STUCTURED PROGRAMMING APPROACH (DEC 18)

Q.1)

a) What is recursion? Write a program to find x^y using recursion. (4 M)

Ans:

- A function that calls itself is called as a recursive function and the implementation of a program that uses recursive function is called as recursion.
- A recursive function must definitely have a condition that exits from calling the function again.
- Hence there must be a condition that calls the function itself if that condition is true.
- If the condition is false then it will exit from the loop of calling itself again.
- The condition could also be implemented vice versa i.e. if the condition is false then the function calls itself else if the condition is true, it will exit from the loop calling itself again.
- Syntax:

```
void recursion() {
  recursion(); /* function calls itself */
}

int main() {
  recursion();
}
```

Program:

```
#include<stdio.h>
#include<conio.h>
#include<math.h>
int powers(int n1, int n2);
void main()
{
int base, power, ans;
clrscr();
printf("Enter a number");
scanf("%d",&base);
printf("Enter power number");
scanf("%d",&power);
ans = powers(base, power);
```

```
printf("%d^%d = %d",base, power, ans);
getch();
}
int powers(int base, int power)
{
 if(power != 0)
    return (base*powers(base, power-1));
 else
    return 1;
}
```

Output:

Enter a number 2

Enter power number 3

2^3 = 8

---

b) State any two library functions string.h along with its syntax, use an example. (4 M)

Ans:

strlen() function:

- This function returns an integer value that is the length of the string passed to the function.
- When returning the length of the string it does not consider the space required for null character.
- Hence it returns the exact length of the string neglecting the space required for null character.

Syntax: strlen(str)

- Example: strlen(a); This will find the length of string of a.

strcpy() function:

- This function copies the second string into the first string passed to the function.

- The second string remains unchanged. Only the first string is changed and gets a copy of the second string.

Syntax: strcpy(str1,str2)

Example: strcpy(b,a); This will copy the string from a to b.

---

c) What is pointer? Explain how the pointer variable declared and initialized. (4 M)

Ans:

- Pointers are variables that are used to store the address of another variable.
- Address of a variable is the memory location number which is allotted to the variable.
  The memory addresses are 0, 1, 2, 3… and so on up to the capacity of the memory.
  The address is normally displayed in hexadecimal form. Hexadecimal form is a representation of number somewhat similar to binary number. Here four binary digits are combined together to form a hexadecimal number.
- Pointers unlike other variables do not store values. As stated they store the address of other variables.
- It is already mentioned in the first statement that pointers are also variables. Hence, we can also have a pointer that is pointing to another pointer.
- Syntax of pointer declaration : Data_type *ptr_name;
- Wherein "Data_type" is the data type of the variable to which the pointer is supposed to point. If we want a pointer to point to an integer than, we need to have the data type of the pointer as "int", for a float type data pointer should also be of the "float" type and so on.
- The "ptr_name" is an identifier i.e. the name of the pointer. The same rules of identifiers apply to the pointer name as to any other variable declaration. The most important difference in the declaration of a pointer is the "*" sign given before the pointer name.
- Hence, according to the syntax seen above, if we want to declare a pointer for "int" type data then we can declare it as given in the example below: int *p;
  Here, the pointer name is "p". Hence, "p" can be used as a pointer to point to any of the variable of type "int".
- Syntax : data_type *var_name;
- Example : int *p; char *p;

---

d) Write the output of following code:

```
#include<stdio.h>
int main()
{
   int val = 1;
   do{
         val++;
         ++val;
   }while(val++>25);
   printf("%d\n",val);
   return 0;
}
```

(4 M)

Ans:

Output: 4

---

e) Write a program to validate whether accepted string is palindrome or not.

