|  |  |
| --- | --- |
| **CS118 Programming Fundamentals** | **LAB 01**   1. General Problem-Solving Concepts 2. Beginning Problem-Solving Concepts for the Computer |
| **NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES** | |

# Overview:

# Problem solving in everyday Life

# Types of problems

# Problem solving with computers

# Difficulties with problem solving

# Constants and Variables

# Data Types

# How the Computer Stores Data

# Decision

# Functions

# Operators

# Expressions and EquationsBullet, etc

# Learning Objectives:

# Describe difference between heuristic and algorithmic solutions

# List and describe six problem solving steps for algorithms solutions

# Use problem solving steps to solve problem

# Differentiate between variables, constants

# Differentiate between character, numeric, and logical data types.

# Identify operators, operands, resultants.

# Identify, use functions

# Identify, use operators according to placement in hierarchy chart.

# Set up, evaluate expressions and equations using variables, constants, operators, and the hierarchy of operations.

# Problem Solving in Everyday Life

# Identify the problem

# Understand the problem

# Identify alternative ways to solve the problem

# Select the best way to solve the problem from the list of alternative solutions.

# List instructions that enable you to solve the problem using the selected solution.

# Evaluate the solution.

# Note: If any of these six steps are not completed well, the results may be less than desired.

# Types of Problem

# Algorithmic Solutions

# Problems do not always have straightforward solutions. Algorithmic solutions are reached in a series of steps.

# Examples:

# Balancing a cheque book

# Baking a cake

# Withdrawing money from ATM machine

# Paying for your parking ticket via auto pay machine

# What else?

# Heuristic Solutions

# Problem which couldn’t be solved through a direct set of steps. Require knowledge, experience & a process of trial and error (repeating six steps more than once). Heuristic solutions are attained through trial and error.

# Example:

# How to buy best stock from market

# Should the company be expanded

# Baking a delicious cake

# Raising up a kid

# What else?

# Combination of both

# Most problems require a combination of algorithmic and heuristic solutions

# Example:

# Repairing a car

# Driving a car

# To win in a computer game

# What else??

# Note: Algorithmic solutions are easier to define for computer use than heuristic ones.

# Problem Solving with Computers

# Solution: instructions followed to produce best result

# Result: outcome, computer-assisted answer

# Program: instructions for solution using computer language

# Computers are built to deal with algorithmic solutions, which are often difficult or very time consuming for humans. People are better than computers at developing heuristic solutions. Solving a complicated calculus problem or alphabetizing 10,000 names is an easy task for the computer, but the problem of how to throw a ball or how to speak English is not. The difficulty lies in the programming. How can problems such as how to throw a ball or speak English be solved in a set of steps that the computer can understand?

# The field of computers that deals with heuristic types of problems is called artificial intelligence. Artificial intelligence enables a computer to do things like build its own knowledge bank and speak in a human language. As a result, the computer’s problem-solving abilities are similar to those of a human being. Artificial intelligence is an expanding computer field, especially with the increased use of Robotics.

# Until computers can be built to think like humans, people will process most heuristic solutions and computers will process many algorithmic solutions. Therefore, this book will deal only with algorithmic solutions. Heuristic problem solving can help determine alternative solutions. However, for computer use, they must be transformed into an algorithmic format.

# Difficulties with Problem Solving

# Lack of problem solving experience

# Inadequate solution steps

# Incorrect problem definition

# Alternatives chosen incorrectly

# Invalid logic

# Incorrect solution evaluation

# People have many problems with problem solving. Some have not been taught how to solve problems. Others are afraid to make a decision for fear it will be the wrong one. Often, when people go through the problem-solving process, they complete one or more of the steps inadequately. They may not define the problem correctly or may not generate a sufficient list of alternatives. When choosing the best alternative, they may eliminate good alternatives or list the pros and cons too hastily. They may not use a logical sequence of steps in their solution, or they may focus on details before the framework for the solution is in place. Finally, they may incorrectly or haphazardly evaluate the solution.

# The problem-solving process is not easy. It takes practice and time to perfect, but in the long run the process proves to be of great benefit.

# When solving problems on the computer, one of the most difficult tasks for the problem solver is writing the instructions. Take the task of deciding which number is the largest from a group of three numbers. Almost anyone can immediately tell which is the largest, but many cannot explain the steps they followed to arrive at it. Most General Problem-Solving Concepts 7 people will say, “I can’t explain how I know, I just know it!” This explanation is not good enough for the computer. The computer is a tool that will perform only tasks that the user can explain.

# The computer has a specific system of communication that programmers and users must learn. This system demands that no step in the solution to a problem be left unstated and that all steps be in the proper order. You must assume the computer knows nothing except what you tell it and think of it as an ignorant but efficient aid to problem solving.

# What Problem Can Be Solved By Computer

# When the solution can be produced by a set of step-by-step procedures or actions.

# This step-by-step action is called an algorithm.

# The algorithm will process some inputs and produced output.

# Solving problem by computer undergo two phases:

# Phase 1:

# Organizing the problem or pre-programming phase.

# Phase 2:

# Programming phase.

# PRE-PROGRAMMING PHASE

# This phase requires five steps:

# Analyzing the problem.

# Developing the Hierarchy Input Process Output (HIPO) chart or Interactivity Chart (IC).

# Developing the Input-Process-Output (IPO) Chart.

# Drawing the Program flowcharts.

# Writing the algorithms

# Analyzing The Problem

# Understand and analyze the problem to determine whether it can be solved by a computer.

# Analyze the requirements of the problem.

# Identify the following:

# Data requirement.

# Processing requirement or procedures that will be needed to solve the problem.

# The output.

# All These requirements can be presented in a Problem Analysis Chart (PAC)

|  |  |  |
| --- | --- | --- |
| Data | Processing | Output |
| Given in the problem or provided by the user. | List of processing required or procedures. | Output requirement. |

# Example # 01: Payroll Problem

# Calculate the salary of an employee who works by hourly basis. The formula to be used is

# Salary = Hour works \* Pay rate

|  |  |  |
| --- | --- | --- |
| Data | Processing | Output |
| Hours work,Pay rate | Salary = Hours work \* payrate | Salary |

# Example # 02:

# Write a Problem Analysis Chart (PAC) to find an area of a circle where

# area = pi \* radius \* radius

|  |  |  |
| --- | --- | --- |
| Data | Processing | Output |
| radius | area = 3.14 x radius x radius | area |

# Example # 03:

# Write a problem analysis chart (PAC) that asks a user to enter the distance of a trip in miles, the miles per gallon estimate for the user’s car, and the average cost of a gallon of gas. Calculate and display the number of gallons of gas needed and the estimated cost of the trip.

|  |  |  |
| --- | --- | --- |
| Data | Processing | Output |
| Distance,miles per gallon,cost per gallon | gas needed = distance / miles per gallon.estimated cost = cost per gallon x gas needed | Display gas neededDisplay estimated cost |

# Developing the Hierarchy Input Process Output (HIPO) or Interactivity Chart

# The problem is normally big and complex.

# Thus, requires big program.

# Thus, the processing can be divided into subtasks called modules.

# Each module accomplishes one function.

# These modules are connected to each other to show the interaction of processing between the modules.

# Main/control module controls the flow all other modules.

# The IC is developed using top-down-method: top to down left to right order (also refer to order of processing).

# Modules are numbered, marked for duplication, repetition or decision.

# The interaction will form a hierarchy, called Hierarchy Input Process Output Chart (HIPO) or Interactivity Chart (IC). Programming which use this approach (problem is divided into subtasks) is called *Structured Programming*.

