Chapter 3
Problem Solving

Problem Solving
The Backbone of Programming

Problem solving, or breaking down the solution to a problem into sequential steps is by far the most difficult part of computer programming. There are many activities we can engage in that will help sharpen our problem solving skills. These activities include working crossword puzzles, playing bridge or chess, and working on math problems. It is a skill that can be developed with practice and one that leaves us quickly when the mind is not pushed beyond television and movies. If you think our ability to concentrate has not been compromised by the media just watch a few television commercials and see how long you are focused on any one image or thought. Not long! The ability to focus on one small part of a solution while not getting overwhelmed by the entire problem is something we will spend a great deal of time on in this class. Coding programs that have been well designed is an easy task – learning how to do the design well takes practice.

This section is designed to help get us started using some non-computer related and hopefully fun puzzles to "play" with. There are various techniques you will use such as analogy, trial and error, process of elimination, visualization and some very basic math. Have fun!

Favorite Television Shows

Six friends, seated around a table, are discussing their favorite TV programs: “Friends”, golf tournaments, “My Wife and Kids”, “20/20”, “Everybody Loves Raymond”, and “Will and Grace.” Based on the following information, determine each person’s favorite TV program and where each person sits: (assume Scott is seated as shown below)

A) The boy who likes golf tournaments sits directly to the right of Doug

B) Scott sits between the two people who watch “Will and Grace” and golf tournaments

C) No one sits between Trish and Debbie

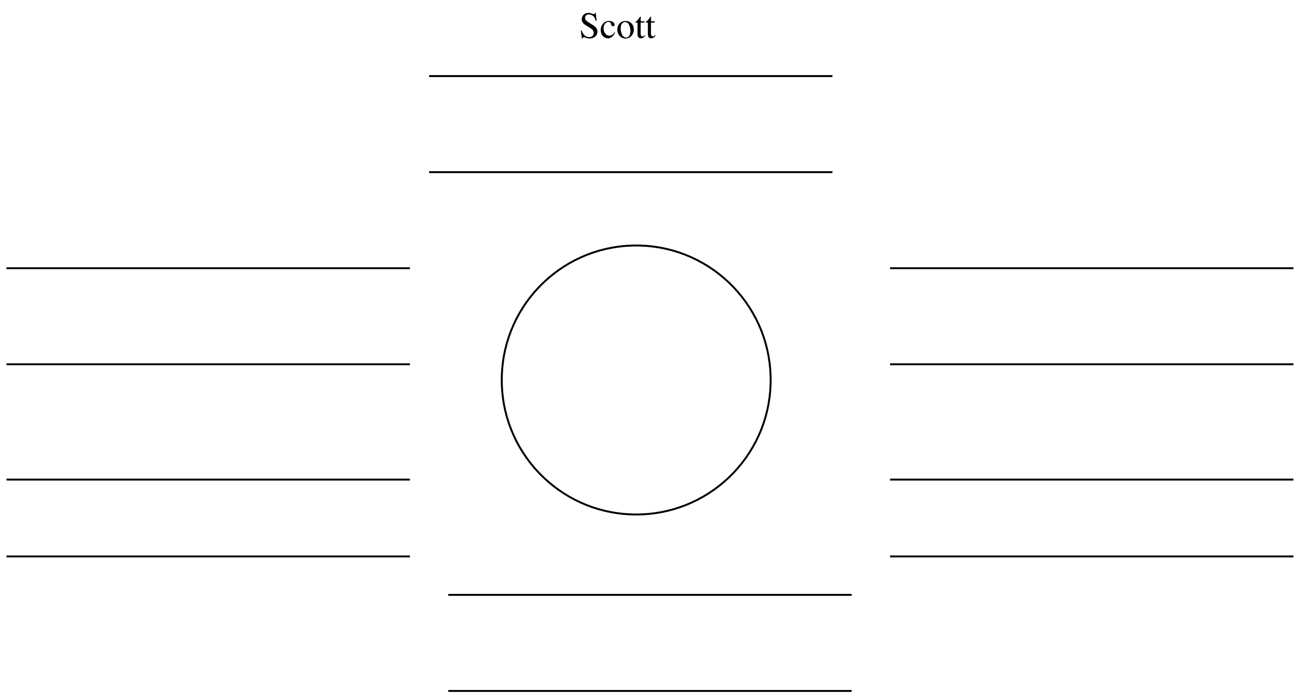
D) Debbie sits directly to the left of the boy who likes “Friends”

E) The boy who likes “My Wife and Kids” does not sit next to a girl

F) Debbie does not care for “Everybody Loves Raymond”

G) Carlos does not sit next to a girl

H) Erik does not sit next to the girl who likes “Will and Grace”



Do You Owe Me Money??

