Program/ Course: **B. Tech. (Sem.–1st/2nd)**
Name of Subject: **Programming for Problem Solving**
Subject Code: **ESC-104**
Paper ID: **15935**

Time Allowed: 03 Hours **Max. Marks: 60**

NOTE:

1) **Part A and B is compulsory**
2) **Part C** has Two Questions Q8 and Q9. Both are compulsory, but with internal choice
3) Any missing data may be assumed appropriately

Part – A [Marks: 02 each]

Q1.

a) List various functions of operating system.
b) Give the precedence of operators in C.
c) Define recursion.
d) What is ROM? Why it is called non volatile memory?
e) Write about 'strlen' and 'strcat' functions.
f) Explain how we declare a structure variable.

Part – B [Marks: 04 each]

Q2. Write syntax for opening and writing on file using file stream operations.
Q3. Explain in detail different types of memories available in a computer system.
Q4. How are pointers important in C? How are they declared? Describe with the help of an appropriate program.
Q5. Define array. Write a program to add ten numbers using arrays.
Q6. Design a program to find the factorial of a number entered by the user.
Q7. What do you mean by term Data type? Explain in detail the various data types supported by C language, along with their memory requirements.

Part – C [Marks: 12 each]

Q8. Explain importance of control statements? Explain 'Continue', 'Goto' and 'Break' statements with examples.

OR

Describe following with examples:
(a) Ackerman function
(b) Flow chart, Algorithms and Pseudocode

Q9. Compare the following with examples:
(a) Formal and Actual arguments
(b) Call by value and Call by reference

OR

Explain in detail the working of linear search algorithm? Write a program to illustrate the concept linear search.

****

Dt
Pg. B+

* If-else

→ This statement is used to carry out one of the two sets of statements either within if block or in else block.

* nested - If

→ Within one if block or else block another if - - else statement can be placed. Such statements are called nested if statements.

* The ladder if else statement has more than one expression.

Syntax of switch

```
switch (expression)
{
    case value1:
            set of statements;
            break;
```

```
Switch (expression)
{
    case value1:
            set of statements;
            break;
```

Be Positive...

Eg. of Switch:

```
# include <stdio.h>
void main()
{
   int week;
printf("Enter the week no.(1-7)")
scanf("%d", &week);

switch(week)
{
    Case 1:
    printf("Monday");
      break;

    Case 2:
    printf("Tuesday");
    break;

    Case 3:
    printf("Wednesday")
    break;

    Case 4:
    printf("Thursday")
    break;

    Case 5:
    printf("Friday")
    break;
```

```
case 6:
printf ("Saturday");
break;

case 7:
printf ("Sunday");
break;

default;
    printf ("Invalid statement");
    break;
}
}
```

If - else	switch
(1) Allows complex expression	(1) Allows simple expressions
(2) The expression evaluates to true to false	(2) The results of the expression will be either integer or character.

Goto:-

→ A goto statement in C programming language provides an unconditional jump from goto to a labelled statement.

Looping statements :-

A loop is a sequence of instructions that is repeated until a certain condition is reached.

Entry controlled loop

for statement

```
for (initialization; expression; flow)
{
    Set of statements → body of for loop
}
```

- While loop

```
while (condition)
{
    set of statements → body of while loop
}
```

→ Exit controlled loop

do-while

```
do
{
    set of statements
}
while(condition)
```

* Break & Continue

→ Break → Break is used in terminating the loop immediately after the loop is encountered.

* Continue:- It is sometimes desirable to skip some statements inside the loop and continue with the next iteration of the loop. In such case, continue is used.

Pg No. B+

```
#include<stdio.h>
void main()
{
int i;

for(i=0;i<10,i++)
{
printf("%d",&i);
}
getch()
}
```

Output

0
1
3
3
3
:
:
:
10

for - while

```
#include<stdio.h>
void main()
{
int i=1;

while(i<=5)
{
printf("%d",i);
i++;
}
getch();
}
```

Output

1
2
3
4
5

```
#include<stdio.h>
void main()
int i=6.
do
{printf("valid statement.
}
while(i<=5)
{
printf("%d",&i);
}
getch();
}.
```

Be Positive...

