

VELAMMAL ENGINEERING COLLEGE, CHENNAI – 66
(An Autonomous Institution, Affiliated to Anna University, Chennai)

Course code	19CS102L	Semester	I			
Category	Engineering Science Course(ESC)		L	T	P	C
Course Title	C Programming Lab		0	0	4	2

LIST OF EXPERIMENTS

1. Simple C Programs using statements and expressions
2. Solving problems using decision making and looping.
3. Programs using single and multidimensional arrays.
4. Solving simple problems using built-in and user defined String functions.
5. Execute searching and sorting of strings.
6. Programs using built-in and user defined functions
7. Programs using Recursive Function and conversion
8. Simple programs to understand Storage class in C
9. Simple programs using pointers
10. Programs using Dynamic Memory allocation
11. Use structures and unions for solving various applications.
12. Implement various operations of Sequential file.
13. Execute the functions of Random access file.
14. **Mini Project:** A batch of 3 students can select a topic on their interest and should involve all the concepts of C programming to the maximum extent in developing the application.

1. SIMPLE C PROGRAMS USING STATEMENTS AND EXPRESSIONS

EX.NO.:

DATE:

DATA TYPES

AIM:

To write a C program using data type.

ALGORITHM:

- i) Start the program
- ii) Initialize the value of x as integer, y as float and z as character data type.
- iii) Print the value of the variable x, y, z.
- iv) Stop the program.

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int x=2;
    float y=2.2;
    char z='c';
    clrscr();
    printf("Integer=%d\nFloat=%f\nCharacter=%c",x,y,z);
    getch();
}
```

OUTPUT:

Integer=2
Float=2.2
Character=c

RESULT:

Thus a C program was written using data types and executed successfully.

EX.NO.:

DATE:

EXPRESSION EVALUATION

EVALUATE THE FOLLOWING EXPRESSION $a=3x^2+2x+1$.

AIM:

To write a C program to evaluate the following expression $a=3x^2+2x+1$.

ALGORITHM:

- i) Start the program
- ii) Declare the data type of the variables (i.e) a, x as integer.
- iii) Get the value of the variable 'x' as input using scanf() function.
- iv) Expression is written into executable form.
- v) Print the output variable 'a' using the printf() function.
- vi) Stop the program.

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int x,a;
    clrscr();
    printf("Enter the value of x: ");
    scanf("%d",&x);
    a=3*x*x+2*x+1;
    printf("The value of 'a'=%d",a);
    getch();
}
```

OUTPUT:

Enter the value of x: 7

The value of 'a'=162

RESULT:

Thus a C program was written to evaluate the expression and executed successfully.

EX.NO.:

DATE:

AREA AND CIRCUMFERENCE OF A CIRCLE

AIM: Program to calculate area and circumference of a circle

ALGORITHM:

- Step 1: Start
- Step 2: Read the value radius
- Step 3: Compute $\text{area} = 3.14 * \text{radius} * \text{radius}$
- Step 4: Compute $\text{circumference} = 2 * 3.14 * \text{radius}$
- Step 5: Display area and circumference
- Step 6: Stop

PROGRAM CODE

```
#include<stdio.h>
#include<conio.h>
void main()
{
    float radius, area, circumference;
    clrscr();
    printf("\nEnter the Radius to find the Area and Circumference of the Circle\n");
    scanf("%f", &radius);
    area = 3.14 * radius * radius;
    circumference = 2 * 3.14 * radius;
    printf("\nThe Area of The Circle is %f\nThe Circumference of the Circle is %f", area,
    circumference);
    getch();
}
```

OUTPUT:

Enter the Radius to find the Area and Circumference of the Circle

12

The Area of the Circle is 452.160004sq. units

The Circumference of the Circle is 75.360001units

RESULT:

Thus the area and the circumference of a circle is calculated and the output is verified.