There are three brothers named John, James and William, who look exactly alike. John and James always lie, but William always tells the truth. You meet one of the brothers on the street one day, and want to know if he is John (because John owes you money). You are allowed to ask ONE question that can be answered “yes” or “no”, but the question may not contain more than 3 words. What question would you ask?

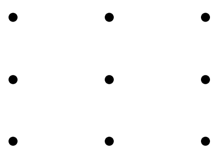
The Three Hats

Three men are condemned to die, one of whom is blind. The king decides he will offer them an opportunity to be set free. The three men are arranged in a circle facing one another. The king produces 5 hats: 2 black and 3 white. The king places a hat on the head of each person and then destroys the two remaining hats. The men have no idea which hats have been destroyed. The king instructs them, “The first one of you who can tell me the color of his hat will be set free.” A period of time passes in silence and then finally the blind man tells the king the color of his hat and is set free. What color hat was the blind man wearing and how did he know?
This does not have a “trick” answer – your answer should be very logical and well thought out. Be able to explain your answer from the viewpoint of each of the three prisoners.

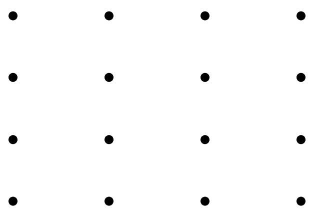
Hint: Each sighted man can see the blind man’s hat as well as that of the other sighted man. What does the pause in time infer?

Miscellaneous Problems

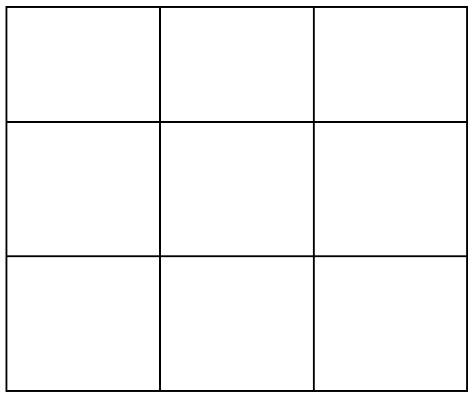
1. Draw four line segments through the 9 points without lifting your pencil or retracing a line.



2. Draw six line segments through the 16 points without lifting your pencil or retracing a line.



3. How many squares are in the figure? Hint: There are more than 9.



4. A farmer has some chickens and some goats. Together there are 43 heads and 108 legs. How many chickens does the farmer have? How many goats?

5. Step 1: Write down the year of your birth.
   Step 2: Add it to the year of some important event in your life.
   Step 3: Add the age you will be this year.
   Step 4: Add the number of years since the important event took place.
   Step 5: Multiply the current year by 2.
   Step 6: What did you discover? Try to explain why the answers in steps 4 & 5 are the same.

6. A mechanic's brother won the car race. But the man who won had no brother. How is this possible?

Basic Problem Solving Vocabulary

There is a basic vocabulary that is used in problem solving and programming. It is necessary that you develop this vocabulary if you are to understand the lectures and communicate with lab personnel. There are several important vocabulary pages in the shrink wrap and it is essential that you commit them to memory. These definitions will appear on several quizzes as well as on larger exams.

ALGORITHM – A step by step process for solving a problem.

TOP-DOWN DESIGN – A design methodology used to break a problem into small parts progressing from the general to the specific. This is also called design by level or divide and conquer.

STRUCTURE CHART – A “picture” of the top down design. This chart shows the basic program modules and how they are related.

MODULE – One small part of the solution.

FLOWCHART – A “picture” of an algorithm using specific symbols to indicate various programming constructs.

PSEUDOCODE – A terse, English like description of your algorithm.

