EQ. 4 .OR. IOPT .EQ. 3) WRTFIL= .TRUE.
               END
          C
          C
          C
               SUBROUTINE DEBUG(SUBNAM,LINE)
               CHARACTER*6 SUBNAM
               INTEGER LINE
               LOGICAL BUGON
               COMMON /BUGFLG/BUGON
               IF (BUGON) WRITE(*,*) '*** DEBUG ***  ',SUBNAM,'  ',LINE
```

```
      END

C
      SUBROUTINE INPUTT(TVAL,PROMPT)
      LOGICAL WRTFIL,RDFIL,TMPOFF,SAME
      CHARACTER FPRMPT*80,FLINE*80
      COMMON/INPFIL/ WRTFIL,RDFIL,LNCOUNT
      CHARACTER PROMPT*80,ALINE(80)*1,LINE*80,TYPE*1,FMAT*8,SVAL*80
     1,TSVAL(80)*1,TVAL*80
      REAL RVAL(40)
      EQUIVALENCE (ALINE,LINE)
      EQUIVALENCE(SVAL,TSVAL)
      SVAL=TVAL
C                           ********************************************
C                           * FIND LENGTH OF THE PROMPT STRING AND
C                           * DISPLAY WITH THE PROPER FORMAT
C                           ********************************************
      CALL DEBUG('INPUT ',1)
      TMPOFF=.FALSE.
      WRITE(*,*)
      CALL SLEN(PROMPT,LEN,80,FMAT)
      IF (LEN .NE. 0)  THEN
          WRITE(*,FMAT) PROMPT
          IF (RDFIL) THEN
             LNCOUNT=LNCOUNT+1
             READ(105,FMAT,END=199,ERR=199) LINE
             CALL STCOMP(PROMPT,LINE,SAME)
             IF (.NOT. SAME) THEN
                RDFIL=.FALSE.
                CALL BEEP
                WRITE(*,*)
                WRITE(*,*) '    ******** ERROR IN INPUT FILE '
                WRITE(*,*) '    ********    AT LINE ',LNCOUNT
                WRITE(*,*) '    ******** NOW USING KEYBOARD INPUT ONLY '
                WRITE(*,*)
             ENDIF
          ENDIF
          GOTO 120
 199      RDFIL = .FALSE.
          CALL BEEP
 120      IF (WRTFIL) WRITE(106,FMAT) PROMPT
      ENDIF
      CALL DEBUG('INPUT ',2)
      IF (LEN .EQ. -1) WRITE(*,*)  '****** WARNING - No ($) sign found
     +at end of PROMPT string *****'
C                           ********************************************
C                           * CHECK FOR PROPER TYPE AND STORE THE DEFAULT
C                           * INPUT VALUE IN THE DUMMY VARIABLES
C                           * ( RVAL  =  IVAL  =  SVAL )
C                           ********************************************
C                           ********************************************
C                           * CHECK THE DEFAULT STRING VALUE FOR ILLEGAL
```

```
C                        * CHARACTERS.  THIS COULD INDICATE AN IMPROPE
C                        * STRING LENGTH DEFINITION ERROR.
C                        *******************************************
         DO 2 I=80,1,-1
            ICH=ICHAR(TSVAL(I))
            IF ( (ICH .LT.32 .OR. ICH.GT.126) .AND. ICH.NE.0 ) THEN
               WRITE(*,*) '    ************ WARNING ************'
               WRITE(*,*) '    * Illegal chatacter found in TEXT *'
               WRITE(*,*) '    * variable  passed to  subroutine *'
               WRITE(*,*) '    * INPUT.  Type  TEXT  must  be  a *'
               WRITE(*,*) '    * string defined to be  of length *'
               WRITE(*,*) '    * CHARACTER*80.                   *'
               WRITE(*,*) '    ***********************************'
               GOTO 3
            ENDIF
 2       CONTINUE
C                        *******************************************
C                        * DISPLAY THE DEFAULT VALUE (EITHER REAL,INTE
C                        * OR TEXT)
C                        *******************************************
 3    CALL DEBUG('INPUT ',3)
      WRITE(*,'(1H ,A2,A77,A1)') '< ', SVAL, '>'
C                        *******************************************
C                        * READ THE USER'S RESPONSE AND RETURN IF
C                        * THE DEFAULT VALUE IS ACCEPTED.  OTHERWISE
C                        * PROCESS THE RESPONSE AND CHECK FOR PROPER
C                        * INPUT.
C                        *******************************************
 5    IF (RDFIL) THEN
         LNCOUNT=LNCOUNT+1
         READ(105,'(A80)',END=299,ERR=299) LINE
         GOTO 130
 299     RDFIL=.FALSE.
         CALL BEEP
         READ(*,'(A80)') LINE
 130     IF (ALINE(1) .EQ. '@') THEN
            CALL BEEP
            READ(*,'(A80)') LINE
            TMPOFF=.TRUE.
            RDFIL=.FALSE.
         ELSE
            WRITE(*,*) LINE
         ENDIF
      ELSE
         READ(*,'(A80)') LINE
      ENDIF
      FLINE=LINE
      IF (WRTFIL) THEN
         IF(ALINE(1) .EQ. '!' ) THEN
            WRITE(*,*)
            WRITE(*,*)
            WRITE(*,*)'*** PROGRAM TERMINATED AND DATAFILE SAVED***'
```

