PARISUTHAM
INSTITUTE OF TECHNOLOGY & SCIENCE
AICTE APPROVED / AFFILIATED TO ANNA UNIVERSITY CHENNAI
NAAC 'A' GRADE ACCREDITED

P I T S
Towards Excellence

C PROGRAMMING LAB

Submitted by

D.MAHALAKSHMI, AP/CSE

RAJA, AP/CSE

EX NO:1a Input and Output statement

AIM:

To write a C program to perform Input and Output statement.

ALGORITHM

Step 1: START

Step 2: Declare the variables.

Step 3: Get the values of Student rollNo,name,mark1,mark2,mark3 and mark4.

Step 4: Calculate the total mark and average mark using arithmetic operators.

Step 5: Calculate the percentage.

Step 6: Print the details of student.

Step 7: Stop.

PROGRAM

```
#include <stdio.h>
int main()
{
char name;
int roll;
float mark1,mark 2;
   printf("Enter Student Name: ");
   scanf(“%s”,&name);
   printf(“Enter Roll Number:”);
   scanf("%d", &roll);
   printf("Enter Mark 1: ");
   scanf("%lf", &mark1);
   printf(“Enter Mark 2:”);
   scanf(“%f”,&mark2);
printf(“Student Name = %s”,name);
printf(“Roll Number = %d”,roll);
```

```
printf("Mark1 = %f\n", mark1);
   printf("Mark2 = %lf", mark2);

   return 0;
}
```

OUTPUT

Enter Student Name: Naresh

Enter Roll Number: 1345

Enter Mark1:98

Enter Mark2:89

Student Name= Naresh

Roll Number= 1345

Mark1=98

Mark2=89

RESULT:

Thus the C program executed and output verified.

EX NO:1b OPERATORS

AIM:

To write a C program to perform the operators.

ALGORITHM

Step 1: START

Step 2: Declare integer variables.

Step 3: Set values to the variable.

Step 4: Perform increment and decrement operation.

Step 5: Print values pre and post increment(++) and decrement(--) operators.

Step 6: Perform assignment operation(+=,-=,*=,/=).

Step 7: Print the computed values.

Step 8: Perform relational operation(<,>,==,<=,>=).

Step 9: Print the computed values.

Step 10: Print the logical operation(&&,||,!=).

Step 11: Print the computed values.

Step 12: Stop.

PROGRAM

```
#include<stdio.h>
#include<conio.h>
int main()
{
 int a=10;
 int b=20;
 int c=50;
 int result=1;
 int s=6,t=4;
 int i=5,j=5,k=10 ;
clrscr();
//Program for increment and decrement operator
printf("\nIncrement and Decrement operators");
 printf("\nValue of a before = %d",a);
 ++a;
 printf("\nValue of a after increment = %d",a);
 printf("\nValue of b before = %d",b);
 --b;
 printf("\nValue of a after decrement = %d",b);
 //Program for Assignment operartor
 printf("\nAssignment operators");
printf("\nValue of Result= %d",result,"c=%d",c);
result+=c;
printf("\nValue of Result= %d",result);
result-=c;
printf("\nValue of Result= %d",result);
result*=c;
printf("\nValue of Result=%d",result);
//Program for Relational operator
printf("\nRelational Operators");
printf("\ns=%d,t=%d",s,t);
```

```
printf("\n%d==%d is %d\n",s,t,s==t);
printf("\n%d>%d is %d\n",s,t,s>t);
printf("\n%d<=%d is %d\n",s,t,s<=t);
printf("\n%d!=%d is %d\n",s,t,s!=0);
//Program for Logical operator
printf("\nLogical operators");
printf("\ni=%d,j=%d,k=%d",i,j,k);
printf("\ni=j AND k>j is %d",(i==j)&&(k>j));
printf("\ni=j OR k<j is %d",(i==j)||(k<j));
printf("\ni!=j OR k<j is %d",(i!=j)||(k<j));

getch();
return 0;
}
```

RESULT:

Thus the C program executed and output verified.

EX No.2 Decision Making constructs

AIM:

To write a C program to simulate simple calculator.

ALGORITHM:

Step1: Start.

Step2: Declare the variables.

Step3: Get the operator to be performed from the user.