DESK CHECK - Working the algorithm with pencil and paper. Draw each memory location and check the algorithm as though you were the computer.

DOCUMENTATION - Anything that provides information about a program. Internal comments in the program, data tables that describe variables and named constants in the program, and external documents such as user's manuals.

Phases of Problem Solving

Design Phase:

- Define the problem.
- Design the general solution.
- Test the solution.
- Modify the solution if necessary and retest.

Implementation Phase:

- Code the program in the selected language. Documentation should be done as you code.
- Test the program by running it on the computer.
- Integrate the program with other software as necessary.

Control/Logic Structures

Dykstra proposed that all computer programs could be written using just three basic control or logic structures. Sequence, selection, and repetition. Prior to this a construct called a branch (associated with the word GOTO) was used extensively. The use of the branch resulted in programs that were poorly written, difficult to follow, even more difficult to debug and virtually impossible to modify. Modern languages such as Pascal, Algol, ADA, C, C++ and Java are built around these three structures.

Sequence – executing instructions one after another as they appear in the program.

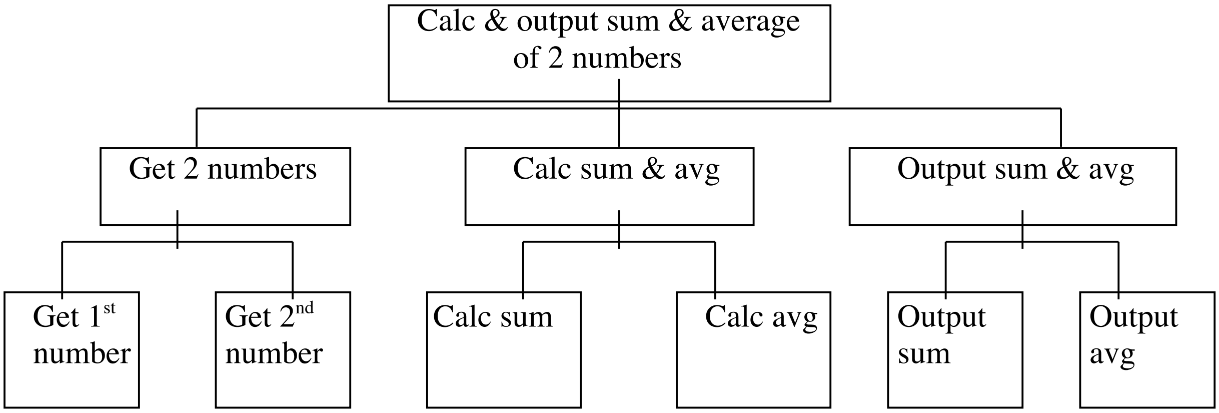
Selection – choosing between two or more alternative actions.

Repetition – repeating a block of code until a condition is met.

A Simple Sequence Problem

We want to develop an algorithm (a step by step process) for a program to calculate the sum and average of two numbers entered by the user of the program.

Step 1: Structure Chart



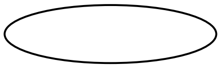
The structure chart shows the relationship among the various components of the program. The top level is a general description of the problem to be solved where the lower levels show all the detail or specifics of the solution. **NOTE: The second level of a structure chart attempts to break the solution into three basic parts – an *input* module, a *processing* module and an *output* module**.

Step 2: Flowchart

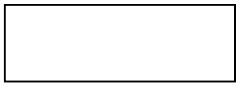
A flowchart provides a detailed picture of the algorithm using special symbols to represent various program statements. A flowchart will always be drawn from top to bottom showing the exact order of the steps. Before we can proceed with the flowchart we need to think about things we will want to computer to store as the program runs. We need it to keep a place in memory for each of the two numbers, the sum of the numbers and the average of the numbers. The term used for these memory locations is *variable*. A *variable* is a place in memory that contains a data value that may be changed during program execution. We do not know exactly where in memory these values will be placed so we will reference them by name. Giving a name to a memory location is called *symbolic addressing*.

We need names for the locations we are requesting. **NOTE: The names that appear on the flowchart must match exactly the names actually used in your program.** We will use the following names: firstNum, secondNum, sum, and average.