While	do-while
(1) While loop check condition before iteration of the loop	(1) Do-while loop verifies the condition after the execution of statements inside the loop
(2) It is entry controlled loop	(2) It is exit controlled loop
(3) The iterations do not occur if, the condition at the first iteration, appears false.	(3) The iteration occurs at least once even if the condition is false at the first iteration.

(4) while (condition)
{
set of statement;
# body of while loop 	
}.
	(4) do
	{
	statements
	# body of do-while loop
	}
	while(condition);

Dt.
Pg. B+

Functions

→ A function is a set of statements that together perform a task. In every C program has at least 1 function which is main() and all the most trivial programs can define additional functions.

→ A function is a set of statements that together perform a task. In every C program there is atleast one function which is main() and all the most trivial programs can define additional functions.

* ① Function declaration / prototype
② Function definition
③ Function call

* Advantages

2 -> byte

Bubble,
Insertion
Selection
Quick sort
Merge sort
Fibonacci series

2d & matrices

File :-

→ sequence of bytes on the disk
→ permanent storage of data
→ readymade structure
→

fopen()

syntax

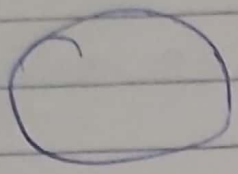
File * fopen

→ A file represents a sequence of bytes on the disk where a group of related data is stored. File is created for permanent storage of data.

→ A pointer is a variable which contains the adress in memory of another variable.

2 bytes

Be Positive...



oval — Start or end

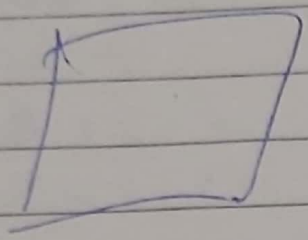
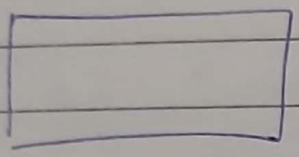
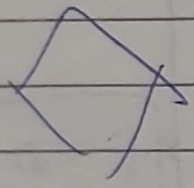


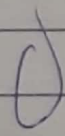
Diagram — Input/Output



Processing



Diamond — Decision



Connector

Direction — Flow lines

: + - * / % .

(1) Arithmetic →

(2) Relational → <, <=, >, >=, ==, !=

(3) Logical → ||, &&, !

(4) Bitwise → >>, <<, &, ^

(5) Special → ++, --, ,

Array

```
# include <stdio.h>
void main()
{ int i;
  char arr[5]

  for(i=0, i<3, i++
  {
    printf("%d", arr[i]
  }
  getch();
}
```

```
# include <stdio.h
void main()
 {
  int a[5].
  int i;
  printf("Enter 5 number");
  for (i=0, i<=5, i++)
    {
      scanf("%d", a[5]);
    }
  for(i=0, i<5, i++
      {
        printf("%d", a[i]);
      }
      getch()
}.
```

Dt
Pg. B+

(iv) Take ~~nothing~~ something, return something.

```
#include <stdio.h>
int add(int, int);
void main()
{
  int x, y, i;
  printf("Enter the value of x & y");
  scanf("%d%d", &x, &y);

  i = add(x, y)
  printf("%d", i);
   getch()
}
int add(int a, int b)
 {
   int c;
   c = a+b;
   return c;
 }
```

Call by value.

```
#include <stdio.h>
void main()
{
  int x, y
  swap(x, y);

  getch()
}
```

Be Positive...

2202C

Dt. Pg. B+

```
swap (int a, int b)
{
    int t;
    t = a;
    a = b;
    b = t;
}
```

* Recursion

```
#include <stdio.h>
int factorial (int);
void main()
{
  int factorial, fact, n;
  printf ("Enter the value of n");
    scanf ("%d", &n);
fact = factorial(n);
  printf ("%d", fact);
    getch();

}
  int factorial(n)
  {
    int i;
    if (i==0 || i==1)
    {
      return 1;
    else
      return n * factorial(n-1);
}
```

Be Positive...

