<u>CS 111: Introduction to Computers and Programming LAB</u>

I Sem, BS. (Common)

(2015)

<u>Prepared by</u> <u>Approved by</u>

(NARENDRA V G) (H.O.D)

DEPT OF COMPUTER SCIENCE & ENGG.

M. I. T., MANIPAL

INSTRUCTIONS TO STUDENTS:

1. Students should be regular and come prepared for the lab practice.

2. In case a student misses a class, it is his/her responsibility to complete that missed experiment(s).

3. Students should bring the observation book, lab journal and lab manual. Prescribed textbook and class notes can be kept ready for reference if required.

4. They should implement the given experiment individually.

5. While conducting the experiments students should see that their programs would meet the following criteria:

   - Programs should be interactive with appropriate prompt messages, error messages if any, and descriptive messages for outputs.
   - Programs should perform input validation (Data type, range error, etc.) and give appropriate error messages and suggest corrective actions.
   - Comments should be used to give the statement of the problem and every function should indicate the purpose of the function, inputs and outputs.
   - Statements within the program should be properly indented.
   - Use meaningful names for variables and functions.
   - Make use of Constants and type definitions wherever needed.

6. Once the experiment(s) get executed, they should show the program and results to the instructors and copy the same in their observation book.

7. Questions for lab tests and exam need not necessarily be limited to the questions in the manual, but could involve some variations and/or combinations of the questions.

CONTENTS

Lab 1: RAPTOR TOOL

Draw flowcharts for each of the following using RAPTOR tool:

Simple Flowcharts

1. Perform the four basic arithmetic Operations
   [e.g. sum=a+b; diff=a-b; prod=a*b; quot=a/b]

2. Evaluate the area of the circle [Area = Pi * R^2]

3. Convert the time in seconds to hours, minutes and seconds [1 hr =3600 sec]

4. Find the sum of the digits of a 3-digit number [ex Number = 123, sum=6][no looping]

5. Convert temperature given in Fahrenheit to Centigrade [C=5/9(F-32)]

6. Convert distance in mm to cm, inch, feet.
   [1 cm =10mm, 1inch=2.5cm, 1 feet =12 inches]

Complex Flowcharts

1. Check whether given number is Even or Odd

2. Find the largest among 3 numbers

3. Calculate the grade for the marks entered.

4. Find sum of natural numbers upto N.

5. Sum of digits of a given number

6. Factorial of a given number

Lab 2: SIMPLE PROGRAMS USING C++

Write C++ programs to do the following:

1. To input P, N and R and Compute simple and compound interest.
   [Hint: SI = PNR/100, CI = $P(1+R/100)^T$-P]

2. To input radius and find the volume and surface area of a sphere.
   [Hint: volume = $(4\pi r^3)/3$, Area=$4\pi r^2$]

3. To convert the temperature given in Fahrenheit to Centigrade. [Hint: C=5/9(F-32)]

4. To convert the time in seconds to hours, minutes and seconds. [Hint: 1 hr =3600 sec]

5. To find the sum of the digits of a four-digit number [Eg: N = 1234, then sum=10]

Lab 3: DECISION MAKING AND BRANCHING CONTROL STRUCTURES

Write C++ programs to do the following:

1. To check whether the given number is odd or even using *if-else* statement.

2. To find the largest among given 3 numbers using *if* statement

3. To compute all the roots of a quadratic equation using switch statement.
   [Hint : x = (-b +/- sqrt(b^2-4ac))/2a]
4. To check whether the given number is zero, positive or negative using *else-if* ladder.

5. To find the smallest among three numbers using conditional operator.

Lab 4: LOOPING CONTROL STRUCTURES

1. To find the sum of natural numbers upto N.

2. To print all even numbers upto a given limit N.

3. To reverse a given number [It shouldn't be the print effect]
   [Ex: 1234, reverse=$4*10^3 + 3 * 10^2 + 2 * 10^1 + 1 * 10^0$ =4321]

4. To convert binary number to decimal.
   Ex: 1101 = $1*2^3 + 1 * 2^2 + 0 * 2^1 + 1* 2^0$ =13

5. To generate the Fibonacci series up to the given limit.

6. To generate the multiplication table for n numbers up to k terms (using nested loops).

```
[ Hint :                  1  2  3   4  5  ….  K

                         2  4  6  8  10  …..2*k
                        ..………………..…
                         n…………………nK ]
```

Lab 5: DECISION MAKING AND LOOPING CONTROL STRUCTURES

1. To generate the prime numbers between given 2 limits.

2. To check all the numbers from 1 to N and display only those numbers for which the sum of the cubes of all the digits equals the number itself (Armstrong Number).