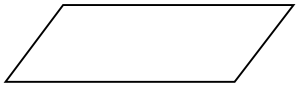
There are three flowchart symbols necessary for programs involving simple sequence.



Begin/End of a block

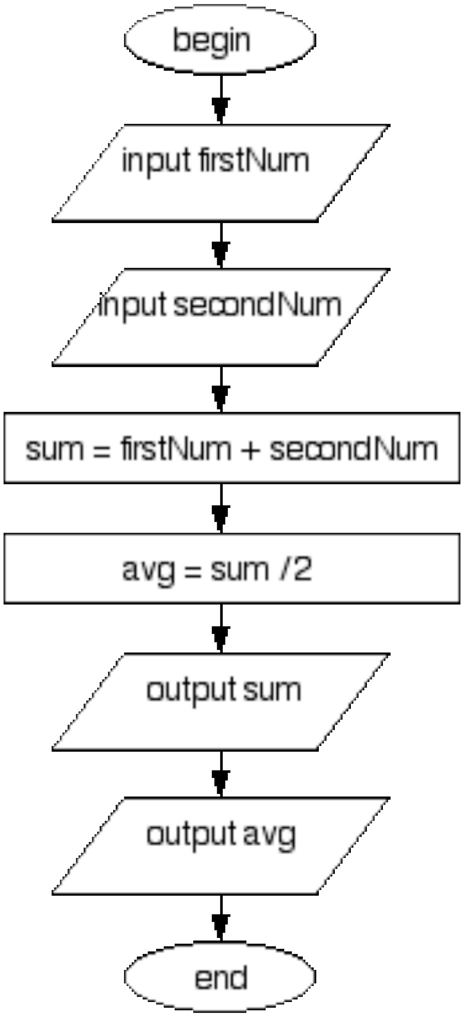


Process



Input/Output

Note that the flowchart begins with the processes outlined on the bottom left of the structure chart – the lowest level where the specific steps have been outlined.



Simple Sequence Exercises

1. Design the algorithm for a program that calculates the total of a retail sale. The program should ask the user for the following: the retail price of the item being purchased and the sales tax rate. Once the information has been entered the program should calculate and display the following: the sales tax for the purchase and the total sale. Draw the flowchart for this algorithm.

2. Design the algorithm for a program that calculates the current balance in a savings account. The program should obtain from the user the following information: the starting balance, the total amount of deposits made, the total amount of withdrawals made, and the monthly interest rate. After the program has calculated the current balance, it should be displayed on the screen. Assume one input for deposits and one input for withdrawals. Draw the flowchart for this algorithm.

3. Draw a flowchart to match the following pseudocode.

```
- Give variable num1 a starting value of 5
- Give variable num2 a starting value of 10
- Add 7 to num2
- Store the value num1 times num2 in variable num3
- Store the value num2 minus num1 in num2
- Output num1, num2 and num3
```

4. Trace the steps in your flowchart from question 3 and show the output produced by this program.

5. Obtain from the user an hourly pay rate and the number of hours worked for the week. Calculate their pay for the week (no overtime, and no taxes). Output the result.

Name ______________________________ Due Date __________

Simple Sequence Problems

1. Draw a structure chart for a program that will obtain from the user a length and width for a rectangle and the radius of a circle. Calculate and output the area and perimeter of the rectangle and the area and circumference of the circle. List the necessary variables, draw the flowchart (using the same names as your variable list), and perform a desk check using inputs of 7 and 10 for the rectangle and 4 for the circle.

2. Draw the structure chart for a program that will obtain from the user a current annual salary and a percent increase due on that salary. Assume that the new pay rate should have been in effect on January 1, and it is now July 1. Calculate, store, and output the new annual salary, the new monthly salary, and the retroactive pay due. Make a list of variables, draw the flowchart (using the same names as your variable list), and perform a desk check using a starting salary of $60,000 and a pay increase of 5%.