Step4: Display the operator's name.

Step5: Get the values for two numbers.

Step6: Case'+' – Add two numbers.

Step7: Case'-' – Subtract two numbers.

Step 8: Case'*' – Multiply two numbers.

Step9: Case'/' – Divide two numbers.

Step10: Stop

PROGRAM

```
#include <stdio.h>
int main()
{
   // declare local variables
   char opt;
   int n1, n2;
   float res;
   printf (" Choose an operator(+, -, *, /) to perform the operation in C Calculator \n ");
   scanf ("%c", &opt); // take an operator
   if (opt == '/' )
   {
      printf (" You have selected: Division");
   }
   else if (opt == '*')
   {
```

```
   printf (" You have selected: Multiplication");
 }

else if (opt == '-')
{
   printf (" You have selected: Subtraction");
 }
   else if (opt == '+')
{
   printf (" You have selected: Addition");
 }
printf (" \n Enter the first number: ");
scanf(" %d", &n1); // take fist number
printf (" Enter the second number: ");
scanf (" %d", &n2); // take second number

switch(opt)
{
   case '+':
      res = n1 + n2; // add two numbers
      printf (" Addition of %d and %d is: %.2f", n1, n2, res);
      break;

   case '-':
      res = n1 - n2; // subtract two numbers
      printf (" Subtraction of %d and %d is: %.2f", n1, n2, res);
      break;

   case '*':
      res = n1 * n2; // multiply two numbers
      printf (" Multiplication of %d and %d is: %.2f", n1, n2, res);
      break;

   case '/':
      if (n2 == 0)   // if n2 == 0, take another number
      {
         printf (" \n Divisor cannot be zero. Please enter another value ");
```

```
            scanf ("%d", &n2);
            }
        res = n1 / n2; // divide two numbers
        printf (" Division of %d and %d is: %.2f", n1, n2, res);
        break;
      default:  /* use default to print default message if any condition is not satisfied */
        printf (" Something is wrong!! Please check the options ");
    }
    return 0;
}
```

OUTPUT

```
Choose an operator(+, -, *, /) to perform the operation in C Calculator
+
You have selected: Addition
 Enter the first number: 9
 Enter the second number: 3
Addition of 9+3 is 12
Choose an operator(+, -, *, /) to perform the operation in C Calculator
*
You have selected: Multiplication
 Enter the first number: 9
 Enter the second number: 3
Multiplication of 9*3 is 27
```

RESULT:

Thus the C program executed and output verified.

Ex no.3a For LOOP

AIM:

To write a C program to generate Factorial number.

ALGORITHM:

Step 1: Start

Step 2: Declare variables i, f1,num

Step 3: Initialize the variables, f1=1

Step 4: Enter the input number for factorial

Step 5: From the value of the integer up to 1, multiply each digit and update the final value.

Step 6: The final value at the end of all the multiplication till 1 is the factorial.

Step 7: Stop

PROGRAM

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

  printf("Input the number : ");
  scanf("%d",&num);

  for(i=1;i<=num;i++)
     f=f*i;
  printf("The Factorial of %d is: %d\n",num,f);
}
```

OUTPUT

Input the number : 5

Expected Output :

The Factorial of 5 is: 120

RESULT:

Thus the C program executed and output verified.

EX No:3b While loop

AIM:

To write a C program to print the reverse number using while loop.

ALGORITHM:

Step 1: START

Step 2: Read the value n;

Step 3: Assign n value to temporary variable ‘an’.

Step 4: Check condition while(n>0).

Step 5: Reverse the number

rem=n%10;

rev=rev*10+rem;

n=n/10;

Step 6: Print the reverse number.

Step 7: Stop.

PROGRAM

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
   int rem,rev=0;
   int n,an;
   printf("Enter any Number\n");
   scanf("%d",&n);
   an=n;
   while(n>0)
   {
      rem=n%10;
      rev=rev*10+rem;
      n=n/10;
```

```
    }
    printf("Reverse of %d is %d",an,rev);
    return 0;
}
```

OUTPUT

```
Enter any number: 123
Reverse of 123 is 321
```

RESULT:

Thus the C program executed and output verified.

Ex No:3c do-while loop

AIM:

To write a C program to the values from 10 to 20 using do-while loop.