EX.NO.:

DATE:

FAHRENHEIT TO CELSIUS CONVERSION

AIM: Program to convert Fahrenheit to Celsius

ALGORITHM:

STEP 1: Start
STEP 2: Read a value from user and store it in fahrenheit.
STEP 3: Compute $X = \text{fahrenheit} - 32$
STEP 4: Compute $\text{celsius} = \text{fahrenheit} / 1.8$
STEP 5: Display celsius
STEP 6: Stop

PROGRAM CODE:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    float fahrenheit, celsius,x;
    clrscr();
    printf("Enter the degrees in fahrenheit:\n");
    scanf("%f", &fahrenheit);
    x=fahrenheit-32;
    celsius = x/1.8;
    printf("\nThe Celsius Value is %6.2f C", celsius);
    getch();
}
```

OUTPUT:

Enter the degrees in fahrenheit:
98.4
The Celsius Value is 36.89 C

RESULT:

Thus the Degree is converted from Fahrenheit to Celsius and the output is verified.

2. SOLVING PROBLEMS USING DECISION MAKING AND LOOPING

EX.NO.:

DATE:

LARGEST OF THREE NUMBERS

AIM:

To write a C program to find the largest of three numbers using 'if-else' conditional statement.

ALGORITHM:

- i) Start the program.
- ii) Declare the variables.
- iii) Read any three values a, b, c.
- iv) Find the biggest number among the three numbers and store it in 'big'.
- v) Display 'big'.
- vi) Stop the program.

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int a,b,c;
    printf("Enter the three numbers: ");
    scanf("%d%d%d",&a,&b,&c);
    if(a>b&&a>c)
        printf("The biggest number is: %d",a);
    else if(b>a&&b>c)
        printf("The biggest number is: %d",b);
    else
        printf("The biggest number is: %d",c);
}
```

OUTPUT:

Enter the three numbers: 7 5 8

The biggest number is: 8

RESULT:

Thus a C program was written to find the biggest of three numbers using if-else conditional statements and executed successfully.

EX.NO.:

DATE:

GRADE OF A STUDENT

AIM: Program to compute the Average for the given marks and print Grade.

ALGORITHM:

STEP 1: Start
STEP 2: Read rno, name, m1, m2, m3, m4, m5, m6
STEP 3: Compute $avg = (m1 + m2 + m3 + m4 + m5 + m6) / 6$
STEP 4: If $avg == 100$ print grade is S
STEP 5: Else if $avg > 90$ and $avg < 100$ print grade is A
STEP 6: Else if $avg > 75$ and $avg \leq 90$ print grade is B
STEP 7: Else if $avg > 60$ and $avg \leq 75$ print grade is C
STEP 8: Else if $avg \geq 50$ and $avg < 60$ print grade is D
STEP 9: Else print fail
STEP 10: Stop

PROGRAM CODING:

```
#include<stdio.h>
#include<string.h>
#include<conio.h>
void main()
{
    int rno, i;
    float m1, m2, m3, m4, m5, m6, avg;
    char name[100], grade, c;
    clrscr();
    printf("\nEnter your Roll Number\n");
    scanf("%d", &rno);
    printf("\nEnter your Name\n");
    scanf("%s", name);
    printf("Hi %s, \nEnter the marks in six subjects\n", name);
    scanf("%f %f %f %f %f %f", &m1, &m2, &m3, &m4, &m5, &m6);
    avg = (m1+m2+m3+m4+m5+m6)/6;
    if(avg==100.0f)
        grade = 'S';
    else if(avg>90.0f && avg<100.0f)
        grade = 'A';
    else if(avg>75.0f && avg<=90.0f)
        grade = 'B';
    else if(avg>60.0f && avg<=75.0f)
        grade = 'C';
```

```

        else if(avg>=50.0f && avg<=60.0f)
            grade = 'D';
        printf("\n\nRollNo\tName\tM1\tM2\tM3\tM4\tM5\tM6\tAVG\tGrade\n");
        printf("%d\t%s\t%03.2f\t%03.2f\t%03.2f\t%03.2f\t%03.2f\t%03.2f\t%03.2f\t%c",
rollno, name, m1, m2, m3, m4, m5, m6, avg, grade);
        getch();
    }

```

OUTPUT:

Enter your Roll Number

1

Enter your Name without space

STUDENT

Hi STUDENT,

Enter the marks in six subjects

78 89 98 99 100 65

RollNo	Name	M1	M2	M3	M4	M5	M6	AVG	Grade
1	STUDENT	78.00	89.00	98.00	99.00	100.00	65.00	88.17	B

RESULT:

Thus we write the program for print the Grade of the Student.