3. Draw a structure chart for a program that obtains from the user, an annual interest rate, the term of the loan (in years), and the amount borrowed. Your program should use the following formula:

$$\text{payment} = \frac{\text{rate}(1 + \text{rate})^{\text{term}}}{(1 + \text{rate})^{\text{term}} - 1} \text{ X loan amount}$$

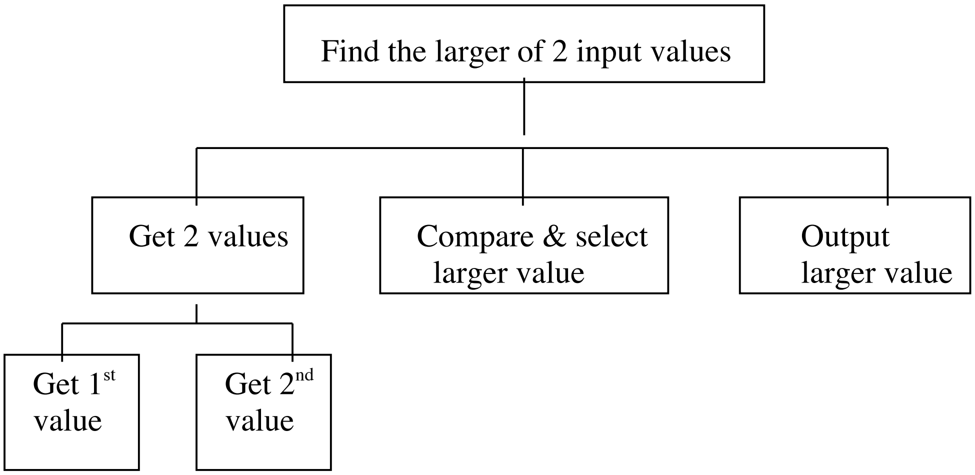
You will need to convert the interest rate to a monthly rate in decimal format, and the term into months. Make a list of necessary variables for this program. Draw the flowchart (using the same names in your variable list), and perform a desk check using test data of $120,000 for the loan amount, 30 years for the term, and an annual interest rate of 7%.

TURN IN THIS SHEET ON TOP OF YOUR NEATLY DRAWN STRUCTURE CHARTS AND FLOWCHARTS.

A Selection Problem

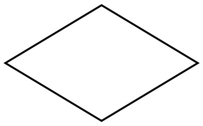
Following is the design for an algorithm to select the larger of two input values. We will make the assumption that the two numbers are not equal.