```
              WRITE(*,*)
              WRITE(*,*)
              STOP
           ENDIF
        ENDIF
        CALL DEBUG('INPUT ',4)
        IF (LINE .EQ. ' ') THEN
             IF (TMPOFF) RDFIL = .TRUE.
             IF (WRTFIL)  WRITE(106,'(A80)')FLINE
             CALL DEBUG('INPUT ',5)
             RETURN
        ELSE
C                            ********************************************
C                            * FOR TEXT SIMPLY RETURN THE STRING ENTERED
C                            ********************************************
                TVAL=LINE
                WRITE(*,*)
                IF (TMPOFF) RDFIL = .TRUE.
                IF (WRTFIL)  WRITE(106,'(A80)') FLINE
                CALL DEBUG('INPUT ',7)
                RETURN
        ENDIF
C                            ********************************************
C                            * TRAP FOR IMPROPER INPUT
C                            ********************************************
   40   WRITE(*,*) '      ****** INPUT ERROR ******       Please Re-enter'
        GOTO 5
        END
C
C
C
        SUBROUTINE INPUTI(IVAL,PROMPT)
        LOGICAL WRTFIL,RDFIL,TMPOFF,SAME
        CHARACTER FPRMPT*80,FLINE*80
        COMMON/INPFIL/ WRTFIL,RDFIL,LNCOUNT
        CHARACTER PROMPT*80,ALINE(80)*1,LINE*80,FMAT*8
        EQUIVALENCE (ALINE,LINE)
C                            ********************************************

C                            * FIND LENGTH OF THE PROMPT STRING AND
C                            * DISPLAY WITH THE PROPER FORMAT
C                            ********************************************
        CALL DEBUG('INPUT ',1)
        TMPOFF=.FALSE.
        WRITE(*,*)
        CALL SLEN(PROMPT,LEN,80,FMAT)
        IF (LEN .NE. 0)  THEN
            WRITE(*,FMAT) PROMPT
            IF (RDFIL) THEN
               LNCOUNT=LNCOUNT+1
               READ(105,FMAT,END=199,ERR=199) LINE
               CALL STCOMP(PROMPT,LINE,SAME)
               IF (.NOT. SAME) THEN
```

```
                RDFIL=.FALSE.
                CALL BEEP
                WRITE(*,*)
                WRITE(*,*) '    ******** ERROR IN INPUT FILE '
                WRITE(*,*) '    ********    AT LINE ',LNCOUNT
                WRITE(*,*) '    ******** NOW USING KEYBOARD INPUT ONLY '
                WRITE(*,*)
             ENDIF
          ENDIF
          GOTO 120
 199      RDFIL = .FALSE.
          CALL BEEP
 120      IF (WRTFIL) WRITE(106,FMAT) PROMPT
       ENDIF
       CALL DEBUG('INPUT ',2)
       IF (LEN .EQ. -1) WRITE(*,*)  '****** WARNING - No ($) sign found
      +at end of PROMPT string *****'
       CALL DEBUG('INPUT ',3)
       WRITE(*,*) '<', IVAL, '>'
C                          ********************************************
C                          * READ THE USER'S RESPONSE AND RETURN IF
C                          * THE DEFAULT VALUE IS ACCEPTED.  OTHERWISE
C                          * PROCESS THE RESPONSE AND CHECK FOR PROPER
C                          * INPUT.
C                          ********************************************
 5     IF (RDFIL) THEN
          LNCOUNT=LNCOUNT+1
          READ(105,'(A80)',END=299,ERR=299) LINE
          GOTO 130
 299      RDFIL=.FALSE.
          CALL BEEP
          READ(*,'(A80)') LINE
 130      IF (ALINE(1) .EQ. '@') THEN
             CALL BEEP
             READ(*,'(A80)') LINE
             TMPOFF=.TRUE.
             RDFIL=.FALSE.
          ELSE
             WRITE(*,*) LINE
          ENDIF
       ELSE
          READ(*,'(A80)') LINE
       ENDIF
       FLINE=LINE
       IF (WRTFIL) THEN
          IF(ALINE(1) .EQ. '!' ) THEN
             WRITE(*,*)
             WRITE(*,*)
             WRITE(*,*)'*** PROGRAM TERMINATED AND DATAFILE SAVED***'
             WRITE(*,*)
             WRITE(*,*)
             STOP
```