ALGORITHM:

Step 1: START

Step 2: Assign a=10.

Step 3: Increment ‘a’ value till the condition is a<=20.

Step 4: Print the values.

Step 5: STOP

PROGRAM

```
#include <stdio.h>
 int main () {
   /* local variable definition */
   int a = 10;
   /* do loop execution */
   do {
      printf("value of a: %d\n", a);
      a = a + 1;
   }while( a < 20 );
   return 0;
}
```

OUTPUT:

10 11 12 13 14 15 16 17 18 19 20

RESULT:

Thus the C program executed and output verified.

Ex No 4: MATRIX

AIM:

To write a C program to add two matrices.

ALGORITHM:

Step 1:START

Step 2: Read the elements of Matrix A

Step 3: Read the elements of Matrix B

Step 4: Set a loop up to the row.

Step 5: Set a inner loop up to the column.

Step 6: Add the element of A and B in column wise and store the resule in sum matrix.

Step 7: After execution of two loops, print the values in sum matrix.

Step 8: STOP.

PROGRAM

```
#include<stdio.h>
 int main()
{
        int i, j, rows, columns, a[10][10], b[10][10];
        int arr[10][10];

        printf("\n Please Enter Number of rows and columns  :  ");
        scanf("%d %d", &i, &j);

        printf("\n Please Enter the First Elements\n");
        for(rows = 0; rows < i; rows++)
        {
                for(columns = 0; columns < j; columns++)
        {
                scanf("%d", &a[rows][columns]);
        }
```

```
}
printf("\n Please Enter the Second Elements\n");
for(rows = 0; rows < i; rows++)
{
        for(columns = 0; columns < j; columns++)
{
        scanf("%d", &b[rows][columns]);
}
}

for(rows = 0; rows < i; rows++)
{
        for(columns = 0; columns < j; columns++)
{
        arr[rows][columns] = a[rows][columns] + b[rows][columns];
        }
}

printf("\n The Sum of Two a and b = a + b \n");
for(rows = 0; rows < i; rows++)
{
        for(columns = 0; columns < j; columns++)
{
        printf("%d \t ", arr[rows][columns]);
}
printf("\n");
}
return 0;
```

OUTPUT

Enter number of rows and columns: 2 3

Enter the first elements

10 20 30

40 50 60

Enter the second elements

25 35 45

55 65 75

The Sum of two a and b = a+b

35 55 75

95 115 135

RESULT:

Thus the C program executed and output verified.

EX No.5 SORTING

AIM:

Write a C program to sort an array in an ascending order.

ALGORITHM

STEP 1: START

STEP 2: INITIALIZE arr[] ={5, 2, 8, 7, 1 }..

STEP 3: SET temp =0

STEP 4: length= sizeof(arr)/sizeof(arr[0])

STEP 5: PRINT "Elements of Original Array"

STEP 6: SET i=0. REPEAT STEP 7 and STEP 8 UNTIL i<length

STEP 7: PRINT arr[i]

STEP 8: i=i+1.

STEP 9: SET i=0. REPEAT STEP 10 to STEP UNTIL i<n

STEP 10: SET j=i+1. REPEAT STEP 11 UNTIL j<length

STEP 11: if(arr[i]>arr[j]) then

temp = arr[i]

arr[i]=arr[j]

arr[j]=temp

STEP 12: j=j+1.

STEP 13: i=i+1.

STEP 14: PRINT new line

STEP 15: PRINT "Elements of array sorted in ascending order"

STEP 16: SET i=0. REPEAT STEP 17 and STEP 18 UNTIL i<length

STEP 17: PRINT arr[i]

STEP 18: i=i+1.

STEP 19: RETURN 0.

STEP 20: END.