Step 1: Structure Chart

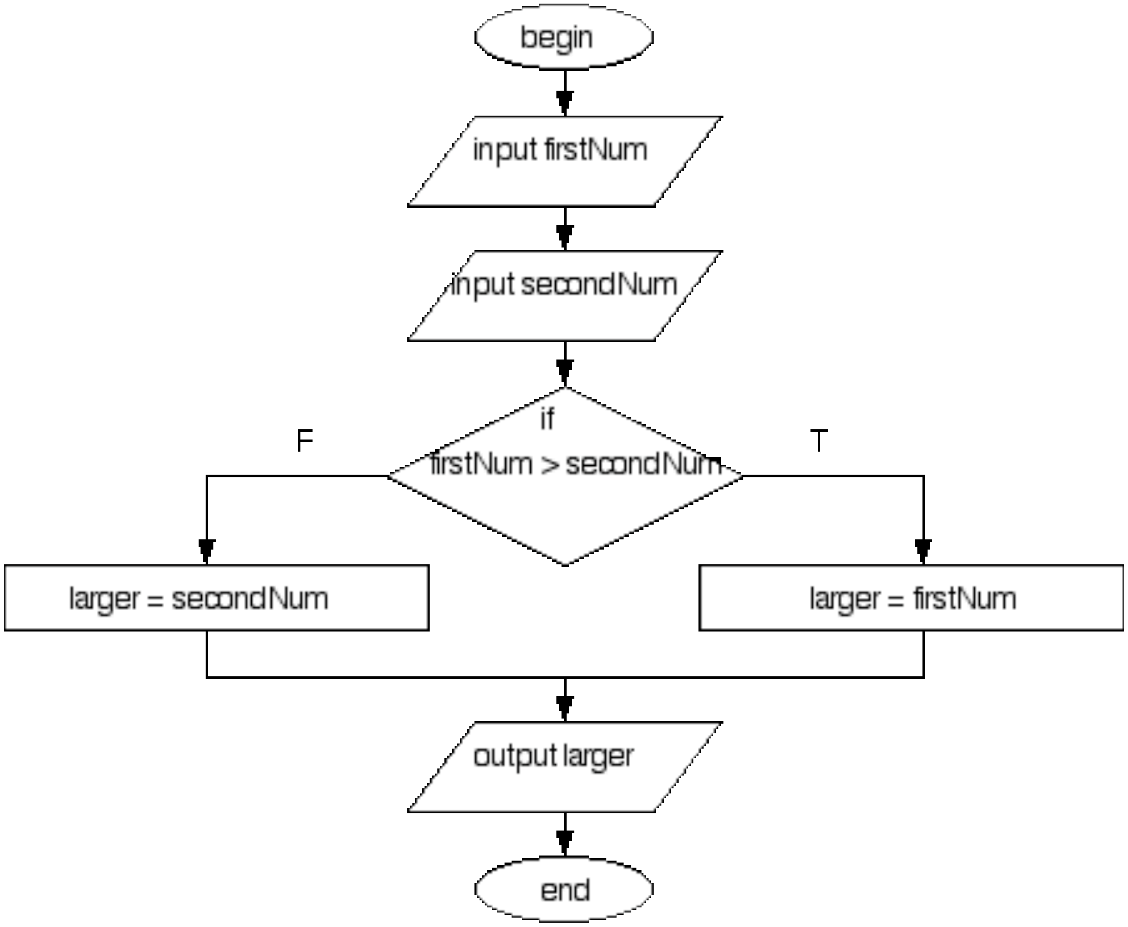

Find the larger of 2 input values
Get 2 values
Compare & select larger value
Output larger value
Get 1st value
Get 2nd value

Step 2: Flowchart

The following variable names will be used: firstNum, secondNum and larger. We will need one more flowcharting symbol:



Decision


begin
input firstNum
input secondNum
if
firstNum > secondNum
F
T
larger = secondNum
larger = firstNum
output larger
end

Selection Exercises *if*

1. Obtain the length and width of a rectangle from the user. Calculate and output the area. If the length and width are equal, output a message indicating that the figure is a square. Make a list of variables, draw the flowchart, and perform a desk check using the following: 4, 8, 5, 5

2. Obtain three test scores from a student. Calculate their average test score and output this value. If their average score is 75% or more output a message indicating that they may proceed to the next class. Make a list of variables, draw the flowchart, and perform a desk check using the following: 45, 55, 75, 80, 75, 88

Selection Exercises *if/else*

1. Obtain a name and age from the user. If the user is 16 or older, output a message indicating they are old enough to drive. For people under 16, output a message indicating how many years they must wait before they can drive legally.

2. Obtain from the user an hourly pay rate and the number of hours worked for the week. Calculate and output their weekly pay according to the following:

- Regular pay is the pay up to 40 hours.
- Overtime pay is pay for the hours over 40. Overtime is paid at a rate of 1.5 times the hourly rate.
- Gross pay is the sum of the regular pay and the overtime pay. Hint: This calculation is not conditional.

3. Obtain a temperature in degrees Fahrenheit from the user. If the temperature is 80 degrees or more, display a message that says "Go play golf" otherwise display a message stating "It's too cold to be outside." Make a variable list, flowchart, and perform a desk check using the following values: 95, 70