EX.NO.:

DATE:

ARMSTRONG NUMBER CHECKING

AIM: Program to check whether a number is Armstrong or not

ALGORITHM:

STEP 1: Start

STEP 2: Read a

STEP 3: Initialize e=a

STEP 4: Repeat the following statements until a>0

(i) $b = a \% 10$

(ii) $c = c + b * b * b$

(iii) $a = a / 10$

STEP 5: check whether c is equal to e

(a) If so print the number is amstrong.

(b) print the number is not amstrong.

STEP 6 : Stop

PROGRAM CODE:

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

```
#include<conio.h>
```

```
void main()
```

```
{
```

```
    long int i, arm, a, no;
```

```
    clrscr();printf("\nEnter a Number\n");
```

```
    scanf("%ld", &a); arm = 0;
```

```
    e = a;
```

```
    while(a>0)
```

```
    {
```

```
        b = a % 10;
```

```
        c = c + b * b * b;
```

```
        a = a / 10;
```

```
    }
```

```
    if(e==arm)
```

```
        printf("\nThe Given Number is an Armstrong Number");
```

```
    else
```

```
        printf("\nThe Given Number is NOT an Armstrong Number"); getch();
```

```
}
```

OUTPUT:

Enter a Number: 153

The Given Number is an Armstrong Number

RESULT: Thus the number is checked whether it is Armstrong or not and the output is verified.

3. PROGRAMS USING SINGLE AND MULTIDIMENSIONAL ARRAYS

EX. NO.:

DATE:

SORTING N NUMBERS

AIM:

To write a C program to sort the number in ascending order using arrays.

ALGORITHM:

- i) Start the program.
- ii) Declare the array and variables.
- iii) Enter the size of the array.
- iv) Enter the elements of the array.
- v) Set a 'for' loop up to the array size minus one.
- vi) Set an inner 'for' loop up to the size of the array.
- vii) If the current array element is greater than the next array element, then exchange their position.
- viii) If not, then go back to the inner loop. After the execution of the inner loop, the outer loop is executed.
- ix) Print the ascending order of the given array.
- x) Stop the program.

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int n,i,j,temp,x[50];
    clrscr();
    printf("Enter the total number of elements in array: ");
    scanf("%d",&n);
    printf("Enter the elements:\n");
    for(i=0;i<n;i++)
        scanf("%d",&x[i]);
    for(i=0;i<n-1;i++)
        for(j=0;j<n;j++)
            if(x[i]>x[j])
```

```

        {
            temp=x[i];
            x[i]=x[j];
            x[j]=temp;
        }
    printf("\nNumbers in ascending order:\n");
    for(i=0;i<n;i++)
        printf("\t%d",x[i]);
    getch();
}

```

OUTPUT:

Enter the total number of elements in array: 8

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

Numbers in ascending order:

1 2 3 5 6 7 8 9

RESULT:

Thus a C program was written to sort the number in ascending order using arrays and executed successfully.

EX.NO.

DATE:

ADDITION OF TWO MATRICES

AIM:

To write the C program to add the two dimensional matrices.

ALGORITHM:

- Step 1: Start
- Step 2: Setup loop For i=0 to 3
- Step 3: Setup loop for j=0 to 3
- Step 4: Read a[i][j]
- Step 5: Setup loop For i=0 to 3
- Step 6: Setup loop for j=0 to 3
- Step 7: Read b[i][j]
- Step 8: Setup loop For i=0 to 3
- Step 9: Setup loop for j=0 to 3
- Step 10: Add a[i][j], b[i][j] to c[i][j]
- Step 11: Print c[i][j]
- Step 12: Stop

PROGRAM CODING:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int a[3][3], b[3][3], c[3][3], i, j, k;
    clrscr();
    printf("\nEnter the elements of A\n");
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            scanf("%d", &a[i][j]);
        }
    }
    printf("\n\nEnter the Elements of Matrix B\n");
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
```

```

scanf("%d", &b[i][j]);
    }
}
for(i=0;i<3;i++)
{
    for(j=0;j<3;j++)
    {
        c[i][j]= a[i][j] + b[i][j];
    }
}
printf("\n\nThe sum of the two given matrices is\n");
for(i=0;i<3;i++)
{
    for(j=0;j<3;j++)
    {
        printf("%d\t", c[i][j]);
    }
    printf("\n");
}
getch();
}