PROGRAM

```
#include <stdio.h>

int main()
{
   //Initialize array
   int arr[] = {5, 2, 8, 7, 1};
   int temp = 0;
//Calculate length of array arr
   int length = sizeof(arr)/sizeof(arr[0]);
//Displaying elements of original array
   printf("Elements of original array: \n");
   for (int i = 0; i < length; i++) {
      printf("%d ", arr[i]);
   }
   //Sort the array in ascending order
   for (int i = 0; i < length; i++) {
      for (int j = i+1; j < length; j++) {
        if(arr[i] > arr[j]) {
            temp = arr[i];
            arr[i] = arr[j];
            arr[j] = temp;
        }
      }
   }

   printf("\n");
   //Displaying elements of array after sorting
   printf("Elements of array sorted in ascending order: \n");
   for (int i = 0; i < length; i++) {
      printf("%d ", arr[i]);
   }
```

```
    return 0;
}
```

OUTPUT

Elements of original array:

5 2 8 7 1

Elements of array sorted in ascending order:

1 2 5 7 8

RESULT:

Thus the C program executed and output verified.

ExNo:6a FUNCTIONS(Call by value)

AIM

To write a C program to swap two numbers using call by value.

ALGORITHM

Step 1: Start the program.

Step 2: Set a ← 35 and b ← 45

Step 3: Call the function swap(a,b)

Step 3a: Start function.

Step 3b: Assign t ← x

Step 3c: Assign x ← y

Step 3d: Assign y ← t

Step 3e: Print x and y.

Step 3f: End function.

Step 4: Stop the program.

PROGRAM

```
#include <stdio.h>
void swapnum( int var1, int var2 )
{
  int tempnum ;
  /*Copying var1 value into temporary variable */
  tempnum = var1 ;
  /* Copying var2 value into var1*/
  var1 = var2 ;
  /*Copying temporary variable value into var2 */
  var2 = tempnum ;
}
int main( )
{
   int num1 = 35, num2 = 45 ;
   printf("Before swapping: %d, %d", num1, num2);
```

```
    /*calling swap function*/
    swapnum(num1, num2);
    printf("\nAfter swapping: %d, %d", num1, num2);
}
```

OUTPUT:

Before swapping: 35, 45

After swapping: 45, 35

RESULT:

Thus the C program executed and output verified.

ExNo:6b FUNCTIONS(Call by reference)

AIM

To write a C program to swap two numbers using call by reference.

ALGORITHM

Step 1: Start the program.

Step 2: Set a ← 35 and b ← 45

Step 3: Call the function swap(&a,&b)

Step 3a: Start function

Step 3b: Assign t ← *x

Step 3c: Assign *x ← *y

Step 3d: Assign *y ← t

Step 3e: End function

Step 4: Print x and y.

Step 5: Stop the program.

PROGRAM

```
#include<stdio.h>
void swapnum ( int *var1, int *var2 )
{
  int tempnum ;
  tempnum = *var1 ;
  *var1 = *var2 ;
  *var2 = tempnum ;
}
int main( )
{
  int num1 = 35, num2 = 45 ;
  printf("Before swapping:");
  printf("\nnum1 value is %d", num1);
  printf("\nnum2 value is %d", num2);
```

```
  /*calling swap function*/
  swapnum( &num1, &num2 );

  printf("\nAfter swapping:");
  printf("\nnum1 value is %d", num1);
  printf("\nnum2 value is %d", num2);
  return 0;
}
```

OUTPUT

Before swapping:

num1 value is 35

num2 value is 45

After swapping:

num1 value is 45

num2 value is 35

RESULT:

Thus the C program executed and output verified.

ExNo:7 RECURSION

AIM

TO write a C program to print sum of natural numbers using recursion.

ALGORITHM

Step 1: START

Step 2: Get n values of natural numbers.

Step 3: Call function findsum().

Step 4: check n<=1

Return sum of the series

N+findsum(n-1)

Step 5: Print the sum of the series.

Step 6: Stop.

PROGRAM

```
#include <stdio.h>
// Recursive function to find the sum of first n natural numbers
int findSum(int n)
{
 if (n<=1)
 {
 return n;
 }
 else
 {
 return n + findSum(n-1);
 }
}
// Driver code
int main()
{
```

```
 int n1 = 5, n2 = 7, n3 = 6;
 printf("n1: %d \n", n1);
 printf("n2: %d \n", n2);
 printf("n3: %d \n", n3);
 printf("Sum of first %d natural numbers: %d \n", n1, findSum(n1));
 printf("Sum of first %d natural numbers: %d \n", n2, findSum(n2));
 printf("Sum of first %d natural numbers: %d \n", n3, findSum(n3));
return 0;
}
```

OUTPUT

```
n1: 5
n2: 7
n3: 6
Sum of first 5 natural numbers: 15
Sum of first 7 natural numbers: 28
Sum of first 6 natural numbers: 21
```

RESULT:

Thus the C program executed and output verified.