# 

# 

# HIPO of payroll problem.

# Example # 04: Extended Payroll Problem

# You are required to write a program to calculate both the gross pay and the net pay of every employee of your company. To determine the gross pay, you have to multiply the accumulated total hours worked by the employee, by the appropriate pay rate. The program should print the cheque that tells the total net pay. The net pay is calculated by subtracting the gross pay with any deductions that may be incurred by the employee.

# 

# Example # 05:

# Write a Hierarchy Input Process Output (HIPO) to find an area of a circle where

# area = pi \* radius \* radius

# 

# Example # 06:

# Write a Hierarchy Input Process Output (HIPO) that asks a user to enter the distance of a trip in miles, the miles per gallon estimate for the user’s car, and the average cost of a gallon of gas. Calculate and display the number of gallons of gas needed and the estimated cost of the trip.

# 

# Developing the Input Process Output (IPO) Chart

# Extends and organizes the information in the Problem Analysis Chart.

# It shows in more detail what data items are inputs, what is the processing or modules on that data, and what will be the result or output.

# It combines information from PAC and HIPO Chart.

# Example # 07: Payroll Problem

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Processing | Module | Output |
| -Hours Worked-Pay Rate-Deduction | -Enter Hourly Worked-Enter Pay Rate-Calculate Gross Pay-Enter Deductions-Calculate Net Pay-Print Cheque-End | 1100120010002100200030000000 | -Net pay |

# Example # 08:

# Write a Input Process Output (IPO) to find an area of a circle where

# area = pi \* radius \* radius

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Processing | Module | Output |
| - radius | - Enter radius- area = 3.14 x radius x radius- Display area- end | 1000200030000000 | - Area of a circle |

# Example # 09:

|  |  |  |  |
| --- | --- | --- | --- |
| Data | Processing | Module | Output |
| Distance in milesMiles per gallonCost gas per gallon | Enter distanceEnter miles per gallonCalculate total gas neededEnter cost gas per gallonCalculate estimated costDisplay total gas and estimated costEnd | 1100120010002100200030000000 | Total gas neededEstimated cost |

# Drawing the Program Flowcharts

# Flowchart is the graphic representations of the individual steps or actions to implement a particular module.

# The flowchart can be likened to the blueprint of a building. An architect draws a blueprint before beginning construction on a building, so the programmer draws a flowchart before writing a program.

# Flowchart is independent of any programming language.

# Flowchart is the logical design of a program.

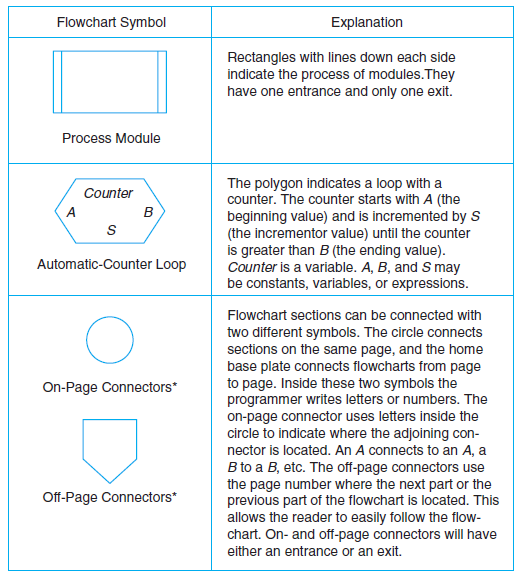
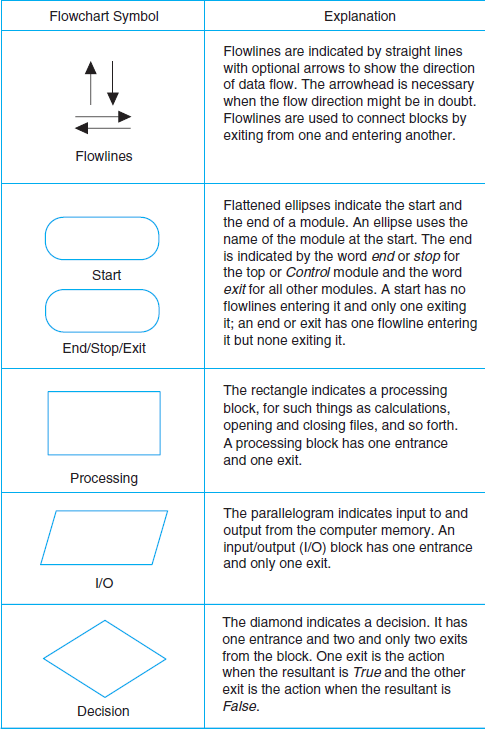
# It is the basis from which the actual program code is developed.

# Flowchart serves as documentation for computer program.

# The flowchart must be drawn according to definite rules and utilizes standard symbols adopted internationally.

# The International Organization for Standardization (IOS) was the symbols shown below (You can draw the symbols using ready-made flowcharting template):

## FLOWCHART SYMBOLS



## Example # 10: Sale Problem

## Draw a flowchart for a problem that to read two numbers. The first number represents the unit price of a product and the second number represents the quantity of the product sold. Calculate and print the total sale.

