C Programming Assignment

Operators and operands

1. Develop a C program to perform operations (+,*,-, / and %) on two whole numbers. Identify suitable data types to represent the numbers and resultant values

2. Develop a C program to add two operands and store the result in one of the operand using addition assignment operator.

3. Write a C program to find the maximum of 2 numbers using Conditional operator.

4. Write a menu based C program to perform operations (+, - and *) on matrices.

5. Write a program to perform operations on complex numbers.

6. Write a program to find whether the given processor is little endian or big endian.

Basic Data Types

1. Develop a C program to calculate simple interest using the formula I=PTR/100. Display Interest with two digit precision after decimal point

Control Sequence

1. Develop a C program having following logic. If i is 20 or j is 20, display as “Atleast one variable is having 20” otherwise display “Both variables are not having 20”. If i is less than or equal to 40 and j is less than or equal to 40, It should display “Both are less than or equal to 40” otherwise, it should display as “Both are not less than or equal to 40”. Implement this using if-else statement as well as with conditional operator.

2. Develop a C program which accepts character type data item from user. In case if user typed
‘A’ or ‘a’, it should display A for Apple
‘B’ or ‘b’, it should display B for Bat
‘D’ or ‘d’, it should display D for Dog
‘F’ or ‘f’, it should display F for Fan
Instead of the above 4 characters, if user types any other character, it should display “Character is not in the range”. Implement this using if-else statement and switch statement.

3. Develop a C program which adds all numbers from 1 to N, except those which are divisible by 5. Implement this using for loop and continue statement.

4. Develop a C program to find factorial of a number N using for loop.

5. Develop a C program to find sum of all odd numbers upto N using while loop.

6. Write a program to print ASCII values of upper case and lower case alphabets and digits (A-Z, a-z and 0-9).

7. Write a Program to find if a given number is Armstrong number.
Hint: (153 = 1^3 + 5^3 + 3^3)

8. Write a program to search for an element in a given list of elements. Use break statement.

9. Write a program to print all the prime numbers in the first N numbers.

10. Write a program to find the sum of digits of a given number.

11. Write a C program to generate two Relatively Prime numbers.

12. Write a program to generate Random number

Storage Class Specifiers

1. Develop multi file program to understand static, auto, register, global, static global variables.What is the scope and lifetime of each of these types of variables.

Arrays

1. Write a program to read your name into a character array. Print the name along with the length of your name and sizeof the array in which name is stored.

2. Use scanf function to read a string of characters (into character type array called text) including alphabets, digits, blanks, tabs etc except new line character. Write a loop that will examine each character in a character-type array and determine how many of the characters are letters, how many are digits, how many are blanks and how many are tabs. Assume that text contains 80 characters.

3. Write a program that reads a number that says how many integer numbers are to be stored in an array, creates an array to fit the exact size of the data and then reads in that many numbers into the array.

Functions

1. Write a program to calculate n!/(n-r)! using functions.

2. Write a recursive function to find factorial of a number.

3. Write a function to swap contents of two variables using functions and pointer variables.4.

4. Write a C program with a function rotoate_right (n, b). This function rotates integer n towards right by b positions.

5. Write a C program with a function tolower, which converts upper case letters to lower case. Use conditional expression.

6. Write a function to get the transpose of a matrix.

7. Write a C program with a function indexr(s,t), which returns the index of right most occurrence of t in s otherwise -1.

8. Write a C program with a recursive function itoa, which converts integer into a string.

Pointers

1. A C program contains the following declaration
int x[8]= {10,20,30,40,50,60,70,80};
    What is the meaning of x?
    What is the meaning of (x+2)?
    What is the value of *x?
    What is the value of (*x+2)?
    What is the value of *(x+2)?.

2. A C program contains the following declaration
float table[2][3] = { { 1.1,1.2,1.3},{2.1,2.2,2.3}};
a) What is the meaning of a table?
b) What is the meaning of (table+1)?
c) What is the meaning of *(table+1)?
d) What is the meaning of (*(table+1)+1)?
e) What is the meaning of (*(table)+1)?
f) What is the value of *(*(table+1) +1)?
g) What is the value of *(*(table)+1)?
h) What is the value of *(*(table+1)?
i) What is the value of *(*(table) + 1)+1?

