Section Assignment and Tests

1a.

Write a C program that will read the number of Kilometers travelled by a bike in a float type variable d, it will also read the liters of Petrol used and store in float type variable f. It will then print the average fuel consumption Kilometers/Liters up to two decimal places.

1b.

Write a C program that will do the following:

i. Read a temperature value in Fahrenheit scale from the keyboard in a variable F

ii. Convert the temperature value to its corresponding value in Centigrade scale and store it in a variable C

iii. Print the value of C in the display

Note: the temperature values are real numbers. Also, they can be positive, negative, or 0.

1c.

Write a C program which reads the total number of days and finds out the number of years, weeks and days. Ignore leap year. A year has 365 days, and week has 7

1d.

A ball is released from the top of a tower of height h meters. It takes T seconds to reach the ground.

Write a c program which will -

•Read the value of h

•And print the position of the ball in T/3 seconds?

(Assume all values to be floating point numbers.)

2a.

Write a C program which reads the marks of a student in Physics, Chemistry, and Mathematics, and stores them in integer variables p, c, m. The program should print (Eligible/Not Eligible) based on the following eligibility criteria for admission to a degree course:

Marks in Maths >=65

Marks in Phy >=55

Marks in Chem>=50

Total in all three subject >=190 or Total in Math and Physics >=140

2b.

Write a C program to read the coordinates (x, y) (in Cartesian system) (float values) of a point and find the quadrant to which it belongs (Quadrant -I, Quadrant -II, Quadrant -III, Quadrant -IV). Assume that the point does not lie on any of the axes. The quadrants are numbered from 1st to 4th as follows: (where the signs of the (x, y) coordinates are I(+,+), II (−,+), III (−,−), and IV (+,−).

2c. Write a C program that reads the length (integers) of three sides of a triangle (variables a, b, c) and prints whether a triangle is Equilateral, Isosceles or Scalene.

2d.

Write a program to compute and print the taxi fare based on the following chart. Total number of Kilometres travelled will be input by the user as an integer.

First 12 KM: Rs. 100/-

Next 4 KM: Rs. 8 / KM

Next 4 KM: Rs 6 / KM

Above 20 KM: Rs 5 / KM

The program will -

i. Read in the distance travelled (integer)

ii. Print out the corresponding fare

For example, if a person has travelled 25 KM, the charges are:

100+4x8+4x6+5x5 = 181Rs

2e.

Write a program to read three distinct integers and print the largest among them.

2f.

Write a program that reads three integers representing time of a day. The integers are stored in variables: SS, MM, HH, representing seconds, minutes and hours. Write a program to print “valid time” if the integers represent a valid time

of the day. For example: 50, 32, 23 is a valid time, whereas 52, 71, 32, is not a valid time.

The program will –

i. Read three integers (SS, MM, HH)

ii. Print “Valid Time” if it is valid, print “Not a Valid Time” if it is not valid time of the day.

2g.

A student has three types of fruits in her hostel rooms – apple, banana, and cherry. There are a number of apples, b number of bananas, and c number of cherries. The values of a, b, and c are entered as inputs. The student eats one of the fruits everyday according to the following rule:

i. At any day she eats the fruit which is present in largest number on that day.

ii. If there are equal number of fruits of multiple types, her order of preference is apple > banana > cherry.

Write a program to print the type of fruit she eats on ith day, where i is input by the user. Assume, $i \leq a + b + c$.

Example:

If a = 5, b = 2, c= 3, then the sequence of eating the fruits is:

apple, apple, apple, cherry, apple, banana, cherry, apple, banana, cherry

3a.

Write a C program to read an integer n, and then print the value of factorial n (i.e., n!). Assume that n < 10. For example, if n = 5, then the program should print 120.

3b.

Write a C program which reads an integer n and then prints the sum of digits of n. For example, if n is 2020 it should print 4.

3c.

Cosine function can be expressed as the following infinite series form:

cosine(x) = $1 - (1/2!)x^2 + (1/4!)x^4 - (1/6!)x^6 + \ldots..$

Write a C program which shall take a floating-point variable x and evaluate the above cosine series to the 5th term of the series and print the approximate value of cosine(x). Do not call the factorial function.

3d.

Write a program in C to find the sum of the series 1 +11 + 111 + 1111 + ... n terms, where n is input by user.

3e.

Write a program in C to find the number and sum of all integer between a and b which are divisible by 9. The values of a and b are input by users.

3f.

Write a program in C to find the prime numbers within a range of numbers. a to b. Where a and b are input by users.

4a.

Write a program which reads the month number of a leap year as an integer m. With January as 1, and December as 12. The program should then print the number of days in the month. Use the case-switch statement.

4b.