ExNo:8 POINTERS

AIM

To write a C program to calculate area of a triangle using pointers and functions.

ALGORITHM

Step 1: START

Step 2: Read base, height

Step 3: Calculate area = 0.5*base*height

Step 4: Print area

Step 5: Stop

PROGRAM

```
#include<stdio.h>
void read(float *b, float *h);
void cal_area(float *b, float *h);
float *a;

int main()
{
Float base, height, area;
Read(&base, &height);
Cal_area(&base, &height, &area);
Printf("Area of triangle with base %f and %f = %.2",base, height, area);
Return 0;
Void read(float *b, float *h)
{
Printf("Enter the base of the triangle:");
Scanf("%f",b);
Printf("Enter the height of the triangle:");
Scanf("%f",h);
}
```

```
Void cal_area(float *b,float *h,float *a)
{
*a=0.5*(*b)*(*h);
}
```

OUTPUT

```
Enter the base of the triangle : 10
Enter the height of the triangle: 5
Area of the triangle with base 10.0 and height 5.0 = 25
```

RESULT:

Thus the C program executed and output verified.

ExNo:9 STRUCTURES

AIM

To write a C program using structures.

ALGORITHM

Step 1: Start

Step 2: Declare student structure

Step 3: Read student name, roll, marks

Step 4: Print student name, roll, marks

Step 5: Stop

PROGRAM

```
#include <stdio.h>
struct student {
    char firstName[50];
    int roll;
    float marks;
} s[5];

int main() {
    int i;
    printf("Enter information of students:\n");

    // storing information
    for (i = 0; i < 5; ++i) {
        s[i].roll = i + 1;
        printf("\nFor roll number%d,\n", s[i].roll);
        printf("Enter first name: ");
        scanf("%s", s[i].firstName);
        printf("Enter marks: ");
        scanf("%f", &s[i].marks);
    }
```

```
    printf("Displaying Information:\n\n");

    // displaying information
    for (i = 0; i < 5; ++i) {
        printf("\nRoll number: %d\n", i + 1);
        printf("First name: ");
        puts(s[i].firstName);
        printf("Marks: %.1f", s[i].marks);
        printf("\n");
    }
    return 0;
}
```

OUTPUT

```
Enter information of students:

For roll number1,
Enter name: Tom
Enter marks: 98

For roll number2,
Enter name: Jerry
Enter marks: 89

Displaying Information:

Roll number: 1
Name: Tom
Marks: 98
```

RESULT:

Thus the C program executed and output verified.

ExNo:10 FILES

AIM

To write a C Program to add, delete ,display ,Search and exit options for telephone details of an individual into a telephone directory using random access file.

ALGORITHM

1. Start.
2. Declare variables, File pointer and phonebook structures.
3. Create menu options.
4. Read the option.
5. Develop procedures for each option.
6. Call the procedure (Add, delete ,display ,Search and exit)for user chosen option.
7. Display the message for operations performed.
8. Stop

PROGRAM

#include <stdio.h> #include <stdlib.h> #include <string.h>

typedef struct Phonebook_Contacts

{

char FirstName[20]; char LastName[20]; char PhoneNumber[20];

} phone;

void AddEntry(phone *); void DeleteEntry(phone *); void PrintEntry(phone *);

void SearchForNumber(phone *);

int counter = 0; char FileName[256]; FILE *pRead;

FILE *pWrite;

int main (void)