# *Solution:*

# Stepwise Analysis of the Sale Problem

# Start of processing

# Read the unit price

# Read the quantity

# Calculate total sale

# Print total sale

# Stop the processing

# *Flowchart:*

READ UNIT PRICE

READ UNIT PRICE

TOTAL SALE =

UNITPRICE \* QUANTITY

PRINT TOTAL SALE

# Example # 11: Finding Average Problem

## Read a sequence of number, find the average of the number and print the average.

## *Solution:*

# Stepwise Analysis of Average Problem

# Start the processing

# Read a number

# Add the number

# Repeat reading until last data

# Calculate the average

# Print the average

# Stop the processing

# *Flowchart:*

# 

# 

# 

# Writing the Algorithm (Pseudocode)

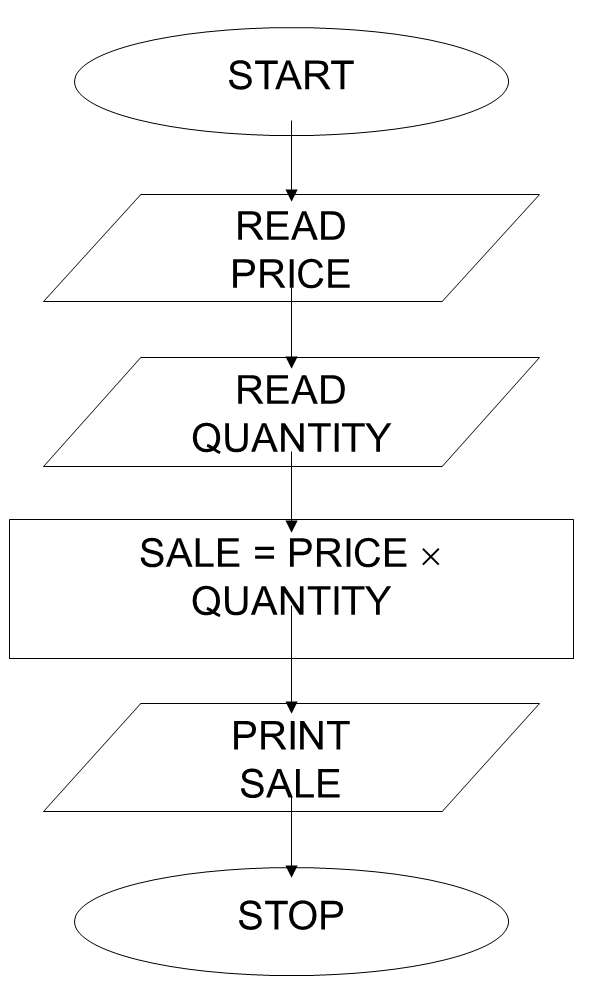
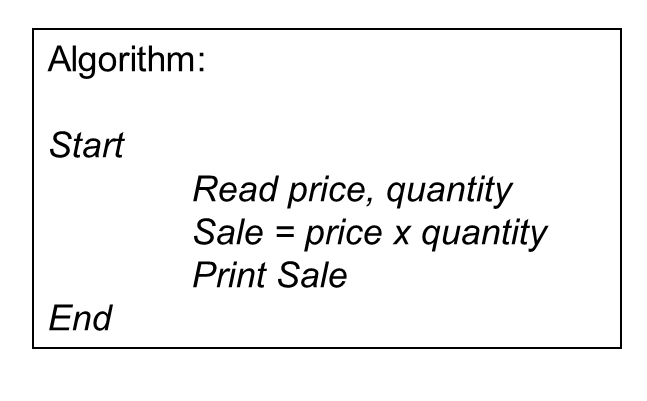
# Pseudocode means an imitation computer code.

# It is used in place of symbols or a flowchart to describe the logic of a program. Thus, it is a set of instructions (descriptive form) to describe the logic of a program.

# Pseudocode is close to the actual programming language.

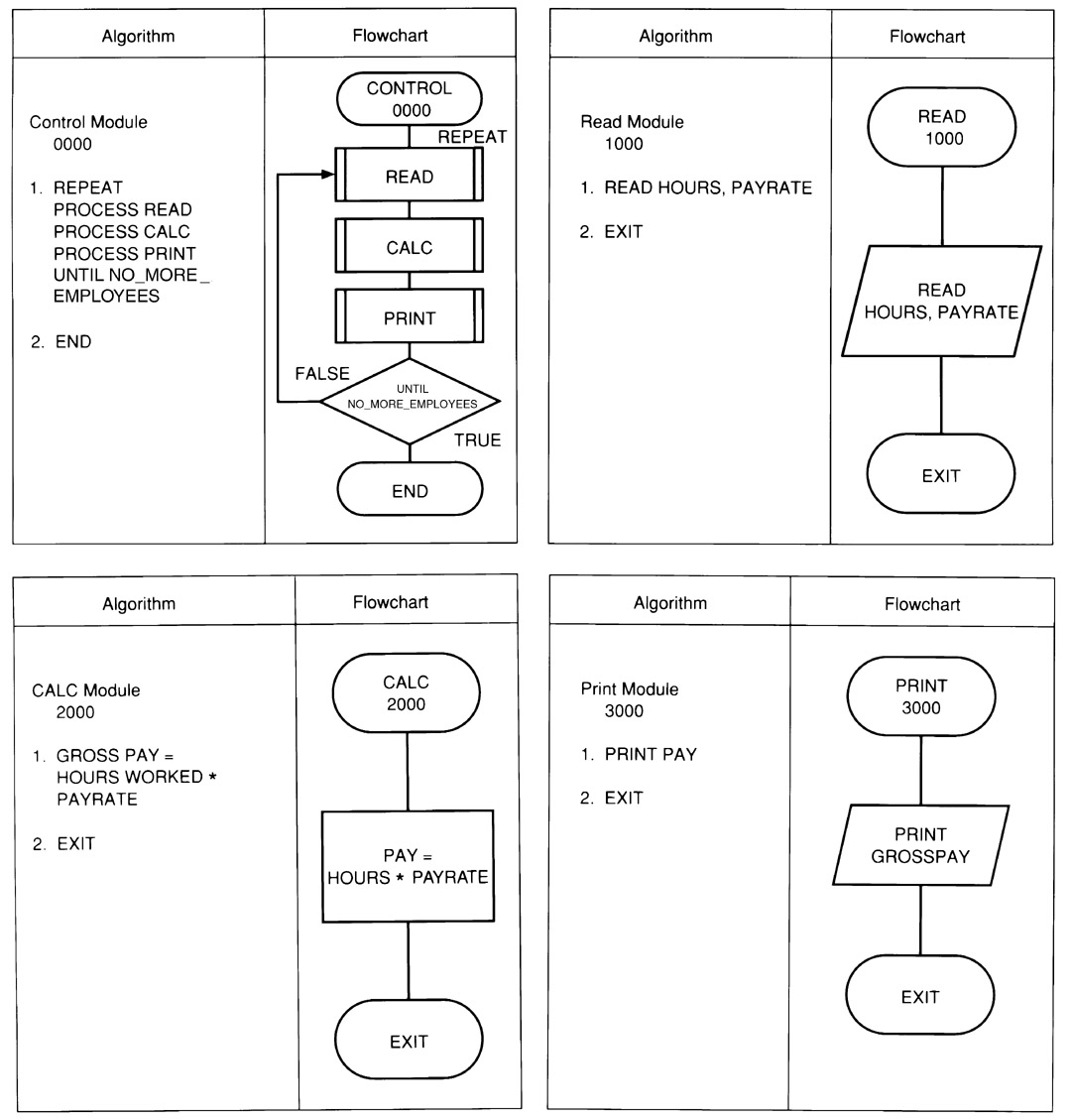
# Using the Pseudocode, the programmer can start to write the actual code.