A Ramanujan number is a positive integer that is expressible as the sum of two cubes in two different ways. Write a program that takes an integer n as input and prints all integers less than or equal to n that can be expressed as the sum of two cubes in two different ways, i.e., find distinct positive integers a, b, c, and d such that a3 + b3 = c3 + d3 <= n. Use four nested for loops. The smallest Ramanujan number is 1729, since 103 + 93 = 123 + 13 = 1729. (Please read the 3's as cubes)

4c.

Write a program that reads an integer N and prints a (2N + 1)-by-(2N + 1) diamond like the one below.

```
....*....
...***...
..*****..
.*******.
*********
.*******.
..*****..
...***...
....*....
```

5a.

Write a program which first reads an integer n (< 100). Then read n floating point numbers and store them in an one dimensional array. The numbers may be both positive or negative but no two of these n numbers have the same magnitudes. Now print the number that is closest to 0. Try not to use the abs/fabs functions.

5b.

Write a program which first reads an integer n (< 100). Then read n integers and store them in an one dimensional array. Assume that these n integers are unique and do not repeat. The program should then print the value of the second largest integer in the array. Do not sort the array.

5c.

Consider two sets A and B containing a and b non-repeating integers respectively. Write a program which first reads two integers a and b. Assume a. b <100. The program then reads a and b number of integers and stores them in arrays A and B. The program should next print:

i. The elements of union of sets A and B. (without repeating elements)

ii. The elements of intersection of sets A and B.

iii. The elements of A - B (i.e., the elements that are present in A but not in B).

6a.

Write a function int integerPower(int base, int exponent) that returns the value of base^exponent. For example, integerPower(3, 4) = 3 * 3 * 3 * 3. Assume that exponent is a positive, nonzero integer, and base is an integer. The function should use a loop to control the calculation. Do not use any math library functions. In the main function read two integers (<5) , call the function and print the result.

6b.

Write a function int digitSum(int x) to compute the sum of digits of x. Example: digitsSm(1572) is 15. Write another function int eqdigitSum(int x, int y) to check if inputs x and y have equal digit sums. The function returns 1, if x and y have the same digit sum, and 0, otherwise. The second function should call the first function. Write a main function to read two integers and print if they have equal digit sum.

6c.

Write a function int arrayMax(int A[], int n), which takes as input an integer array A[], and the number of elements in the array n (<100). The function returns the value of maximum element of the array. Write a main program, to first read n, and then read A[] element by element. Next call the function arrayMax and print the value of the maximum element of the array A[].

7a.

The choose function C(n,k) defines the number of ways k items can be chosen from a set of n items. Mathematically, it is defined as

C(n,k) = n!/(k!(n-k)!). Recursively, it can be defined as C(n,k) = (n* C(n-1,k-1))/k

- **i.** Write a recursive function that computes the above mathematical function. Make sure that you cover the base cases.
- **ii.** Write a program to test the above function. The program inputs n and k and outputs C(n,k). Assume that n >= k and both n and k are positive integers.

7b.

Define a sequence G_n as:

$G_n = 0$ if n = 0,

$= 1$ if n = 1,

$= 2$ if n = 2,

$= G_{n-1} + G_{n-2} + G_{n-3}$ if n >= 3.

Write a recursive function for the computation of G_n for a given n. From the main function read an integer n, (<10) and then print Gn.

7c.

Write a recursive function int arraymax(int A[], int n) to compute the largest integer in an array A[] of n elements. Assume all array

elements to be distinct. Write a main function to read n, then read n distinct integers into the array A[]. Call the function and print the largest element of the array. Do not sort the array.

TEST 1

Q1.

We are all waiting for the day when the number of cases of the pandemic on a day starts falling. Let us consider an equivalent problem. Write a program to check if a given sequence of numbers is sorted (in ascending order) or not. The program should do the following:

i. Read in an integer n

ii. Read n integers one by one in a loop. Do NOT use arrays (you will get zero if arrays are used!)

iii. Check if the integers are in ascending order and print a message accordingly. This can be done by checking if the integer read in the current iteration of the loop is ≥ the integer read in the previous iteration (remember it in another variable) for all integers read. It should then print "Integers in ascending order."

iv. If the integers entered are not in ascending order, your program should break out of the loop when the first time it detects an out of order number. It should print "Integers not in ascending order."