{

phone *phonebook;

```
phonebook = (phone*) malloc(sizeof(phone)*100); int iSelection = 0;


if (phonebook == NULL)

{


printf("Out of Memory. The program will now exit"); return 1;

}

else {}


do

{

printf("\n\t\t\tPhonebook Menu"); printf("\n\n\t(1)\tAdd Friend"); printf("\n\t(2)\tDelete Friend"); printf("\n\t(3)\tDisplay Phonebook Entries"); printf("\n\t(4)\tSearch for Phone Number"); printf("\n\t(5)\tExit Phonebook"); printf("\n\nWhat would you like to do? "); scanf("%d", &iSelection);

if (iSelection == 1)

{

AddEntry(phonebook);

}

if (iSelection == 2)

{

DeleteEntry(phonebook);

}

if (iSelection == 3)

{

PrintEntry(phonebook);

}

if (iSelection == 4)

{

SearchForNumber(phonebook);

}


if (iSelection == 5)
```

```
{
printf("\nYou have chosen to exit the Phonebook.\n"); return 0;
}
} while (iSelection <= 4);
}
void AddEntry (phone * phonebook)
{
pWrite = fopen("phonebook_contacts.dat", "a"); if ( pWrite == NULL )
{
perror("The following error occurred "); exit(EXIT_FAILURE);
}
else
{
counter++;
realloc(phonebook, sizeof(phone));

printf("\nFirst Name: ");
scanf("%s", phonebook[counter-1].FirstName); printf("Last Name: ");
scanf("%s", phonebook[counter-1].LastName); printf("Phone Number (XXX-XXX-XXXX):
"); scanf("%s", phonebook[counter-1].PhoneNumber);
printf("\n\tFriend successfully added to Phonebook\n");

fprintf(pWrite, "%s\t%s\t%s\n", phonebook[counter-1].FirstName, phonebook[counter-
1].LastName, phonebook[counter-1].PhoneNumber);
fclose(pWrite);
}
}

void DeleteEntry (phone * phonebook)
{
int x = 0;
int i = 0;
char deleteFirstName[20]; // char deleteLastName[20];
```

```
printf("\nFirst name: "); scanf("%s", deleteFirstName); printf("Last name: "); scanf("%s",
deleteLastName);


for (x = 0; x < counter; x++)

{

if (strcmp(deleteFirstName, phonebook[x].FirstName) == 0)

{

if (strcmp(deleteLastName, phonebook[x].LastName) == 0)

{

for ( i = x; i < counter - 1; i++ )

{

strcpy(phonebook[i].FirstName, phonebook[i+1].FirstName);
strcpy(phonebook[i].LastName, phonebook[i+1].LastName);
strcpy(phonebook[i].PhoneNumber, phonebook[i+1].PhoneNumber);

}

printf("Record deleted from the phonebook.\n\n");

--counter; return;

}

}

}

printf("That contact was not found, please try again.");

}


void PrintEntry (phone * phonebook)

{

int x = 0;


printf("\nPhonebook Entries:\n\n ");

pRead = fopen("phonebook_contacts.dat", "r"); if ( pRead == NULL)

{

perror("The following error occurred: "); exit(EXIT_FAILURE);

}
```

```
else
{
for( x = 0; x < counter; x++)
{
printf("\n(%d)\n", x+1);
printf("Name: %s %s\n", phonebook[x].FirstName, phonebook[x].LastName);
printf("Number: %s\n", phonebook[x].PhoneNumber);
}
}
fclose(pRead);
}
void SearchForNumber (phone * phonebook)
{
int x = 0;
char TempFirstName[20]; char TempLastName[20];

printf("\nPlease type the name of the friend you wish to find a number for."); printf("\n\nFirst Name: ");
scanf("%s", TempFirstName); printf("Last Name: "); scanf("%s", TempLastName); for (x = 0; x < counter; x++)
{
if (strcmp(TempFirstName, phonebook[x].FirstName) == 0)
{
if (strcmp(TempLastName, phonebook[x].LastName) == 0)
{

printf("\n%s %s's phone number is %s\n", phonebook[x].FirstName, phonebook[x].LastName, phonebook[x].PhoneNumber);
}
}
}
}
```

OUTPUT

Phonebook Menu

(1) Add Friend

(2) Delete Friend"

(3) Display Phonebook Entries

(4) Search for Phone Number

(5) Exit Phonebook

What would you like to do? 1 First Name: Ram

Last Name: Mohan

Phone Number (XXX-XXX-XXXX): 717-675-0909

Friend successfully added to Phonebook

Phonebook Menu

(1) Add Friend

(2) Delete Friend"

(3) Display Phonebook Entries

(4) Search for Phone Number

(5) Exit Phonebook

What would you like to do? 5

You have chosen to exit the Phonebook.

RESULT:

Thus the C program executed and output verified.