```

OUTPUT:

Enter the elements of A

```

1   2   3
4   5   6
7   8   9

```

Enter the Elements of Matrix B

```

9   8   7
6   5   4
3   2   1

```

The sum of the two given matrices is

```

10  10  10
10  10  10
10  10  10

```

RESULT:

Thus we write a C program to add the two dimensional matrices.

EX.NO.:

DATE:

PRODUCT OF TWO MATRICES

AIM:

To write the C program to find the Product of Matrices.

ALGORITHM:

- Step 1: Start
- Step 2: Setup loop For i=0 to 3
- Step 3: Setup loop for j=0 to 3
- Step 4: Read a[i][j]
- Step 5: Setup loop For i=0 to 3
- Step 6: Setup loop for j=0 to 3
- Step 7: Read b[i][j]
- Step 8: Setup loop For i=0 to 3
- Step 9: Setup loop for j=0 to 3
- Step 10: Let c[i][j] = 0
- Step 11: Setup loop for k=0 to 3
- Step 12: Let c[i][j] = c[i][j] + a[i][k] * b[k][j]
- Step 13: Setup loop for i=0 to 3
- Step 14: Setup loop for j=0 to 3
- Step 15: Print matrix c[i][j]
- Step 16: Stop

PROGRAM CODING:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int a[3][3], b[3][3], c[3][3], i, j, k;
    printf("\nEnter the elements of A\n");
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            scanf("%d", &a[i][j]);
        }
    }
    printf("\n\nEnter the Elements of Matrix B\n");
    for(i=0;i<3;i++)
    {
```

```

        for(j=0;j<3;j++)
        {
            scanf("%d", &b[i][j]);
        }
    }
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            c[i][j]=0;
            for(k=0;k<3;k++)
            {
                c[i][j] = c[i][j] + a[i][k] * b[k][j];
            }
        }
    }
    printf("\nThe Product of the Two Matrices are\n\n");
    for(i=0;i<3;i++)
    {
        for(j=0;j<3;j++)
        {
            printf("%d\t", c[i][j]);
        }
        printf("\n");
    }
    getch();
}

```

OUTPUT:

```

Enter the elements of A
1   2   3 4   5   6 7   8   9
Enter the Elements of Matrix B
9   8   7 6   5   4 3   2   1
The Product of the Two Matrices are
30   24   18
84   69   54
138  114   90

```

RESULT:

Thus we write the C program to find the Product of two matrices.

4. SOLVING SIMPLE PROBLEMS USING BUILT-IN AND USER DEFINED STRING FUNCTIONS

STRING OPERATIONS SUCH AS COMPARE, CONCATENATE, STRING LENGTH

EX.NO.:

DATE:

AIM:

To understand the function parameter passing techniques and implement the string operation using the same technique.

ALGORITHM:

Step 1. Start

Step 2. Input the two strings] read s1, s2

Step 3. perform two string compare using a function] Call the function Str2cmp(s1,s2)

Step 4. perform two string concatenation] Call the function Str2cat(s1,s2)

Step 5. Find the length of two string using a function] Call the function length(s1), length(s2)

Step 6. Stop

PROGRAM :