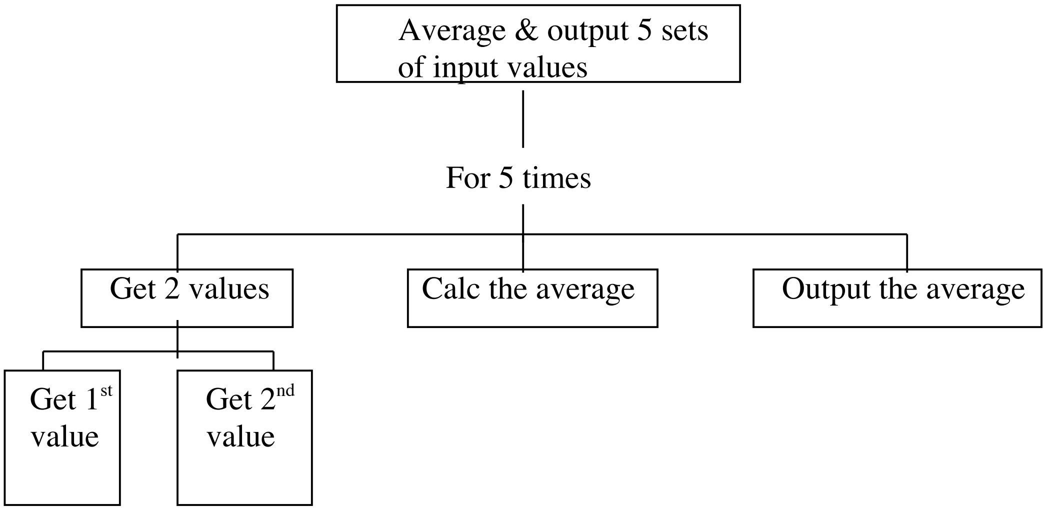
4. Obtain a temperature in degrees Fahrenheit from the user. If the temperature is 80 degrees or more display the message "Go play golf" otherwise, if the temperature is 70 - 79 degrees display the message "Put on a jacket", otherwise display the message "It is way too cold." Make a variable list, flowchart, and perform a desk check using the following values: 95, 72, 50

5. Obtain three numbers from the user. We will assume that the numbers are not the same. Find the largest of these three numbers and store that value in a variable. Output this variable with an appropriate message. Make a variable list, flowchart, and perform a desk check using the following values: 3, 1, 2, 1, 2, 3, 2, 3, 1

A Repetition Problem

Following is an algorithm to average and output five sets of input values.

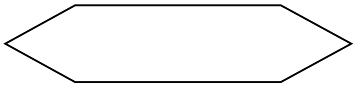
Step 1: Structure chart



Step 2: Flowchart

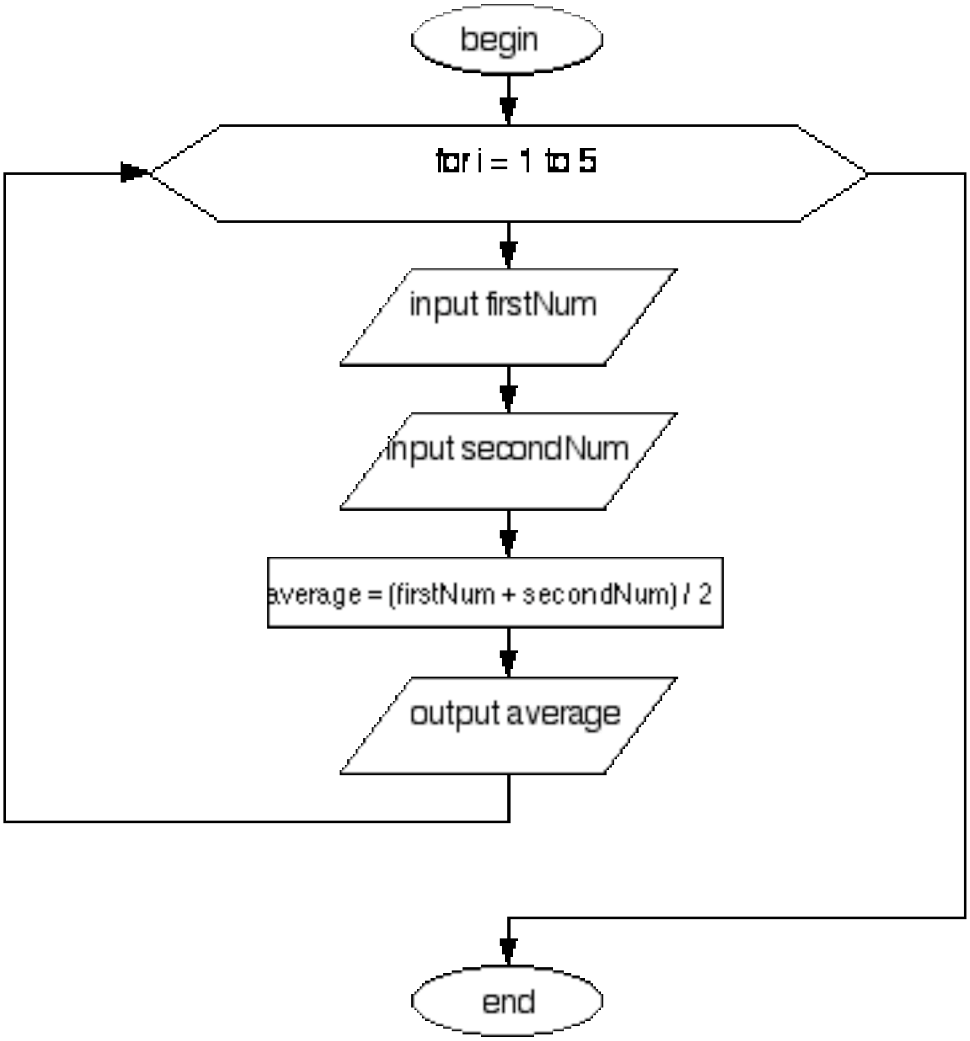
Variable names: firstNum, secondNum, average, and i (the variable to control the loop)