## Example # 12:

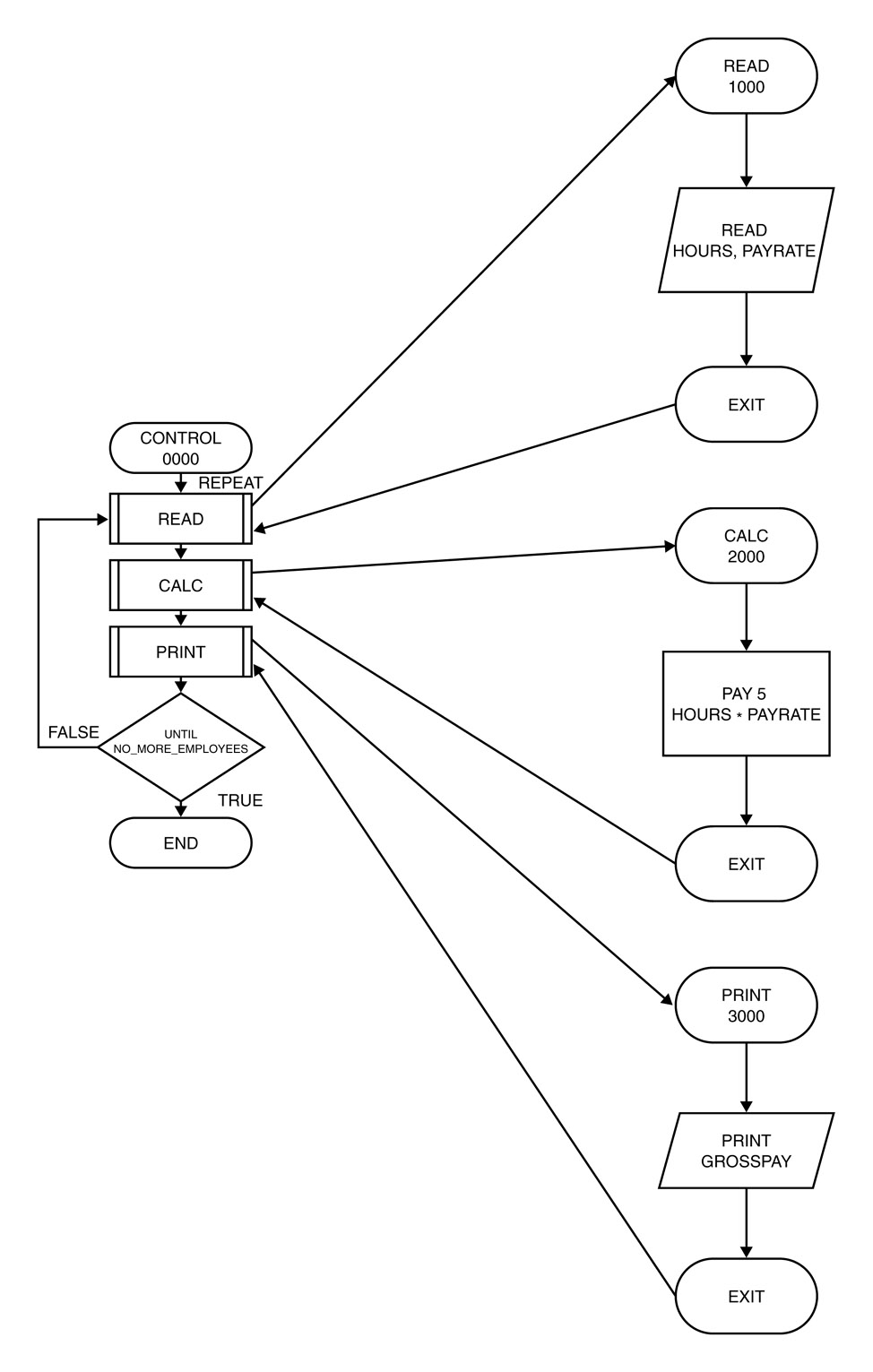
**** ****

**Example # 13:**

## Payroll system(Flowchart & Algorithm)



**Order of Execution of Instructions : Payroll System**



***PROBLEM***

To find the smallest of two numbers.

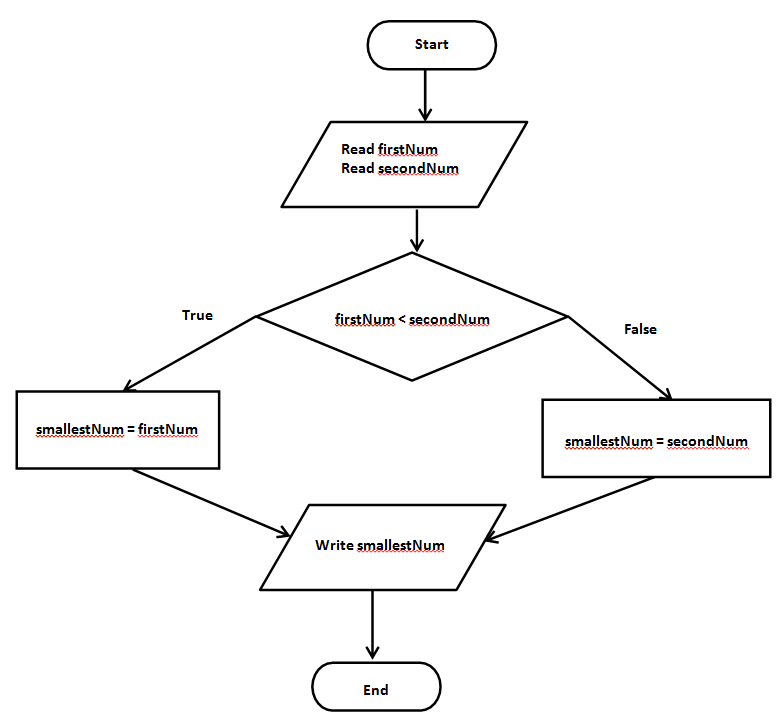
**STEP1: ANALYZING THE PROBLEM**

|  |  |
| --- | --- |
| GIVEN DATA | REQUIRED RESULTS |
| firstNumber secondNumber | smallestNumber |
| PROCESSING REQUIRED | SOLUTION ALTERNATIVES |
| Compare first number with second number FirstNumber < SecondNumber | 1. Define the two numbers as constants. 2. Define the two numbers as input values. |

**STEP2: WRITING THE ALGORITHM**

1. Start
2. Read firstNumber and secondNumber.
3. If firstNumber < secondNumber then smallestNumber = firstNumber Else smallestNumber = secondNumber
4. Write smallestNumber.
5. Exit

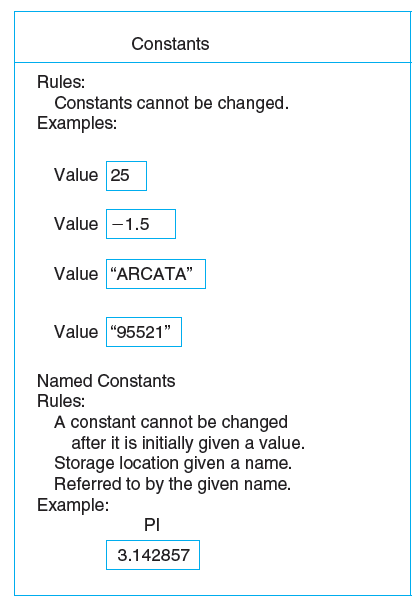
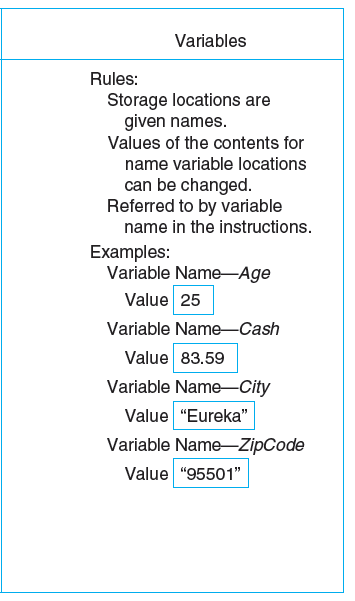
**STEP3: DRAWING THE FLOWCHART**