3. A C program contains the following declaration
char *color[6] = {“red”, “green”, “blue”, “white”, “black”, “yellow”};
a. What is the meaning of color?
b. What is the meaning of (color+2);
c. What is the value of *color?
d. What is the value of *(color+2)?
e. How do color[5] and *(color + 5) differ?.

4. Write a program to count the number of ‘e’ in the following array of pointer to
```
char * s [ ] = {
                “we will teach you how to “ ;
                “Move a mountain “ ;
                “Level a building “ ;
                “Erase the past “;
                “Make a million “ ;
} the string.
```

5. Write a function ``replace" which takes a pointer to a string as a parameter, which replaces all spaces in that string by minus signs, and delivers the number of spaces it replaced.
Thus, char *cat = "The cat sat";
        n = replace(cat);
        should set
        cat to "The-cat-sat"
        and
        n to 2.

Strings

1. Write a program to convert lower case string to upper case string and vice versa.

2. Write a program to reverse a string using recursive functions

3. Write a program to read n number of strings using two-dimensional character array, sort them and display the sorted list of strings on the screen.

4. Write a program to read n number of strings and display them on the screen. Use array of pointers and dynamic memory allocation techniques.

5. Write a C program with a function any (s1, s2). This function returns the first location (index of location) in the string s1 which matches with any string in s2 otherwise.

6. Write a C program with a function delete (s1, c). This function deletes each character in s1 which matches character c.

7. Write a Program to implement strtok library function.

8. Write a C program with a function deletes2 (s1, s2). This function deletes each character in string s1 which matches any character in string s2.

9. Write a function expand (s, t) which converts characters like newline and tab into visible escape sequences like \n and \t as it copies the string s to t. Use switch statement and also display both s and t at the end.

10. Write a function expand (s1, s2) which expands shorthand notations of s1 like a-d into abcd and 0-9 to 0123456789 in s2. For example if the string in s1 is 0123a-e1-4 then s1 is expanded in s2 to 0123abcde1234.

11. Write a program to print out all rotations of a string typed in. For eg:if the input is “Space”, the output should be: space paces acesp cespa espac.

12. Implement string library functions. strrev, strcpy, strcat, strcmp with same return values and all error handling features using pointers.

Structures,Unions and Enumeration

1. Write a program to represent time of the day in hrs, mins and secs. Use structures.

2. Define structure with two members (one int and other char). Also define s union with two members (one int and other char). Print the sizes of structure and union in number of bytes.

3. Define a structure declaration for each of the following situations. Assume a 16-bit integer word

a) Define three bit fields, called a, b and c, whose widths are 6-bits, 4-bits and 6-bits, respectively

b) Declare a structure-type variable v having the composition defined in part (a) above. Assign initial values 3, 5 and 7 respectively, to the three bit fields. Are the bit fields large enough to accommodate these values?

c) What are the largest values that can be assigned to each of the bit fields defined in part (a) above?

d) Define three bit fields, called a, b and c, whose widths are 8 bits, 6 bits and 5 bits, respectively. How will these fields be stored within the computer’s memory?

e) Define three bit fields, called a, b and c, whose widths are 8 bits, 6 bits and 5-bits respectively. Separate a and b with 2 vacant bits.

4. Develop a program to generate marks sheet of C-DAC, Hyderabad Students (DSSD, DESD and DAC courses). Modules are different for each course. Implement this using structures, unions, arrays, loops and variables.

5. Write a program to search for a given element in a list of elements using Linear Search. Use flag to represent the status of search. Define flag as an enumeration variable whose value is either true or false.

6. Write a menu driven C program to perform operations on Complex numbers. Use enumeration data type to identify the different operations on Complex numbers.

Files ,Console I/O and Command line arguments

1. Experiment to find out what happens when printf argument string contains \x, where x is some character (a, b, c, \, ^ etc). What are your observations.

2. Write a program to remove all the comments from a ‘C’ program.

3. Write a program that will concatenate two files, that is append the contents of one file at the end of another file and write the results into a third file. You must be able to execute command at DOS prompt as follows:
C > CONCAT Source 1.txt source 2.txt Target.txt.

4. Write a c program to printing the same file on the console.

5. Write a program which reads a line of characters. Each character entered from the keyboard is tested to determine its case, and is then written to the data file in opposite case. Display the contents of the file. Also use ftell and fseek to determine the current file position and to change the file position.