```
            ENDIF
          ENDIF
          CALL DEBUG('INPUT ',4)
          IF (LINE .EQ. ' ') THEN
              IF (TMPOFF) RDFIL = .TRUE.
              IF (WRTFIL)  WRITE(106,'(A80)')FLINE
              CALL DEBUG('INPUT ',5)
              RETURN
          ELSE
                DO 10 I=80,1,-1
                    LAST=I
                    IF (ALINE(I) .NE. CHAR(32)) GOTO 20
   10           CONTINUE
C                           ********************************************
C                           * RIGHT JUSTIFY THE ENTRY
C                           ********************************************
   20           IOFFSET=80-LAST
                DO 30 I=LAST,1,-1
                    ALINE(I+IOFFSET)=ALINE(I)
                    ALINE(I)=CHAR(32)
   30           CONTINUE
C                           ********************************************
C                           * READ THE INPUT VALUE FROM THE CHARACTERS
C                           * IN VARIABLE LINE.  BRANCH TO THE TRAP FOR
C                           * IMPROPER INPUT IF AN ERROR OCCURS.
C                           ********************************************
                READ(LINE,'(I80)',ERR=40) IVAL
                WRITE(*,*)
                IF (TMPOFF) RDFIL = .TRUE.
                IF (WRTFIL)  WRITE(106,'(A80)') FLINE
                CALL DEBUG('INPUT ',6)
                RETURN
                WRITE(*,*)
                IF (TMPOFF) RDFIL = .TRUE.
                IF (WRTFIL)  WRITE(106,'(A80)') FLINE
                CALL DEBUG('INPUT ',7)
                RETURN
          ENDIF
C                           ********************************************
C                           * TRAP FOR IMPROPER INPUT
C                           ********************************************
   40     WRITE(*,*) '      ****** INPUT ERROR ******       Please Re-enter'
          GOTO 5
          END
C
C
C
          SUBROUTINE INPUTR(RVAL,PROMPT)
          LOGICAL WRTFIL,RDFIL,TMPOFF,SAME
          CHARACTER FPRMPT*80,FLINE*80
          COMMON/INPFIL/ WRTFIL,RDFIL,LNCOUNT
          CHARACTER PROMPT*80,ALINE(80)*1,LINE*80,FMAT*8
          EQUIVALENCE (ALINE,LINE)
```

```
C                           *********************************************
C                           * FIND LENGTH OF THE PROMPT STRING AND
C                           * DISPLAY WITH THE PROPER FORMAT
C                           *********************************************
      CALL DEBUG('INPUT ',1)
      TMPOFF=.FALSE.
      WRITE(*,*)
      CALL SLEN(PROMPT,LEN,80,FMAT)
      IF (LEN .NE. 0)  THEN
         WRITE(*,FMAT) PROMPT
         IF (RDFIL) THEN
            LNCOUNT=LNCOUNT+1
            READ(105,FMAT,END=199,ERR=199) LINE
            CALL STCOMP(PROMPT,LINE,SAME)
            IF (.NOT. SAME) THEN
               RDFIL=.FALSE.
               CALL BEEP
               WRITE(*,*)
               WRITE(*,*) '    ******** ERROR IN INPUT FILE '
               WRITE(*,*) '    ********    AT LINE ',LNCOUNT
               WRITE(*,*) '    ******** NOW USING KEYBOARD INPUT ONLY '
               WRITE(*,*)
            ENDIF
         ENDIF
         GOTO 120
  199    RDFIL = .FALSE.
         CALL BEEP
  120    IF (WRTFIL) WRITE(106,FMAT) PROMPT
      ENDIF
      CALL DEBUG('INPUT ',2)
      IF (LEN .EQ. -1) WRITE(*,*)  '****** WARNING - No ($) sign found
     +at end of PROMPT string *****'
      CALL DEBUG('INPUT ',3)
      WRITE(*,*) '<', RVAL, '>'
C                           *********************************************
C                           * READ THE USER'S RESPONSE AND RETURN IF
C                           * THE DEFAULT VALUE IS ACCEPTED.  OTHERWISE
C                           * PROCESS THE RESPONSE AND CHECK FOR PROPER
C                           * INPUT.
C                           *********************************************
    5 IF (RDFIL) THEN
         LNCOUNT=LNCOUNT+1
         READ(105,'(A80)',END=299,ERR=299) LINE
         GOTO 130
  299    RDFIL=.FALSE.
         CALL BEEP
         READ(*,'(A80)') LINE
  130    IF (ALINE(1) .EQ. '@') THEN
            CALL BEEP
            READ(*,'(A80)') LINE
            TMPOFF=.TRUE.
            RDFIL=.FALSE.
```