# Important Concepts to Learn

# 

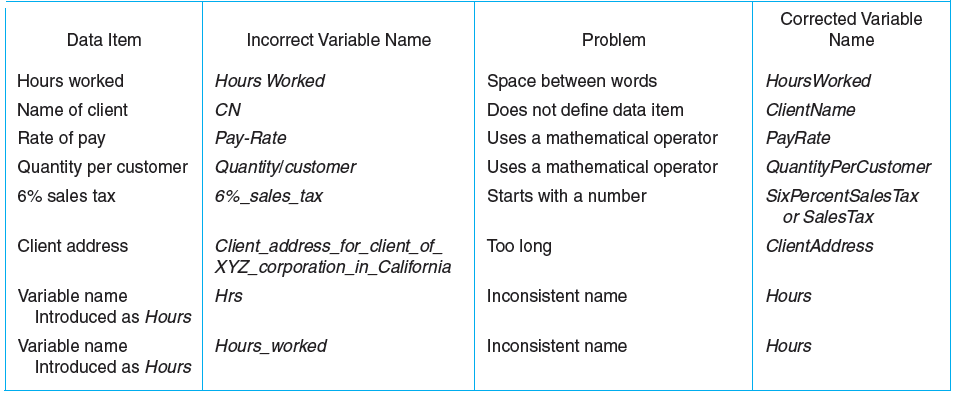
# Constants and Variables on the Computer

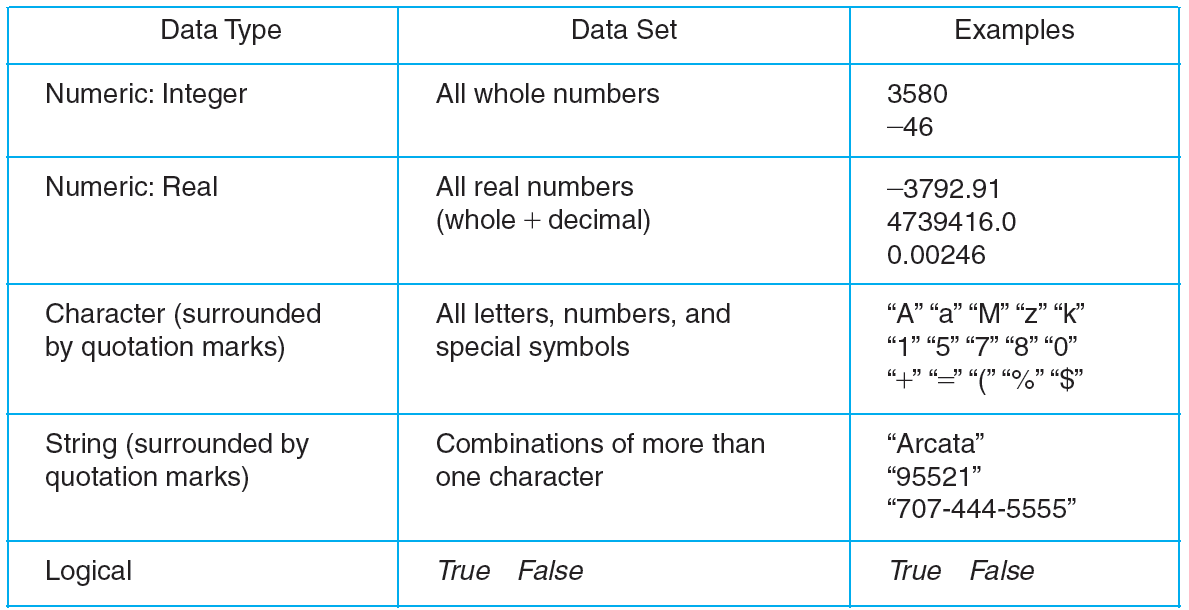
**Rules for Naming and Using Variables**

1. Name a variable according to what it represents.
2. Do not use spaces.
3. Start a variable name with a letter.
4. Do not use a dash or any other symbol that is used as a mathematical operator.
5. Consistent usage of variable name.
6. Consistent use of upper, lowercase characters in variable names
7. Use naming convention specified by your company

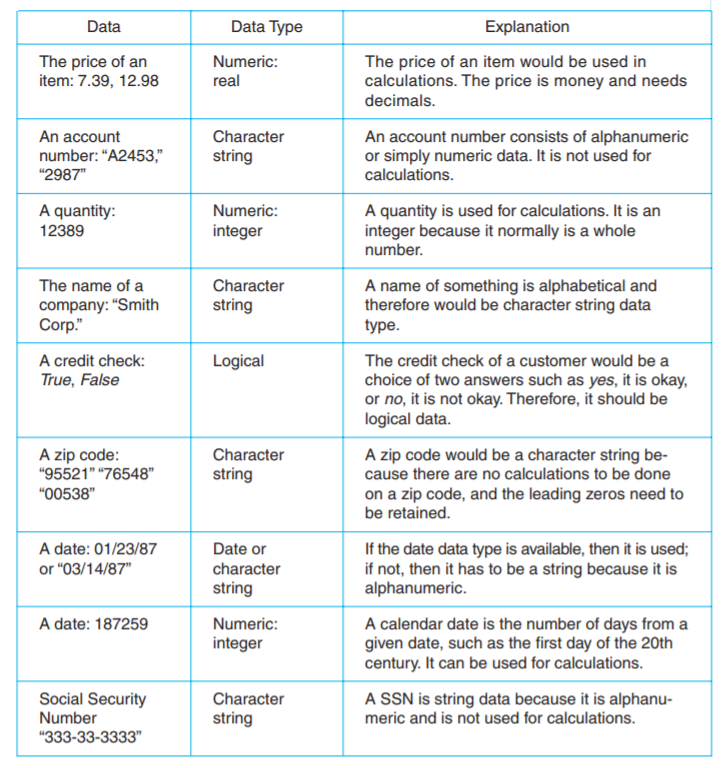
**Incorrect Variable Names**



**Data Types and Their Data Sets**



**Examples of Data Types**

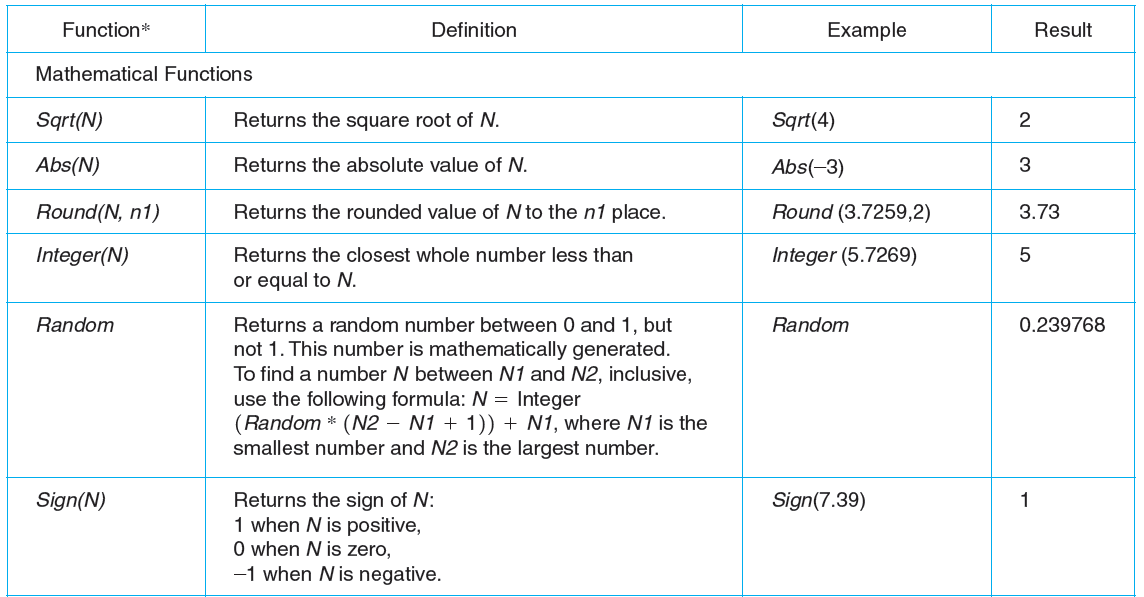


**Functions**

Functions have been divided into classes.