6. Write a program to embed assembly language code in C program.

7. Given as input an integer number of seconds, print as output the equivalent time in hours, minutes and seconds. Recommended output format is something like 7322 seconds is equivalent to 2 hours 2 minutes 2 seconds.

8. Read a positive integer value, and compute the following sequence: If the number is even, halve it; if it's odd, multiply by 3 and add 1. Repeat this process until the value is 1, printing out each value. Finally print out how many of these operations you performed.
Typical output might be:

```
Inital value is 9
Next value is 28
Next value is 14
Next value is 7
Next value is 22
Next value is 11Next value is 34
Next value is 17
Next value is 52
Next value is 26
Next value is 13
Next value is 40
Next value is 20
Next value is 10
Next value is 5
Next value is 16
Next value is 8
Next value is 4
```

Next value is 2
Final value 1, number of steps 19
If the input value is less than 1, print a message containing the word
Error
and perform an exit(0);

9. Write a program to count the vowels and letters in free text given as standard input. Read text a character at a time until you encounter end-of-data. Then print out the number of occurrences of each of the vowels a, e, i, o and u in the text, the total number of letters, and each of the vowels as an integer percentage of the letter total.

Suggested output format is:

```
    Numbers of characters:
    a 3 ; e 2 ; i 0 ; o 1 ; u 0 ; rest 17
    Percentages of total:
    a 13%; e 8%; i 0%; o 4%; u 0%; rest 73%
```

Read characters to end of data using a construct such as

```
        char ch;
        while(( ch = getchar() ) >= ) {
    /* ch is the next character */ ....}
```

to read characters one at a time using getchar() until a negative value is returned.

10. Write a program to read English text to end-of-data (type control-D to indicate end of data at a terminal, see below for detecting it), and print a count of word lengths, i.e. the total number of words of length 1 which occurred, the number of length 2, and so on.

Define a word to be a sequence of alphabetic characters. You should allow for word lengths up to 25 letters. Typical output should be like this:

```
length 1 : 10 occurrences
length 2 : 19 occurrences
length 3 : 127 occurrences
length 4 : 0 occurrences
length 5 : 18 occurrences
    ....
```

11. Write a program last that prints the last n lines of its text input. By default n should be 5, but your program should allow an optional argument so that

```
    last -n
```

prints out the last n lines, where n is any integer. Your program should make the best use of available storage. (Input of text could be by reading a file specified from the command or reading a file from standard input)

Bitwise Operations

1. Write a program to count number of bits as “1” in an 8 bit number.

2. Write a program in which define a as an unsigned integer whose value is (hexadecimal) 0xa2c3. Write the corresponding bit pattern for this value. Then evaluate each of the following bitwise expressions, first showing the resulting bit pattern and then the equivalent hexadecimal value. Utilize the original value of a in each expression. Assume that a is stored in a 16-bit word
a) ~a
b) a ^ 0x3f06c) a | 0x3f06
d) a | ~0x3f06
e) a >> 3
f) a << 5

g) a ^ ~a
h) a | ~a
i) (a & ~0x3f06) << 8
j) a & ~ (0x3f06 >> 8

3. Write a C program that will accept a hexadecimal number as input, and then display a menu that will permit any of the following operations to be carried out

- a) Display the hexadecimal equivalent of the one’s complement
- b) Carry out a masking operation and then display the hexadecimal equivalent of the result
- c) Carry out a bit shifting operation and then display the hexadecimal equivalent of the result
- d) Exit

If the masking operation is selected, prompt the user for the type of operation (bitwise and, bitwise exclusive or, or bitwise or) and then a (hexadecimal) value for the mask. If the bit shifting operation is selected, prompt the user for the type of shift (left to right), and then the number of bits.

4. Write a C program that will illustrate the equivalence between

- ◦ Shifting a binary number to the left n bits and multiplying the binary number by 2n
- ◦ Shifting a binary number to the right n bits and dividing the binary number by 2n.

Choose the initial binary number carefully, so that bits will not be lost as a result of the shifting operation.

Preprocessor

1. Write a C program to calculate factorial of a number. Factorial function has to be written as a multiline macro.

2. Develop sample programs using preprocessor operators #, ## and also conditional compilation.