```
            ELSE
               WRITE(*,*) LINE
            ENDIF
         ELSE
            READ(*,'(A80)') LINE
         ENDIF
         FLINE=LINE
         IF (WRTFIL) THEN
            IF(ALINE(1) .EQ. '!' ) THEN
               WRITE(*,*)
               WRITE(*,*)
               WRITE(*,*)'*** PROGRAM TERMINATED AND DATAFILE SAVED***'
               WRITE(*,*)
               WRITE(*,*)
               STOP
            ENDIF
         ENDIF
         CALL DEBUG('INPUT ',4)
         IF (LINE .EQ. ' ') THEN
              IF (TMPOFF) RDFIL = .TRUE.
              IF (WRTFIL)  WRITE(106,'(A80)')FLINE
              CALL DEBUG('INPUT ',5)
              RETURN
         ELSE
                  DO 10 I=80,1,-1
                       LAST=I
                       IF (ALINE(I) .NE. CHAR(32)) GOTO 20
   10             CONTINUE
C                               ********************************************
C                               * RIGHT JUSTIFY THE ENTRY
C                               ********************************************
   20             IOFFSET=80-LAST
                  DO 30 I=LAST,1,-1
                       ALINE(I+IOFFSET)=ALINE(I)
                       ALINE(I)=CHAR(32)
   30             CONTINUE
C                               ********************************************
C                               * READ THE INPUT VALUE FROM THE CHARACTERS
C                               * IN VARIABLE LINE.  BRANCH TO THE TRAP FOR
C                               * IMPROPER INPUT IF AN ERROR OCCURS.
C                               ********************************************
                  READ(LINE,'(F80.0)',ERR=40) RVAL
                  WRITE(*,*)
                  IF (TMPOFF) RDFIL = .TRUE.
                  IF (WRTFIL)  WRITE(106,'(A80)') FLINE
                  CALL DEBUG('INPUT ',6)
                  RETURN
                  WRITE(*,*)
                  IF (TMPOFF) RDFIL = .TRUE.
                  IF (WRTFIL)  WRITE(106,'(A80)') FLINE
                  CALL DEBUG('INPUT ',7)
                  RETURN
         ENDIF
```

```
C                            ********************************************
C                            * TRAP FOR IMPROPER INFUT
C                            ********************************************
   40 WRITE(*,*) '      ****** INPUT ERROR ******      Please Re-enter'
      GOTO 5
      END
C
C
C
C
C
C
      SUBROUTINE SLEN(STRING,LEN,MAXLEN,FMAT)
C*********************************************************************
C            Subroutine to determine the length and format specification
C            of STRING.  The character '$'  MUST be the last character
C            of the string.  The subroutine searches each character of
C            STRING until '$' is found or MAXLEN is exceeded.  If '$' is
C            found, LEN is set equal to (location of '$'  - 1). Otherwis
C            LEN = -1.  If LEN does not equal 0, the output format FMAT
C            determined based on the length of the string.  If LEN = -1,
C            output format is based on the maximum length of the string
C            MAXLEN.
C
C            Variables:
C
C                  STRING    Character string whose length is to be
C                            determined.
C
C                  LEN       The length of the string is returned in LEN
C
C                  MAXLEN    The maximun length of STRING
C
C                  FMAT      An 8 character string used to return the
C                            Format specification required to WRITE
C                            the variable STRING.
C
C*********************************************************************
      CHARACTER STRING(1)*1,FMAT*8
      DO 10 I=1,MAXLEN
          IF ( STRING(I) .EQ. '$') THEN
              LEN=I-1
              CALL STRFMT(LEN,FMAT)
              RETURN
          ENDIF
   10 CONTINUE
      LEN=-1
      CALL STRFMT(MAXLEN,FMAT)
      RETURN
      END
C
C
C
```