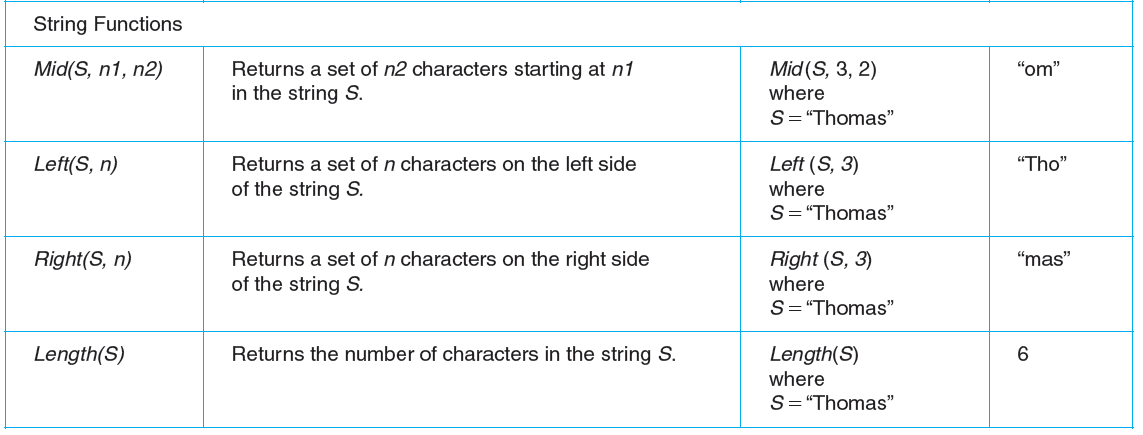
1. **Mathematical functions.**

Often used in science and business, mathematical functions calculate such things as square root, absolute value, or a random number. Other mathematical functions used primarily for scientific purposes have not been included in the table.



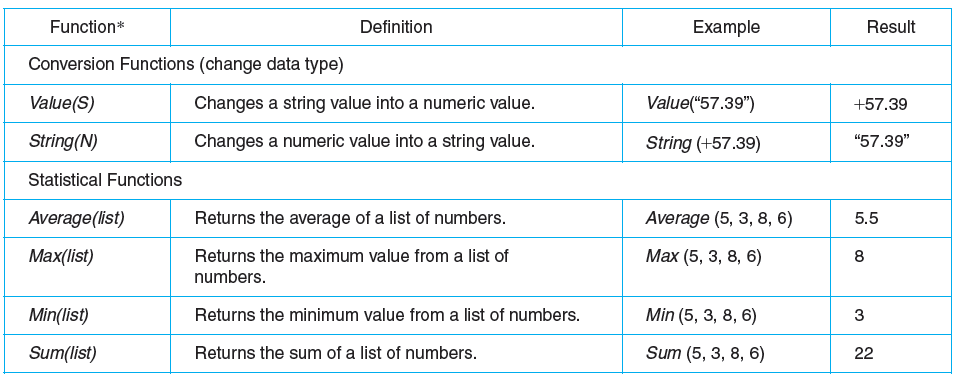
1. **String functions.**

These are used to manipulate string variables. For example, they copy part of the string into another variable, find the length or the number of characters in the string, and so forth.



1. **Conversion Function.**

These functions are used to convert data from one data type to another. For example, since character strings cannot be used in calculations, one of these functions would convert a string value to a numeric value.

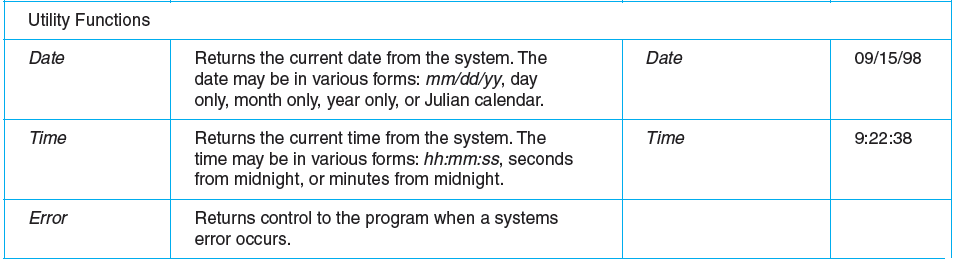


1. **Statistical functions.**

These functions are used to calculate things such as maximum values, minimum values, and so forth.

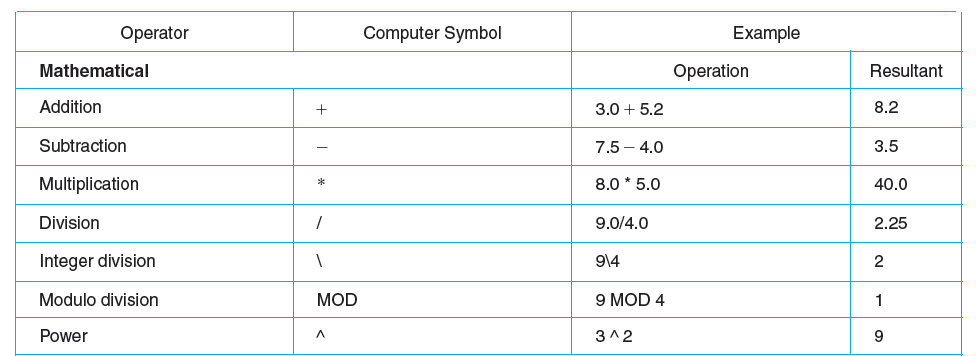
1. **Utility Function.**

This class is very important in business programming because most reports require some use of utility functions. They access information outside the program and the language in the computer system. Examples of these include date and time functions.