```
#include<string.h>
#include<stdlib.h>
#include<stdio.h>
void com2str(char str1[100], char str2[100]);
void concat2str(char str1[100], char str2[100]);
void length(char str[100]);
int main()
{
    int ch;
    char s1[100], s2[100];
    printf("Enter a first string :");
    scanf("%s",s1);
    printf("Enter a second string :");
    scanf("%s",s2);
    printf("The length of first string :");
```

```

        length(s1);
        printf("\nThe length of second string :");
        length(s2);
        printf("\nResults of two strings compare :");
        com2str(s1,s2);
        printf("Results After concatenation of two strings :");
        concat2str(s1,s2);
        getchar();
    }
void com2str(char str1[100], char str2[100])
{
    if (strcmp(str1,str2)==0)
        printf("Both the strings are equal      \n");
    else
        printf("Both the strings are not equal    \n");
}
void concat2str(char str1[100], char str2[100])
{
    strcat(str1,str2);
    printf("%s\n",str1);
}
void length(char str1[100])
{
    int x;
    x=strlen(str1);
    printf("%s = %d ",str1, x);
}

```

OUTPUT:

Enter a first string: SCI

Enter a second string: ENCE

The length of first string: SCI = 3

The length of second string: ENCE = 4

Results of two strings compare: Both the strings are not equal Results After concatenation of two strings: SCIENCE

RESULT:

Thus a C program was written to perform string operation using the same technique is executed.

LOWER CASE TO UPPER CASE

EX.NO.:

DATE:

AIM:

To write a c program to convert lower case to upper case.

ALGORITHM:

Step 1: Start

Step 2: Input a string

Step 3: Convert string to uppercase

Strupr function

Step 3.1: repeat Steps 3.1.1 to 3.1.3 till *s != '\0'

Step 3.1.1: Assign d=*s-'a'

Step 3.1.2: if d>=0 and d<26 then *s='A'+d

Step 3.1.3: s=s+1

Step 4: Display String

Step 5: End

PROGRAM :

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

```
void strupr(char *s)
```

```
{
```

```
    int d;
```

```
    while(*s!='\0')
```

```
    {
```

```
        d=*s-'a';
```

```
        if((d>=0)&&(d<26))
```

```
            *s='A'+d;
```

```
            s++;
```

```
    }
```

```
}
```

```
int main()
```

```
{
```

```
    char a[10];
```

```
    printf("Enter the string:\n");
```

```
scanf("%s",a);  
strupr(a);  
printf("\nThe string in uppercase is: %s\n",a);  
return 0;  
}
```

OUTPUT:

Enter the string: sam
The string in uppercase is: SAM

RESULT:

The program to convert the upper case to lower case is executed successfully using C.

5. EXECUTE SEARCHING AND SORTING OF STRINGS

EX.NO.:

DATE:

SEARCHING THE GIVEN STRINGS

AIM:

To write a C program to search the string from a given set of Strings

ALGORITHM:

1. Start
2. Declare two character array s1&s2.
3. Initialize the s1 with the given string -“Beauty is in the eye of the beholder”
4. Initialize the s2 with the string to be searched- “the”
5. Declare three integer type variables and assign “0” to them.
6. Verify each character of the search string s2 with the given string s1 using while loop.
7. **If** sequence of characters matching with the length of searched string then the search string is found from the given string. Print the string is found, if not print otherwise.
8. Stop

PROGRAM:

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

```
#include <string.h>
```

```
int main() {
    char s1[] = "Beauty is in the eye of the beholder";
    char s2[] = "the";

    int n = 0;
    int m = 0;
    int times = 0;
    int len = strlen(s2);    // contains the length of search string

    while(s1[n] != '\0') {

        if(s1[n] == s2[m]) {    // if first character of search string matches

            // keep on searching

            while(s1[n] == s2[m] && s1[n] != '\0') {
                n++;
                m++;
            }
        }
    }
}
```

```

// if we sequence of characters matching with the length of searched string
if(m == len && (s1[n] == ' ' || s1[n] == '\0')) {