Solution 1:

```
#include<stdio.h>


int main()
{
   int temp,n,a; //declaring variables


   printf("Enter n: "); //inputting value of n
   scanf("%d",&n);
```

```
    printf("Enter Integer 1: "); //inputting first integer
    scanf("%d",&temp);

    for(int i=1;i<n;i++)  //creating a loop for rest n-1 inputs
    {
        printf("Enter Integer %d: ",i+1); //inputting a again and again for (n-1) times
        scanf("%d",&a);

        if(a>=temp) //if a>= previous value, we print Integers in ascending order
        {
            printf("Integers in ascending order\n");
        }
        else if(a<temp)//if a< previous value, we print Integers in ascending order
        {
            printf("Integers not in ascending order\n");
            break;
        }
        temp=a; //initializing temp to recent value of a
    }

    return 0;
}
```

Q2.

A number of persons present in a shop would like to maintain social distancing. This can be enforced by considering the coordinates of location of a person and the separation between two persons. Let us consider an equivalent problem. *Two circles are said to overlap if there exists a common point lying inside or on the boundary of both circles.*

a.

Write a function overlap(x1, y1, r1, x2, y2, r2) that receives the centre and radius of two circles as arguments. (x1, y1) and (x2, y2) are the centres of the two circles having radius r1 and r2 respectively. The function returns 1 if the circles overlap and 0 otherwise.

Assume all co-ordinate and radii values are non-negative. Assume all values are integers.

b.

Write a main( ) program that reads the value of an integer N and then reads in the center and radius of N circles. Assume N < 10. The radii are stored in an array R[ ] and the coordinates of the centers are stored in two arrays, X[ ] and Y[ ]. The coordinate (X[k], Y[k]) represents the center of the kth circle, while R[k] represents the radius of the kth circle.

Your program must then use the function overlap() to determine whether all pairs of circles overlap. If not, then it must print the centre and radius of every distinct pair of circles which do not overlap.

Solution 2:

```
#include<stdio.h>

#include<math.h>


//Part 1
```

```
int overlap(int x1,int y1,int r1,int x2,int y2,int r2)
{
   int dist;
   dist=sqrt(pow(x1-x2,2)+pow(y1-y2,2));

   if(dist>(r1+r2))
   {
      return 0;
   }
   else
   {
      return 1;
   }
}

//Part 2
int main(void)
{
   int n;
   printf("Enter N: ");
   scanf("%d",&n);

   int R[n],X[n],Y[n];
```

```
for(int i=0;i<n;i++)
{
   printf("Enter X[%d]:",i+1);
   scanf("%d",&X[i]);

   printf("Enter Y[%d]:",i+1);
   scanf("%d",&Y[i]);

   printf("Enter R[%d]:",i+1);
   scanf("%d",&R[i]);
}
int flag=1;

for(int i=0;i<n;i++)
{
   for(int j=i+1;j<n;j++)
   {
      if(overlap(X[i],Y[i],R[i],X[j],Y[j],R[j])==0)
      {
         flag=0;
         break;
      }
   }
    if(flag==0)
```

```
        {
            break;
        }
    }

    if(flag==1)
    {
        printf("All circles overlap\n");
    }
    else
    {

        for(int i=0;i<n;i++)
        {
            for(int j=i+1;j<n;j++)
            {
                if(overlap(X[i],Y[i],R[i],X[j],Y[j],R[j])==0)
                {
                    printf("Circles %d and %d does not overlap\nCenters=(%d,%d) and
(%d,%d)\nRadii=%d and %d \n\n",i+1,j+1,X[i],Y[i],X[j],Y[j],R[i],R[j]);
                }
            }
        }
    }
}
```

8a.

Define a structure to represent a circle in terms of coordinates of its centre (a, b) and its radius r. Assume these values to be float.

Write a function which takes as input two such structure variables and returns 1 if the circles intersect or if they touch at a point, and 0 if they do not. (Think a bit on how you can handle the case of touching at a point using float variables.)

In the main program input six numbers in a1, b1, r1, a2, b2, r2, order and store them in two such structure variables. Next, call this function and print if the input circles, intersect or touch at a point, or do not.

8b.

Write a program which reads an integer n, and then reads a n x n integer matrix. The program should then print the transpose of this matrix. If A' is a transpose of the matrix A, then A[i][j] = A[j][i]. Print the original and the transposed matrix in the usual matrix format. For example, if n = 2, and the input matrix is :

1 2

3 4

the output matrix is :

1 3

2 4

8c. Write a program which reads an integer n, and then reads a n x n integer matrix such that it is filled up with the numbers 1 to n^2.. Each number can be used only once. Assume that the users follows these rules. Now, check and print whether the matrix is a magic square. A magic square is the one where the sum of each of the rows, each of the columns and each of the two major diagonals is the same.

Here is an example of a 3 x 3 magic square:

4 9 2

3 5 7

8 1 6

Here is an example of a 4 x 4 magic square:

7 12 1 14

2 13 8 11

16 3 10 5

9 6 15 4

9a.