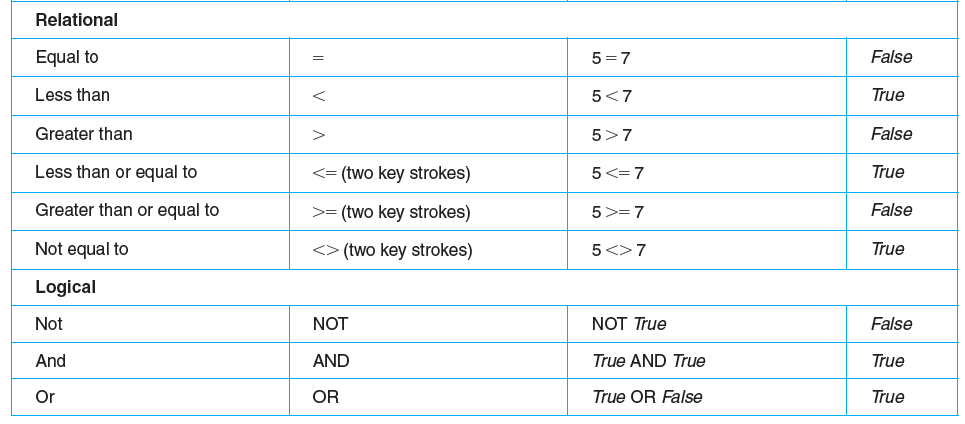


**Operators and Their Computer Symbols**

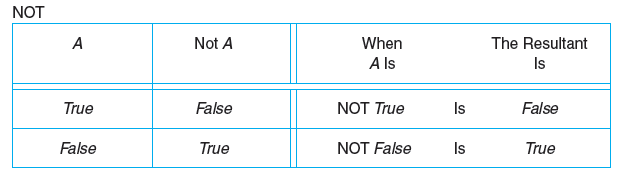
**Mathematical Operators:**

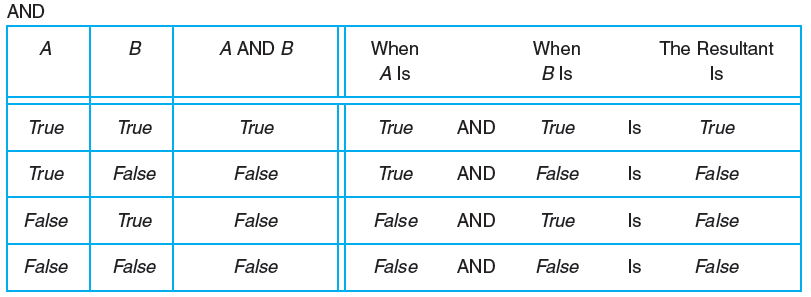


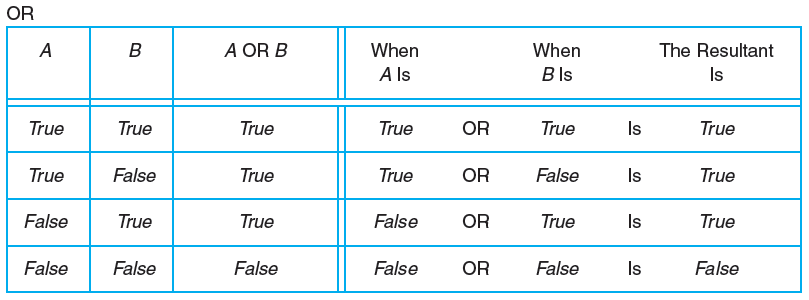
**Relational Operators:**



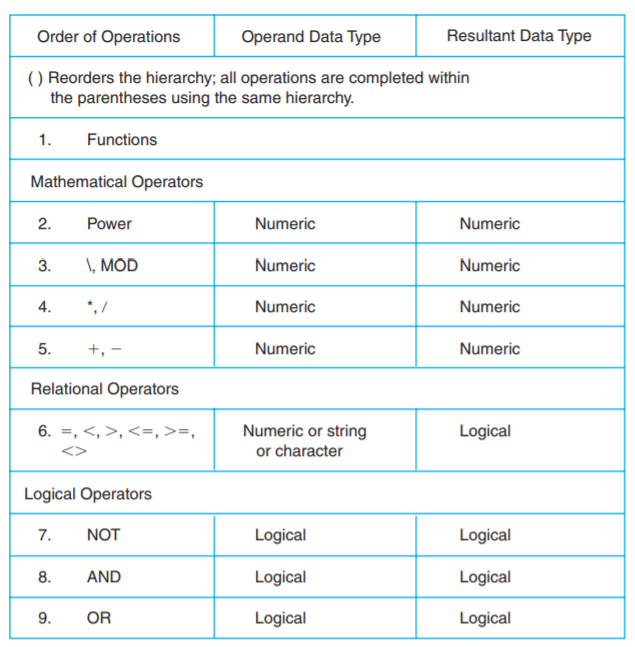
**Logical Operators:**



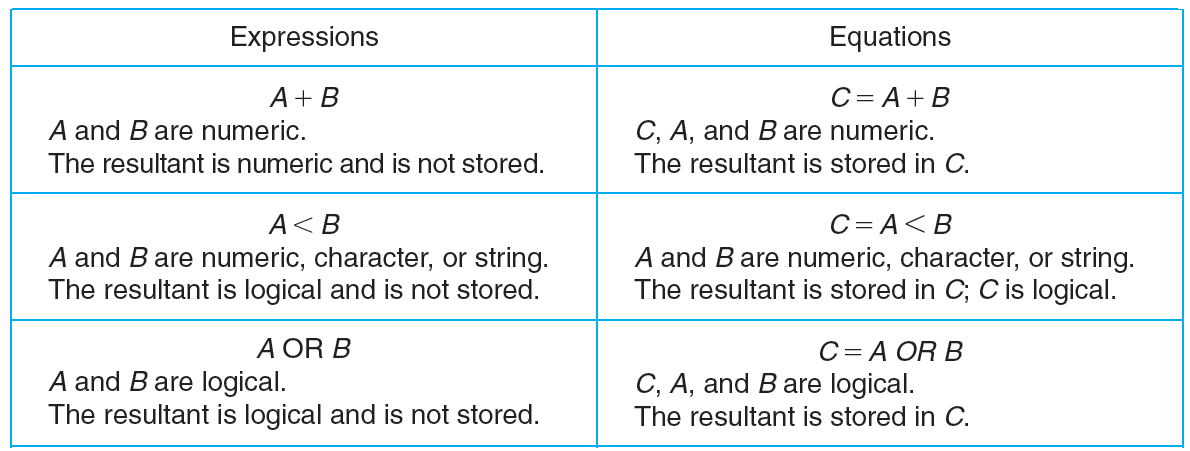




**Hierarchy of Operations**



**Expressions and Equations**



**Evaluating a Mathematical Expression**

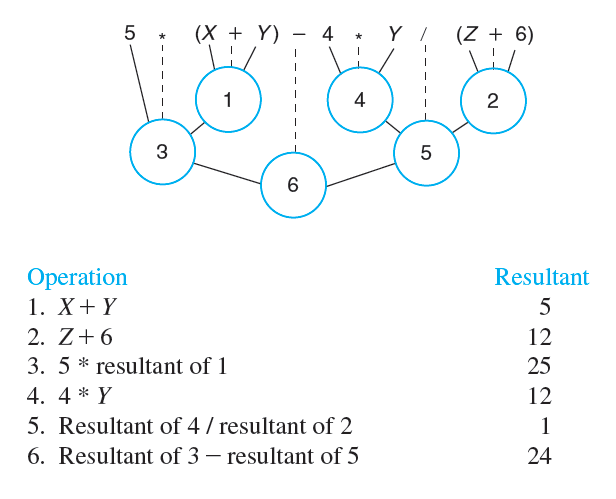
**Example:**

To find out if proposed solutions are correct, it is important for the programmer to evaluate, or test, all expressions and equations. Assume the programmer has written the expression

**5 \* (X + Y) - 4 \* Y/(Z + 6)**

The programmer uses the following values to evaluate the expression:

**X = 2, Y = 3, Z = 6**



**Example:**

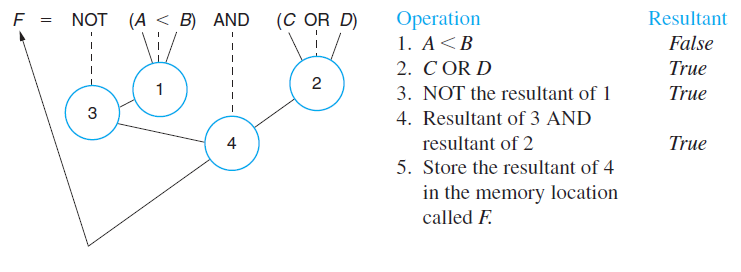
Assume the programmer has written the following equation:

**F = NOT(A < B) AND (C OR D)**

The programmer uses the following values to evaluate the equation:

**A = 4, B = 2, C = True, D = False**

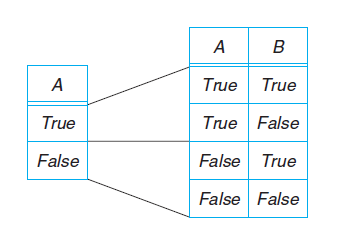
The operations are completed in hierarchical order from left to right, as illustrated in the



**Developing a Table of All Possible Resultants of a Logical Expression**

**Example:**

* Two unknowns—*A* and *B.*
* Four combinations: *B* can be either *True* or *False* for each value of *A..*

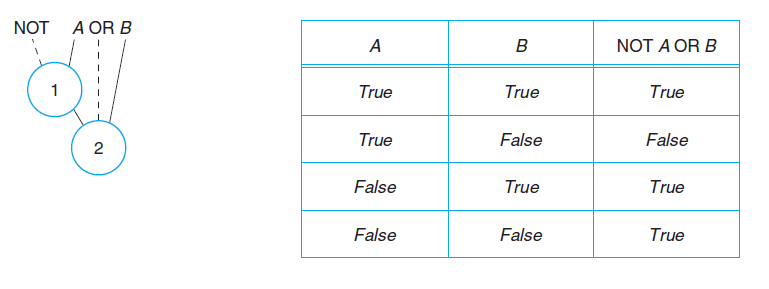


**Developing a Table of All Possible Resultants of a Logical Expression**

**Example:**

**Given the expression**

**NOT A OR B**



**Decision:**

Following are the main types of decision statements.

|  |  |  |
| --- | --- | --- |
| **Statement** | **Description** | **C - Syntax** |
| [if statement](http://www.tutorialspoint.com/cplusplus/cpp_if_statement.htm) | An if statement consists of a boolean expression followed by one or more statements. | if (expression)  {  Statement;  } |
| [if...else](http://www.tutorialspoint.com/cplusplus/cpp_if_else_statement.htm) [statement](http://www.tutorialspoint.com/cplusplus/cpp_if_else_statement.htm) | An if statement can be followed by an optional else statement, which executes when the boolean expression is false. | if (expression)  {  statement1;  }  else  {  statement2;  } |
| [if](http://www.tutorialspoint.com/cplusplus/cpp_nested_switch.htm)-else-if | This construct is useful where two or more alternatives are available for selection. | If (condition)  {  statement 1;  }  else if (condition)  {  statement 2;  }  else  {  statements n ;  } |
| [nested if](http://www.tutorialspoint.com/cplusplus/cpp_nested_if.htm) [statements](http://www.tutorialspoint.com/cplusplus/cpp_nested_if.htm) | An entire *if...else* construct is written within either the body of the if statement or the body of an else statement. | if(condition\_1)  {  If (condition\_2)  {  block statement\_1;  }  else  {  block statement\_2;  }  }  else  {  block statement\_3;  }  block statement\_4; |

**LAB#01 EXERCISES**

**Question # 01:**

The marks obtained by a student in 5 different subjects are input through the keyboard.

The student

gets a division as per the following rules: Percentage above or equal to 60 First division Percentage between 50 and 59 Second division Percentage between 40 and 49 Third division Percentage less than 40 Fail

Write a program to solve the problem.

**Question # 02:**

An admission charge for The Little Rep Theater varies according to the age of the person. Develop a solution to print the ticket charge given the age of the person. The charges are as follows:

|  |  |
| --- | --- |
| a. Over 55: | PKR 10.00 |
| b. 21–54: | PKR 15.00 |
| c. 13–20: | PKR 10.00 |
| d. 3–12: | PKR 5.00 |
| e. Under 3: | Free |

**Question # 03:**

Write a program to check data using the following criteria. The data are assumed correct when

1. Number is less than 1000
2. Number is positive.
3. Number is divisible by 2

**Question # 04:**

A hotel has a pricing policy as follows:

1. 2 people: $85
2. 3 people: $90
3. 4 people: $95
4. Additional people: $6 per person

If the customer is staying on company business, there is a 20% discount. If the customer is over 60

years of age, there is a 15% discount. A customer does not receive both discounts.

Given the above data, print the cost of the room.

**Question # 05:**

Write if statements to do the following:

1. If character variable taxCode is ’T’, increase price by adding the taxRate percentage of price to it.
2. If integer variable opCode has the value 1, read values for X and Y and calculate and print their sum.
3. If integer variable currentNumber is odd, change its value so that it is now 3 times currentNumber plus 1 otherwise change its value so that it is now half of currentNumber.
4. Assign true or 1 to the boolean variable leapYear if the integer variable year is a leap year. (A leap year is a multiple of 4, and if it is a multiple of 100, it must also be a multiple of 400.)

**Question # 06:**

Name the data type for each of the following constants. Explain your answer.

1. 5.38
2. “87654”
3. True
4. “A”
5. “707-434-5555”
6. “New York”
7. -389
8. 2.45E6
9. 48976.0
10. False

**Question # 07:**

Find the result of the following operations:

1. 5 + 4
2. 10/2
3. True OR False
4. 20 MOD 3
5. 5 < 8
6. 25 MOD 70
7. “A” > “H”
8. NOT True
9. 25/70
10. False AND True
11. 20 \* 0.5
12. 35 <= 35
13. 35/7
14. False OR False
15. True AND True
16. 50 MOD 5
17. -35 < 67
18. 4.0 ^ 3
19. 60\9
20. 35 < 35
21. True AND False

**Question # 08:**

Evaluate the following equations, given the values A = 12, B = 3, C = 6, D = 2:

1. F=A + B/C – D ^ 2
2. F=(A + B)/C – D ^2
3. F = A + B/(C – D ^2)
4. F=(A + B) MOD C
5. F=(A + B)/ D ^2

**Question # 09:**

Write the following equations in computer form:

1. X = Y + 3Z – (Z + Y)/ (Z – 3)
2. X = 5Y + (3Z – 1)/4(3Z + 1) – Y
3. X = (X – Y)2

**Question # 10:**

Create a table that gives all possible answers for the following logical equations. (Include the structure of the order of processing—see page 33 for example.) Make clear how you set up the table.

1. R = A OR B
2. R = NOT A OR B
3. R = A AND B AND (B OR C)
4. R = NOT (A OR B) AND NOT (B OR C)
5. R = B AND NOT (A OR C) OR NOT (B AND C)

**Question # 11:**

Evaluate the following for the values A = 5, B = 2, C = True, D = False. (Include the structure of the order of processing.)

1. R = A + 3 > B - 1 AND C OR D
2. R = NOT C OR D OR A - 3 <= B

**Question # 12:**

What is wrong with these variable names? Can you correct them?

1. ***City Name*** referencing the name of a city.
2. ***Client-name*** referencing a client name.
3. ***City/State*** referencing a city and state.
4. ***LN*** referencing a last name.
5. ***Street address***
6. ***Street\_Address\_for\_Joe’s\_Hardware\_Supply\_Incorporated\_Client***