(4 M)

Ans:

```
#include<stdio.h>
#include<conio.h>
#include<string.h>
void main()
{
char a[100],b[100] ;
clrscr();
printf("Enter a string:");
gets(a);
strcpy(b,a);
strrev(b);
if(strcmp(a,b)==0)
```

```
                printf("The entered string is pallindrome");
else
                printf("The entered string is not pallindrome");
getch();
}
```

Output:

Enter a string: madam

The entered string is palindrome

Q.2)

a) Write a program to multiply two matrices after checking compatibility.

(10 M)

Ans:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[10][10], b[10][10],mul[10][10];
int i,j,c,r,n;
clrscr();
printf("Enter the number of columns:");
scanf("%d",&c);
printf("Enter the number of rows:");
scanf("%d",&r);
printf("Enter the elements of first matrix:");
for(i=0;i<c;i++)
{
 for(j=0;j<r;j++)
 {
  scanf("%d",&a[i][j]);
 }
}
printf("Enter the elements of second matrix:");
```

```
for(i=0;i<c;i++)
{
 for(j=0;j<r;j++)
 {
  scanf("%d",&b[i][j]);
 }
}
printf("\nThe Matrix after multiplication is:\n");
for(i=0;i<c;i++)
{
 for(j=0;j<r;j++)
 {
  mul[i][j] = a[i][j]*b[i][j];
 }
}
for(i=0;i<c;i++)
{
 for(j=0;j<r;j++)
 {
  printf("%d\t",mul[i][j]);
 }
 printf("\n");
}

getch();
}
```

Output:

```
Enter the number of columns: 3
Enter the number of rows: 3
Enter the elements of first matrix: 1 2 3 4 5 6 7 8 9
Enter the elements of second matrix: 1 2 3 4 5 6 7 8 9
The Matrix after multiplication is:
1       4       9
16      25      36
49      64      81
```

b) What is file? What are different functions available to read data from file? Specify the different modes in which files can be opened along with syntax.

(10 M)

Ans:

- For file handling or accessing the contents of file, there are certain predefined functions available in the C programming language.
- An important thing required to access files is the "FILE pointer". This pointer is used to point to the values stored in the file. A file pointer is hence to be created for accessing the files. The syntax for creating a file pointer is as given below: FILE *<identifier for pointer>; For e.g. FILE *fp;
- Hence in every program we write in this section to access files, we will use this kind of pointer declaration. This pointer is used to point the data to be accessed in the file i.e. whenever a data is read or written in the file, it is from the location pointed by the file pointer "fp".
- **File operations are as follows:**
  1. **fopen()**: This function is used to open a file to be accessed in the program.
     - The file to be opened is to be passes as a string parameter to the function and also the mode of opening the file is to be passed as the string to the function.
     - Hence the syntax of the function with parameters is as given below : <file pointer identifier> = fopen("<file name>", " <mode of opening the file>")
     - For e.g. fp=fopen("test.txt","w"); This example statement opens the file "test.txt" in write mode and the pointer used along with the function to read/write is the file pointer "fp".
     - The various modes in which a file can be opened are as listed below :
       I. "r" indicates that the file is to be opened indicates in the read mode.
       II. "w" indicates that the file is to be opened indicates in the write mode. When a file is opened in write mode the data written in the file overwrites the previously stored data in the file.
       III. "a" indicates that the file is to be opened in the append mode. In this mode the data written into the file is appended towards the end of the already stored data in the file. The earlier stored data remains as it is.
       IV. "w+" indicates that the file is to be opened in write and read mode (v) "r+" indicates that the file is to be opened in read and write mode.
  2. **fclose():** This function is used to close the file opened using the file pointer passed to the function.

- The syntax with parameters to call this function is as given below : fclose(<file pointer identifier>);
- For e.g. fclose(fp);

3. **fputc():** This function is used to put a character type data into the opened file using the fopen() function, pointed by a file pointer.

- The syntax to call this function along with the parameters to be passed is as shown below : fputc(<char type data>, <file pointer identifier>);
- For e.g. : fputc(c,fp); This example will store the character value of the char type data variable "c" into the opened file and pointed by the file pointer fp, at the position pointed by the pointer fp in the file.

4. **getc()**: This function is used to get a character from the file pointed by the corresponding file pointer passed to the function.
    - It is exactly opposite the fputc() function. This function brings the character from the file opened and pointed by the file pointer variable passed to the function.
    - The syntax of the function call with the parameters to be passed is as given below :getc(<file pointer identifier>);
    - For e.g. getc(fp);

Q.3)

a) Write a program to find transpose of matrix without making use of another matrix. (10 M)

Ans:

```
#include<stdio.h>
#include<conio.h>
void main()
{
int a[10][10];
int i,j,c,r,n;
clrscr();
printf("Enter the number of columns");
scanf("%d",&c);
printf("Enter the number of rows");
scanf("%d",&r);
printf("Enter the elements of matrix");
for(i=0;i<c;i++)
{
```

```
 for(j=0;j<r;j++)
 {
  scanf("%d",&a[i][j]);
 }
}
printf("The matrix before transpose:\n");
for(i=0;i<c;i++)
{
 for(j=0;j<r;j++)
 {
  printf("%d\t",a[i][j]);
 }
 printf("\n");
}
printf("The matrix after Transpose:\n");
for(i=0;i<c;i++)
{
 for(j=0;j<r;j++)
 {
  printf("%d\t",a[j][i]);
 }
 printf("\n");
}
getch();
}
```

Output:

Enter the number of columns: 3

Enter the number of rows: 3

Enter the elements of matrix: 1 2 3 4 5 6 7 8 9

The matrix before transpose:

1 2 3

4 5 6

7 8 9

The matrix after transpose:

1 4 7

2 5 8

3 6 9

b) Define structure consisting of following elements

1. student roll_no

2. student name

3. student percentage

Write a program to read records of 5 students of and display same. (10 M)

Ans:

- The Structure for above given elements can be defined as :

```
struct student
{
char stud_name[20];
int roll_no;
int percentage;
};
```

Program:

```
#include<conio.h>
#include<stdio.h>
struct student
{
char stud_name[20];
int roll_no;
int percentage;
};
void main ()
{
struct student st[100];
int n,i,j;
clrscr();
for(i=0;i<=4;i++)
{
printf("Enter the student's name, roll number and percentage:");
scanf("%s %d %d" , st[i].stud_name,&st[i].roll_no,&st[i].percentage);
```

```
}
printf("Name\tRoll No\tpercentage\n");
printf("-----------------------------------------\n");
for(i=0;i<=4;i++)
{
printf("%s\t%d\t%d\n",st[i].stud_name, st[i].roll_no, st[i].percentage);
}
getch();
}
```

Output:

```
Enter the student's name, roll number and percentage: ketan
01
90
Enter the student's name, roll number and percentage: carry
02
91
Enter the student's name, roll number and percentage: tom
03
92
Enter the student's name, roll number and percentage: jerry
04
95
Enter the student's name, roll number and percentage: bob
05
96
Name            Roll No         Percentage
-------------------------------------------------
ketan            01              90
carry            02              91
tom              03              92
jerry            04              95
bob              05              96
```

Q.4)

a) Write a program to calculate summation of series
1/1! + 2/2! + 3/3! +….+ n/n! (10 M)

Ans:

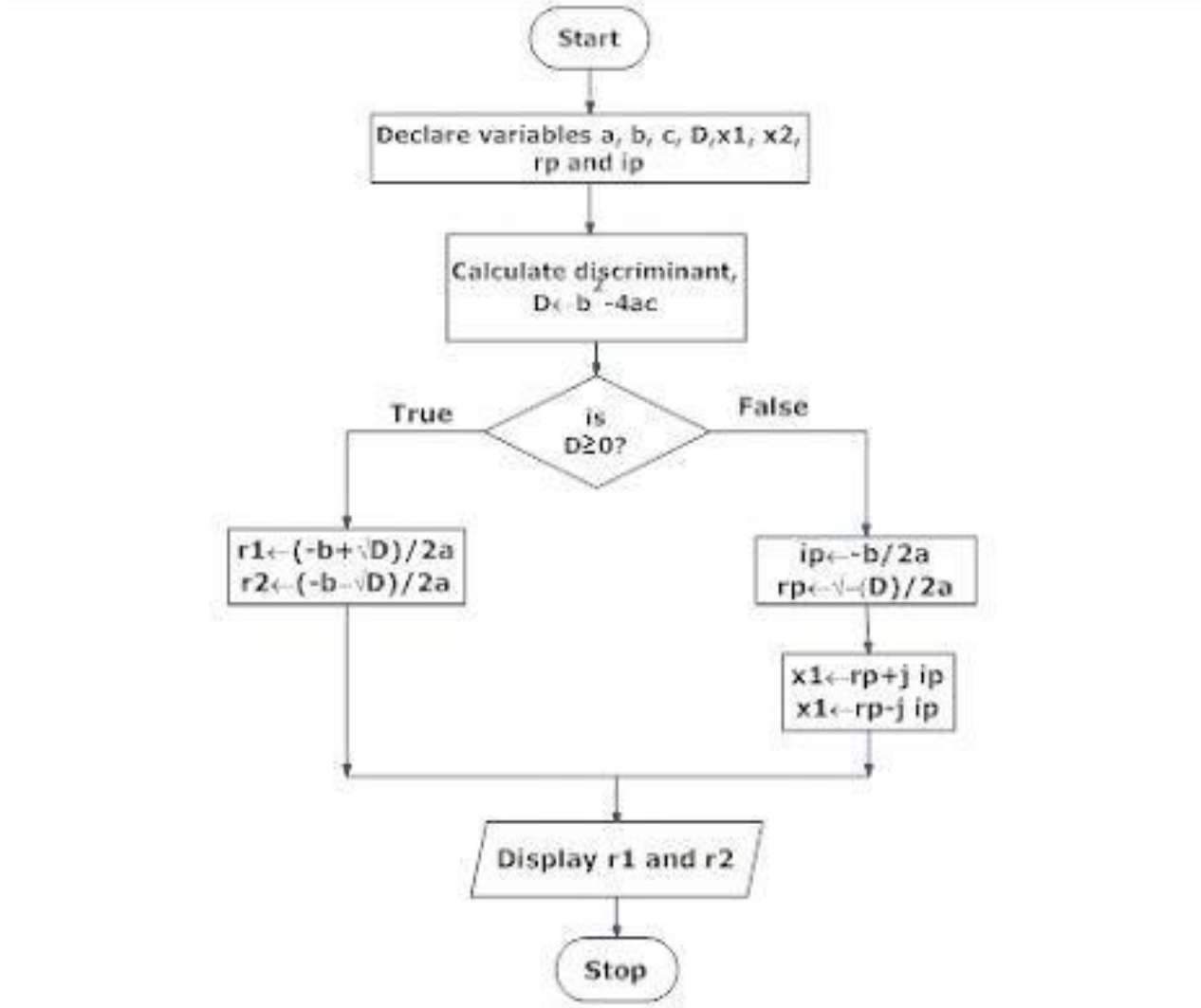
```
#include <stdio.h>
#include <conio.h>
double sumseries(double m)
{
   double sum2 = 0, f = 1, i;
   for (i = 1; i <= m; i++)
   {
                    f = f * i;
                    sum2 = sum2 +(i / f);
   }
   return(sum2);
}
double sumseries(double);
main()
{
   double number,sum;
   clrscr();
   printf("\n Enter the value:  ");
   scanf("%lf", &number);
   sum = sumseries(number);
   printf("\n Sum of the above series = %lf ", sum);
}
```

Output:
Enter the value: 4
Sum of the above series = 2.666667

---

b) Draw the flowchart for finding the roots of quadratic equation. Write program for same. (10 M)

Ans:




```
#include <stdio.h>
#include <math.h>
int main()
{
   double a, b, c, discriminant, r1, r2, rp, ip;
   printf("Enter coefficients a, b and c: ");
   scanf("%lf %lf %lf",&a, &b, &c);
   discriminant = b*b-4*a*c;
   if (discriminant > 0)
   {
      r1 = (-b+sqrt(discriminant))/(2*a);
       r2 = (-b-sqrt(discriminant))/(2*a);
      printf("r1 = %.2lf and r2 = %.2lf",r1 , r2);
```

```
    }
    else if (discriminant == 0)
    {
        r1 = r2 = -b/(2*a);
        printf("r1 = r2 = %.2lf;", r1);
    }
    else
    {
        rp = -b/(2*a);
        ip = sqrt(-discriminant)/(2*a);
        printf("r1 = %.2lf+%.2lfi and r2 = %.2f-%.2fi", rp, ip, rp, ip);
    }
    return 0;
}
```

Output:

```
Enter coefficients a, b and c: 2.3
4
5.6
Roots are: -0.87+1.30i and -0.87-1.30i
```

Q.5)

a) Write a program to implement calculator with following operations using switch case

1. add two numbers

2. subtract two numbers

3. multiply two numbers

4. divide two numbers **(10 M)**

Ans:

```
#include <stdio.h>
#include<conio.h>
void main()
{
    int num1,num2;
    float result;
    char ch;
```

```
    clrscr();
    printf("Enter first number: ");
    scanf("%d",&num1);
    printf("Enter second number: ");
    scanf("%d",&num2);

    printf("Choose operation to perform (+,-,*,/): ");
    scanf(" %c",&ch);

    result=0;
    switch(ch)
    {
          case '+':
             result=num1+num2;
             break;

          case '-':
             result=num1-num2;
             break;

          case '*':
             result=num1*num2;
             break;

          case '/':
             result=(float)num1/(float)num2;
             break;
          default:
             printf("Invalid operation.\n");
    }

    printf("Result: %d %c %d = %f\n",num1,ch,num2,result);
    getch();
}
```

Output:

```
Enter first number: 10
Enter second number: 3
Choose operation to perform (+,-,*,/,%): /
Result: 10 / 3 = 3.333333
```

b) What do you mean by file? What are the different functions available to read data from file? Specify the different modes in which file can be opened along with syntax. (10 M)

Ans:

- For file handling or accessing the contents of file, there are certain predefined functions available in the C programming language.
- An important thing required to access files is the "FILE pointer". This pointer is used to point to the values stored in the file. A file pointer is hence to be created for accessing the files. The syntax for creating a file pointer is as given below: FILE *<identifier for pointer>; For e.g. FILE *fp;
- Hence in every program we write in this section to access files, we will use this kind of pointer declaration. This pointer is used to point the data to be accessed in the file i.e. whenever a data is read or written in the file, it is from the location pointed by the file pointer "fp".
- **File operations are as follows:**
  1. **fopen()**: This function is used to open a file to be accessed in the program.
     - The file to be opened is to be passes as a string parameter to the function and also the mode of opening the file is to be passed as the string to the function.
     - Hence the syntax of the function with parameters is as given below : <file pointer identifier> = fopen("<file name>", " <mode of opening the file>")
     - For e.g. fp=fopen("test.txt","w"); This example statement opens the file "test.txt" in write mode and the pointer used along with the function to read/write is the file pointer "fp".
     - The various modes in which a file can be opened are as listed below :
       I. "r" indicates that the file is to be opened indicates in the read mode.
       II. "w" indicates that the file is to be opened indicates in the write mode. When a file is opened in write mode the data written in the file overwrites the previously stored data in the file.
       III. "a" indicates that the file is to be opened in the append mode. In this mode the data written into the file is appended towards the end of the already stored data in the file. The earlier stored data remains as it is.

IV. "w+" indicates that the file is to be opened in write and read mode (v) "r+" indicates that the file is to be opened in read and write mode.

2. **fclose():** This function is used to close the file opened using the file pointer passed to the function.
   - The syntax with parameters to call this function is as given below : fclose(<file pointer identifier>);
   - For e.g. fclose(fp);

3. **fputc():** This function is used to put a character type data into the opened file using the fopen() function, pointed by a file pointer.
- The syntax to call this function along with the parameters to be passed is as shown below : fputc(<char type data>, <file pointer identifier>);
- For e.g. : fputc(c,fp); This example will store the character value of the char type data variable "c" into the opened file and pointed by the file pointer fp, at the position pointed by the pointer fp in the file.

4. **getc()**: This function is used to get a character from the file pointed by the corresponding file pointer passed to the function.
   - It is exactly opposite the fputc() function. This function brings the character from the file opened and pointed by the file pointer variable passed to the function.
   - The syntax of the function call with the parameters to be passed is as given below :getc(<file pointer identifier>);
   - For e.g. getc(fp);

Q.6) **(10 M)**

a) Write a program to generate following patterns.

1. 1

1 2

1 2 3

1 2 3 4

1 2 3 4 5

Ans:

```
#include <stdio.h>
```

```
#include<conio.h>
void main()
{
int i,j;
clrscr();
for(i=1;i<=5;i++)
{
for(j=1;j<=i;j++)
{
 printf("%d\t",j);
}
printf("\n");
}
getch();
}
```

Output:

```
1
1  2   1   2   3
1   2   3   4
1   2   3   4   5
```

2. **1**
2 3
3 4 5
6 7 8 9

Ans:

```
#include<stdio.h>
int main()
{
   int i,j,k;
   for(i=1;i<=5;i++)
   {
      j=i;
      for(k=1;k<=i;k++)
      {
         printf("%d",j++);
      }
      printf("\n");
   }
   return 0;
}
```

Output:

```
1
2  3
3  4  5
6  7  8  9
```

---

b) Explain call by value and call by reference with example.

Ans:

Call by value:

- In this case the value of parameters is passed to the called function.
- In this case the actual parameters are not accessible by the called function.
- This is implemented by using a simple variable name.
- Hence the actual parameters remain unchanged in case of call by value.

Example:

```
#include<stdio.h>
```

```
#include<conio.h>
void main()
{
int a,b;
void swap (int a, int b);
clrscr();
printf("Enter two numbers:");
scanf("%d %d",&a,&b);
printf("The values of a and b in the main function before calling the swap function are
%d and %d\n",a,b);
swap(a,b);
printf("The values of a and b in main function after calling the swap function are %d
and %d\n",a,b);
getch();
}
void swap (int a, int b)
{
int temp;
temp=a;
a=b;
b=temp;
printf("The values of a and b in the swap function after swapping are %d and
%d\n",a,b);
}
```

Call by reference:

- In this case the reference of variable is passed to the function by passing the address of parameters.
- In this case, since the address of the variables are available the called function can access the actual parameters.
- This are implemented by use of pointer variables.
- Hence the parameters can be altered if required in case of call by reference method.

Example:

```
#include<stdio.h>

#include<conio.h>

void main()

{
```

```
int a,b;
void swap (int *p1, int *p2);
clrscr();
printf("Enter two numbers:");
scanf("%d %d",&a,&b);
printf("The values of a and b in the main function before calling the swap function are %d and %d\n",a,b);
swap(&a,&b);
printf("The values of a and b in main function after calling the swap function are %d and %d\n",a,b);
getch();
}
void swap (int *p1, int *p2)
{
int temp;
temp=*p1;
*p1=*p2;
*p2=temp;
printf("The values of a and b in the swap function after swapping are %d and %d\n",*p1,*p2);
}
```