Recursion is a programming technique that allows the programmer to express operations

Dt.
Pg. B+

Recursion

```
#include <stdio.h>
int ~~fact~~ fib(int)
void main()
{
  int t, i;
  printf("Enter the number of terms");
  scanf("%d", &t);
  for(i=0, i<t, i++)
  {
    printf("%d", fib(i));
  }
  getch();
  clrscr();
}
int ~~fact~~ fib(n)
{
  if(n<=1)
    return 1;
  else
    return fib(n-1) + fib(n-2);
}
```

1) 2 = 0+0

16 | 2 | 1

2.

10 | 1 2 | 1 8+1=3

1 0

2

Ackermann function

```
# include <stdio.h>
~~void~~
~~void~~ void main()
    {
      int a,b;
      printf("Enter 2 numbers");
      scanf("%d%d", &a, &b);
      printf("Output = %d", A(a,b));
      getch();
      clrscr();
    }

int A(int m, int n)
  {
    if (m==0)
      return n+1;
    else if (n==0)
      return A(m-1,1);

    else
      return A(m-1, A(m, n-1));
    }
```

1 2
(0 2)
=3

Dt
Pg. B+

Basic Algorithms:

A search algorithm is any algorithm which solves one search problem.

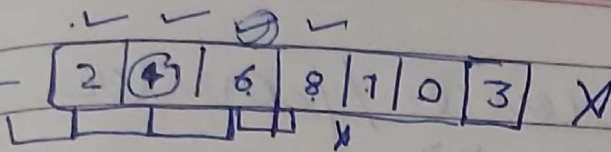
Linear search.

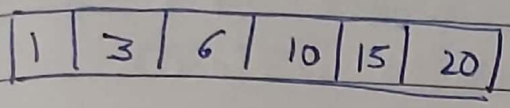
(Sequential Search) → check every element

```
void main()
{
int i, arr[5], num;
printf("Enter the elements");
for(i=0; i<5; i++)
{
scanf("%d", &arr[i]);
}
printf
("%d", &num);
for(i=0; i<5; i++)
{
if(num == Arr(i))
{
printf
break;
}
}
```

,i+1);

(3) Bubble (6) Quick sort





Call by Value.

```
#include<stdio.h>
void main()
{
  int x,y;
  printf("enter the two numbers");
  scanf("%d %d",&x,&y);
  swap(x,y)
  printf("the value of a is %d and
          value of b is %d",a,b);
  getch();
}
void swap(int a,int b)
{
  int t;
  t=a;
  a=b;
  b=t;
}
```

Call by reference / pointer

```
#include <stdio.h>
void main()
{
    int x, y;
    printf("Enter the value of x and y");
    scanf("%d %d", &x, &y)
    swap(&x, &y);
    printf("The value of a is %d
            and the value of b is %d",
                a, b)
    getch()
}
swap(int *a, int *b)
{
    int t;
    t = *a
    *a = *b
    *b = t;
}
```

Dt.
Pg. B+

Structure :-

→ Structure is a user defined data type in C language which allows us to combine data of different types together.

```
#include <stdio.h>
struct emp
{
    int e_serial;
    char e_name[20];
    float e_salary;
};
void main()
{
    struct emp e;
printf("enter the number of serial number");
scanf("%d", &e.serial);

printf("enter the employee name");
scanf("%d", &e.name);

printf("enter the salary of employee");
scanf("%d", &e.salary);

printf("%d", e.e_serial);
printf("%d", e.e_name);
printf("%d", e.e_salary);

getch();
}
```

Dt.
Pg. B+

Linear search

```
void main()
{
 int i, num, Arr[5];
printf ("Enter the number of elements");
 for (i=0 ; i<5 ; i++).
  {
    scanf ("%d", &Arr[i]);
  }
printf (" Enter the number to be searched");
  scanf ("%d" &num);
   for (i=0, i<5, i++)
   {
     if (num == Arr[i])
   {
   printf ("No. found at position
                    %" ,i+1);
  else
    {
   printf (" No. not found ");
  }
    getch();
   }
```