```
      SUBROUTINE STCOMP(STR1,STR2,SAME)
C
C           Subroutine to compare two strings .
C
      CHARACTER STR1(1)*1,STR2(1)*1
      LOGICAL SAME
      SAME=.TRUE.
      DO 10 I=1,80
         IF (STR1(I) .EQ. '$') RETURN
         IF (STR1(I) .NE. STR2(I)) THEN
            SAME=.FALSE.
            RETURN
         ENDIF
10    CONTINUE
      END
C
C
C
      SUBROUTINE STRFMT(LEN,FORMT)
C**********************************************************************
C           Subroutine to determine the format specification required
C           to output a string of length LEN.
C
C                 LEN         Length of string to be output.
C
C                 FORMT       An * character string used to return the
C                             format specification
C
C**********************************************************************
      CHARACTER FORMT*8,TF*8,ATF(8)*1
      EQUIVALENCE (TF,ATF)
      TF='(1X,A  )'
      IF (LEN .GE.10) THEN
         LEN1=LEN/10
         LEN2=LEN-LEN1*10
         ATF(6)=CHAR(LEN1+48)
         ATF(7)=CHAR(LEN2+48)
      ELSE
         ATF(6)=CHAR(LEN+48)
      ENDIF
      FORMT=TF
      END
C
C
C
      SUBROUTINE BEEP
      WRITE(*,*) CHAR(7)
      END
```

APPENDIX C

SAMPLE OUTPUT

***DAYTYP= 1

TOTAL COOLING LOAD FOR 24 HOURS (BTU/HR)

1	0.	5	0.	9	2281347.	13	4386663.	17	5511414.	21	2604965.
2	0.	6	1453835.	10	2718031.	14	4864343.	18	4287839.	22	2105062.
3	0.	7	1296781.	11	3423530.	15	5305188.	19	3736319.	23	0.
4	0.	8	1237590.	12	3944365.	16	5412377.	20	3299229.	24	0.

DAY= 183 RANDOM = .979
COST WITH THERMAL STORAGE = $ 353.63 , KWH = 1.0116E+04
COST WITH CONVENTIONAL COOLING = $ 525.65 , KWH = 1.0354E+04
COST WITH THERMAL STORAGE (RANDOM) = $ 353.45 , KWH = 1.0110E+04
COST WITH CONVENTIONAL COOLING (RANDOM) = $ 518.23 , KWH = 1.0210E+04
KWH LEFTOVER FROM PREVIOUS DAY = .0000E+00
KWH EXTRA FOR THE CHILLER FOR THE DAY = .0000E+00

***DAYTYP= 1

TOTAL COOLING LOAD FOR 24 HOURS (BTU/HR)

1	0.	5	0.	9	2818266.	13	5042815.	17	5719367.	21	2681480.
2	0.	6	1682675.	10	3599091.	14	5521413.	18	4384008.	22	2201111.
3	0.	7	1491812.	11	4098406.	15	5648200.	19	3824186.	23	0.
4	0.	8	1455230.	12	4556545.	16	5614734.	20	3299713.	24	11117.

DAY= 184 RANDOM = .940
COST WITH THERMAL STORAGE = $ 369.82 , KWH = 1.0716E+04
COST WITH CONVENTIONAL COOLING = $ 558.40 , KWH = 1.0983E+04
COST WITH THERMAL STORAGE (RANDOM) = $ 366.07 , KWH = 1.0577E+04
COST WITH CONVENTIONAL COOLING (RANDOM) = $ 535.60 , KWH = 1.0541E+04
KWH LEFTOVER FROM PREVIOUS DAY = 1.1697E+02
KWH EXTRA FOR THE CHILLER FOR THE DAY = .0000E+00

VITA

James Craig Tucker

Candidate for the Degree of

Master of Science

Thesis: USE OF WEATHER FORECAST DATA TO PREDICT THE ICE REQUIREMENTS FOR A THERMAL STORAGE SYSTEM

Major Field: Mechanical Engineering

Biographical:

Personal Data: Born in Pawnee, Oklahoma, October 16, 1963, the son of James A. and Mary A. Tucker. Married to Julia A. Gunter on August 3, 1985.

Education: Graduated from Cleveland High School, Cleveland, Oklahoma, in May, 1981; received the Bachelor of Science degree in Mechanical Engineering from Oklahoma State University in May, 1985; completed requirements for the Master of Science degree at Oklahoma State University in December, 1987.

Professional Experience: Teaching Assistant, School of Mechanical and Aerospace Engineering, Oklahoma State University, August, 1986, to December, 1987; Engineer, Shell Offshore Inc., June, 1985, to June, 1986.

Professional Organizations: Pi Tau Sigma, American Society of Mechanical Engineers, American Society of Heating, Ventilating, and Refrigerating Engineers.