New flowchart symbol:



Counter loop

Note that as the algorithms become a little more complex less of the actual detail is visible in the structure chart. The flowchart will always give the most detailed representation and is what should be used when you code the program.


begin
for i = 1 to 5
input firstNum
input secondNum
average = (firstNum + secondNum) / 2
output average
end

Name ______________________________________
Due Date ___________

Repetition Exercises
`for` Loop

1. Draw a flowchart for a program that will compute the average of 10 exam scores entered by the user. List the variables needed for this program. Perform a desk check using the following values: 75, 25, 51, 49, 88, 12, 22, 23, 52, and 78.

2. Modify problem #1 to allow the user to input the total number of exam scores. List the necessary variables and draw the flowchart. Perform a desk check with the following values: 5, 23, 77, 75, and 50.

3. Draw a flowchart for a program that will obtain 10 exam scores from the user and determine whether the score is passing (a score of 60 or above) or failing. Your algorithm should count the number of passing and failing scores. Output the average of the scores, the number of failing scores, and the number of passing scores. List the necessary variables for this program. Perform a desk check with the following values: 75, 25, 51, 49, 88, 12, 22, 23, 52, and 78.

4. Draw a flowchart for a program that will compute and output the sales tax due with a tax rate of 7% and the total purchase price of 4 items. List the necessary variables for this program. Perform a desk check with the following values: 2.99, 3.99, 7.50, and 2.75.

TURN IN (IN THIS ORDER & STAPLED IN THE UPPER LEFT CORNER)

- this sheet
- flowcharts (DONE ON THE COMPUTER ONLY)

Name ___________________________________

Due Date _______________

Repetition Exercises
`while` Loop

1. You are to flowchart a program for Aunt Ellen's Egg Ranch. The user will enter the number of eggs gathered and the program will output the number of dozens as well as the number of excess eggs. Continue this process until a negative number is entered. Example:

> Enter the number of eggs gathered: 43
> You have 3 dozen and 7 eggs.

List the necessary variables for this program. Perform a desk check with the following input values: 24, 8, 15, and -999.

2. Draw the flowchart for a program that accepts an unknown number of ages from the keyboard until zero is entered. **If the age is greater than or equal to 100** output "You have lived a century" **otherwise if age is greater than or equal to 55** output "AARP here I come" **otherwise** output "You are still a spring chicken." List the necessary variables for this program. Perform a desk check with the following input values: 100, 21, 555, 0.

3. Draw the flowchart for a program that will obtain exam scores from the user as long as the score is greater than or equal to zero. Determine whether the score is passing (60 or above) or failing and output an appropriate message. List the necessary variables for this program. Perform a desk check with the following input values: 60, 45, 75, 0, and -999.

4. Draw the flowchart for a program that will dispense money from an ATM machine. Continue to request a withdrawal amount from the user until they enter -999. If the amount entered is not evenly divisible by twenty, output the message "You must enter multiples of twenty only" otherwise calculate and output the number of twenty dollar bills you will be dispensing. List the necessary variables for this program. Perform a desk check with the following input values: 35, 60, 61, 360, and -999.

TURN IN (IN THIS ORDER & STAPLED IN THE UPPER LEFT CORNER)

- this sheet
- flowcharts (DONE ON THE COMPUTER ONLY)