3. To check whether a given number is perfect number or not.
   [Hint: Sum of all positive divisors of given number excluding the given number is equal to the number]
   Ex: 28 = 1+ 2 + 4 + 7 + 14 = 28 is a perfect number

4. Check whether a given number is palindrome or not.

Lab 6: 1-D ARRAYS

Write C++ programs to do the following:

1. Find the largest and smallest element in an array.

2. To insert an element into an array and to delete an element from an array.

3. To print all the prime numbers in a given array.

4. To arrange the array elements in ascending/descending order using Selection/Bubble sort.

5. To insert an element into a sorted array (after insertion remains sorted)

6. To search for a given number in an array using Binary Search method

7. To delete all the duplicate elements of an array.
   [Ex: input array [3, 4, 1, 3, 5, 5, 3] ; output array [3, 4, 1, 5]

Lab 7: 2-D ARRAYS

Write C++ programs to do the following:

1. To read 2 matrices and add and subtract them. Display all the matrices.

2. To read 2 matrices and multiply them. Display all the matrices.

3. To compute the row sum and column sum of a given matrix.

4. To check whether the given matrix is magic square or not.

5. To check whether the given matrix is a Lower triangular matrix.
   Ex:
   $$\begin{array}{ccc} 1 & 0 & 0 \\ 2 & 3 & 0 \\ 4 & 5 & 6 \end{array}$$
6. To find whether a given matrix is symmetric or not. [Hint: $A = A^T$]

7. To find the trace and norm of a given square matrix.
   [Hint: Trace = sum of principal diagonal elements
   Norm = SQRT (sum of squares of the individual elements of an array)]

Lab 8: STRINGS

Write C++programs without using STRING-HANDLING functions for the following:

1. To count the number of words in a sentence.

2. To input a string and toggle the case of every character in the input string.
   Ex: INPUT: aBcDe
   OUTPUT: AbCdE

3. To search for a given sub string in the main string.

4. To check whether the given string is a palindrome or not.

5. To convert a given string representing a number to an integer.

Lab 9: USER DEFINED FUNCTIONS

Write C++ programs as specified below:

Simple Functions

1. Write a function **Fact** to find the factorial of a given number. Using this function, compute $^{N}C_{R}$ in the main function.

2. Write a function **IsPrime** to check whether the given number is prime or not. Using this function, generate first N prime numbers in the main function.

Functions with Arrays(1D/2D) as Parameter

3. Write a function **Largest** to find the maximum of a given list of numbers. Write a main program to read N numbers and find the largest among them using this function.

4. Write a function **Sort** to sort a list of names which will use a function **compare** to compare two names. (Selection /bubble Sort may be used).

5. Write a function **CornerSum** which takes as parameter a matrix, no. of rows and no. of columns of the matrix and returns the sum of the elements in the four corners of the matrix. Write a main function to test the function.

Lab 10: USER DEFINED FUNCTIONS (C++ Concepts)

Write C++ programs as specified below:

Function Overloading, Default Arguments, Inline Functions

1. Write a function **CalculateSI** to calculate simple interest based on the input parameters *principle*, *rate* and *time*, with *rate* being the default argument having a default value of 10. Write a main function to test the function. Include function calls to **CalculateSI**, one with using the default value and the other with passing a value for *rate.*
2. Overload a function **Area** to compute the area of a square, rectangle and triangle. Write a main function to test the function.
3. Write an inline function **Area** to compute the area of a cube. Write a main function to test the function.

Lab 11: CLASSES & OBJECTS

1. Define a class **Complex** to represent a complex number. Include the following members:

   Data members:

   a. Real

   b. Imaginary

   Member functions:

   a. to assign initial values

   b. to display complex number in a suitable format

   Write a main function to test the class.

2. Define a class **Account** to represent a bank account. Include the following members

   Data members:

   a. Name of the depositor

   b. Account number

   c. Type of account

   d. Balance amount in the account

   Member functions:

   a. to assign initial values

   c. to deposit an amount

   d. to withdraw amount after checking minimum balance

   e. to display name & balance

   Write a main function to test the class.

Lab 12: PROGRAMS ON FILE HANDLING

Write C++ programs as specified below:

1. To open and read a sentence from a file and display the same on the screen.
2. To write a line of text into an existing file.
3. To copy the contents of one file into another file.
4. To print its own source code on the screen.

REFERENCES

1. E. Balaguruswamy, “Object Oriented Programming with C++”, Tata McGraw Hill, 2nd Edition 2007.

2. Herbert Schildt, “C++: The Complete Reference”, Tata McGraw Hill, 4th Edition 2002.

3. Robert Lafore, “Object Oriented Programming with Turbo C++”, Galgotia Publications, 2002.