    // BINGO!! we find our search string.
    times++;
}
} else {          // if first character of search string DOES NOT match
while(s1[n] != ' ') {    // Skip to next word
    n++;
    if(s1[n] == '\0')
        break;
    }
}

n++;
m=0; // reset the counter to start from first character of the search string.
}

if(times > 0) {
    printf("%s' appears %d time(s)\n", s2, times);
} else {
    printf("%s' does not appear in the sentence.\n", s2);
}

return 0;
}

```

OUTPUT:

'the' appears 2 time(s)

RESULT:

Thus the program was successfully run and verified.

EX.NO.:

DATE:

SORTING THE GIVEN STRINGS

AIM:

To write a C program to sort the string in ascending order.

ALGORITHM:

1. Create a 2D character array to store names of some fixed size.
2. Take names as input from users using for loop.
3. Now, sort this array of names using Selection sort.
4. Make a nested for loop, where the upper loop extracts each name and inner loop compares this name by the rest of the names below it.
5. After executing both the loops, rearranging all the names, finally an array of alphabetically sorted array will be obtained.

PROGRAM:

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

void main()
{
    char name[10][8], tname[10][8], temp[8];
    int i, j, n;
    printf("Enter the value of n \n");
    scanf("%d", &n);
    printf("Enter %d names n \n", n);
    for (i = 0; i < n; i++)
    {
        scanf("%s", name[i]);
        strcpy(tname[i], name[i]);
    }
}
```

```

for (i = 0; i < n - 1 ; i++)
{
    for (j = i + 1; j < n; j++)
    {
        if (strcmp(name[i], name[j]) > 0)
        {
            strcpy(temp, name[i]);
            strcpy(name[i], name[j]);
            strcpy(name[j], temp);
        }
    }
}

printf("Input Names \t Sorted names\n");
for (i = 0; i < n; i++)
{
    printf("%s\t\t%s\n", tname[i], name[i]);
}
}

```

OUTPUT:

Enter the value of n: 3

Enter 3 names n ram sam cam

```

-----
Input Namest      Sorted names
-----

```

```

ram              cam
sam              ram
cam              sam
-----

```

RESULT:

Thus a C program was written to sort the strings are executed successfully.

6. PROGRAMS USING BUILT-IN AND USER DEFINED FUNCTIONS

EX.NO.:

DATE:

AIM:

To write a program in C to explain built-in and user defined functions.

ALGORITHM:

1. Declare the function with appropriate prototype before calling and defining the function.
2. Function name should be meaningful and descriptive.
3. Keep same argument data type and return data type in Declaration and Defining the function.
4. Pass the same data type arguments which are provided in Declaration and Definition.
5. The arguments/parameters which are used while calling the functions are known as Actual Arguments/Parameters and Arguments/Parameters which are used in the definition of the function are known as Local (Format) Arguments/Parameters.

PROGRAM:

```
#include<stdio.h>
int  sumTwoNum(int,int);
float averageTwoNum(int,int);
int main()
{
    int number1,number2;
    int sum;
    float avg;
    printf("Enter the first integer number: ");
    scanf("%d",&number1);
    printf("Enter the second integer number: ");
    scanf("%d",&number2);
    sum=sumTwoNum(number1,number2);
    avg=averageTwoNum(number1,number2);
    printf("Number1: %d, Number2: %d\n",number1,number2);
    printf("Sum: %d, Average: %f\n",sum,avg);
    return 0;
}
int sumTwoNum(int x,int y)
{
    int sum;
    sum=x+y;
    return sum;
}
```

```
float averageTwoNum(int x,int y)
{
    float average;
    return ((float)(x)+(float)(y))/2;
}
```

OUTPUT:

Enter the first integer number: 100
Enter the second integer number: 201
Number1: 100, Number2: 201
Sum: 301, Average: 150.500000

RESULT:

Thus the C program is successfully run and verified using functions.

7. PROGRAMS USING RECURSIVE FUNCTION AND CONVERSION

EX.NO.:

DATE:

FACTORIAL USING RECURSION

AIM:

To write a program in C to find the factorial of the given number.