Write a program to read a character string and store it in an array. Suppose the string stores a password. A strong password should have at least 8 characters. Also, it should have at least 1 numeral (0-9) and at least 1 upper case alphabet (A-Z). Print whether the input string is a strong password.

9b.

Input two strings s and t from the user. Assume that the strings do not contain blank or any whitespace character. Write a function that finds if s is a palindrome of t, i.e., whether the reverse of s is equal to t. If it is a palindrome, return 1; otherwise, return 0.

The parameters of the function contain only pointers to characters (not arrays):

```
int ispalindrome(char *s, char *t)
```

Do not use any string library functions. In the main function read two strings. Call the above function and print its output.

9c.

Write a function void dict_first(char s[], int n, char * first), which given an array s of n (< 10) strings (each of length < 20), finds the first string in the array if the strings were arranged according to dictionary order. In the above function, use the strcmp function in the strings.h library to compare dictionary order between two strings. Write a main function which reads an integer n. Then reads n strings consisting of lower case alphabets only, calls the above function, and prints the first string according to dictionary order.

Example, if n = 4, and the input strings are “hello”, “world”, “happy”, “diwali”, output string should be “diwali”.

10a.

Read an integer n (< 100). Input an integer array A[] of n elements from the user. The array can have multiple occurrences of elements. You have to remove the multiple occurrences of elements and store the distinct elements in a new array. The memory for the new array has to be dynamically allocated using malloc such that its size is same as the number of distinct elements in A[].

10b.

Write a program which first reads character string S (of length at most 100). Assume that the string contains only upper case characters. The program should then find out the number of characters in the string S. It should then remove the vowels and store the new string in a new character array T. Size of T should be dynamically allocated. Print the string T and the number of characters it contains. For example, if string S is HELLOWORLD then the new string T is HLLWRLD of size 7.

10c.

A three dimensional vector can be represented as a 1-D array containing three floating point values. Write a program which inputs a positive integer n from the user and then input n three dimensional vectors. The memory for storing these n vectors must be allocated dynamically. Now, compute the pairwise distance among these n vectors and store them in a n x n matrix. The memory for this matrix should also be allocated dynamically. Use can use the sqrt function in math.h.

11a.

Define a structure “node” to store a link list of integers.

Write a function void make_list(node *l, int n) to read n integers and store them in a link list.

Write a main program to read an integer n. Then uses make_list function to read n integers and stores them in a list l. Print the list.

11b.

Define a structure “node” to store a link list of integers.

Write a function void make_list(node *l, int n) to read n integers and store them in a link list.

Write a function int min_list(node *l) which takes as input a link list and returns the value of the smallest element of the link list.

Write a main program to –

Read an integer n. Then uses make_list function to read n integers and stores them in a list l.

Call min_list function compute and then print the smallest element of list l.

Example: Input: 1, 2, 3, 4, 5 Output: 1

11c.

Define a structure “node” to store a link list of integers.

Write a function void make_list(node *l, int n) to read n integers and store them in a link list.

Write a function node *reverse_list(node *l) which takes as input a link list and returns another list that stores the elements in opposite order of the original list.

Write a main program to –

Read an integer n. Then uses make_list to read n integers and stores them in a list l.

Call reverse_list to reverse the list l.

Print elements of the new list.

Example: Input: 1, 2, 3, 4, 5 Output: 5, 4, 3, 2, 1

TEST 2

You have to submit a number of assignments for every subject in a semester. Your academic activity during a semester is defined by a sequence of assignments with associated deadlines and marks. Deadline of an assignment is represented by an integer counting the number of days from start of the semester. Marks is represented by a floating point number. The sequence of assignments for a subject can be represented by a link list of assignments arranged in increasing order of deadlines. The suggested structure for a node in the linked list is as follows:

```
struct node {
 int assnid;
int deadline;
float marks;
struct node *next;
};
```

Assume that deadlines of no two assignments in a semester are the same. Write the following functions:

1. Write function struct node *make_list(int n) to read records for n assignments and store

them in a link list.

2. Write a function, float total_marks(struct node *list), to return the total marks of all the

assignments represented by the linked list passed as the argument.

3. Write a function:

```
struct node *combine(struct node *list1, struct node *list2)
```

to return the linked list obtained by merging the assignments for two subjects in a semester represented by list1 and list2 respectively. Assignments in the merged list must also be ordered by their deadlines.

i. Write a main() function which reads two integers n1, and n2. And then reads the records for two subjects L1, and L2 using the make_list function.

ii. For each subject L1, L2, print the list of assignment ids and their marks (one assignment in each line), followed by the total marks computed using the function, total_marks.

iii. Use the function, combine, to merge L1 and L2 and then print the list of assignment ids and marks in the merged list (one assignment in each line), followed by the total marks computed using the function, total_marks.