ALGORITHM:

Step 1: start

Step 2: Read input value n

Step 3: call the function fact(n)

Step 4: Is n=0 return 1 else goto step 5

Step 5: return(n*fact(n-1))

Step 6: Stop

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int n;
    long int fact();clrscr();
    printf("enter the number whose factorial is to be found");
    scanf("%d",&n);
    printf("the factorial of %d is: %d \n",n,fact(n));
    getch();
}
```

```
long int fact(n)
{
int n;
if(n==0)
return(1);
else
return(n*fact(n-1));
}
```

OUTPUT:

enter the number whose factorial is to be found:6

The factorial of 6 is: 720

RESULT:

Thus the factorial of the given number is found using C functions.

8. STORAGE CLASSES

EX.NO.:

DATE:

AIM:

To understand the concept of storage classes in C Program.

ALGORITHM:

- Step1: Start the Program.
- Step2: Declare different storage classes variables
- Step3: Declare static variable inside a function
- Step4: Execute the function 3 times to understand the behavior of static variable
- Step5: Access the extern variable both in function and main.
- Step5: Display the output
- Step6: Stop the Program.

PROGRAM:

```
#include<stdio.h>
#include<conio.h>
extern int i=10; // extern or global variable
void fun()
{
    static int k=0;
    int z=0;
    k++;
    z++;
    printf("k=%d z=%d",k,z);
    i=i+10;
    printf("i=%d",i); //global variable accessible here
}
void main()
{
    int j=20; // auto variable
    fun();
    fun();
    fun();
    printf(" j=%d",j);
    printf("i=%d",i)
}
```

Output:

```
k=1 z=1  
i=20  
k=2 z=1  
i=30  
k=3 z=1  
i=40  
j=20  
i=40
```

RESULT:

Thus the program was successfully run and verified.

9. SIMPLE PROGRAMS USING POINTERS

EX.NO:

DATE:

CALL BY REFERENCE

AIM:

To find the value by call by reference method.

ALGORITHM:

Step 1: Start

Step 2: Declare and initialise i and j

Step 3: Display i and j values

Step 4: Call the function interchange (&i,&j)

Step 5: Interchange i and j values using temporary variable t

Step 6: Display i and j values and return to the main function

Step 7: Again display the values of i and j in main function

Step 8: Stop

PROGRAM:

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

```
#include<conio.h>
```

```
void main()
```

```
{
```

```
int i=5,j=10;
```

```
clrscr();
```

```
printf("i and j value before interchanging :%d %d \n",i,j);
```

```
interchange(&i,&j);
```

```
printf("i and j values after interchange in main() :%d %d \n",i,j);
```

```
getch();
```

```
}
```

```
void interchange(int *a, int*b)
```

```
{  
int t;  
t=*a; *a=*b; *b=t;  
printf("i and j valuesafter interchanging  inside the function");  
}
```

OUTPUT:

i and j values before interchange :5 10

i and j values after interchanging inside the function :10 5

i and j values after interchange in the main() :10 5

RESULT:

Thus the call by reference method is proved.

10. DYNAMIC MEMORY ALLOCATION

EX.NO.:

DATE:

AIM:

To implement dynamic memory allocation using C Program.

ALGORITHM:

Step1: Start the Program.

Step2: To declare pointer and other variables.

Step3: To get the count for total number of elements.

Step4: To get extra memory allocation using calloc().

Step5: To get and print the numbers one by one.

Step5: Again reallocate memory using realloc().

Step6: To get and print the numbers one by one.

Step7: To de-allocate memories by free().

Step8: Stop the Program.

PROGRAM:

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

```
#include <stdlib.h>
```

```
int main()
```

```
{
```

```
    int* ptr;
```

```
    int n, i, sum = 0;
```

```
    n = 5;
```

```
    printf("Enter number of elements: %d\n", n);
```

```
ptr = (int*)calloc(n, sizeof(int));
if (ptr == NULL)
{
    printf("Memory not allocated.\n");
    exit(0);
}
else
{
    printf("Memory successfully allocated using calloc.\n");
    for (i = 0; i < n; ++i)
    {
        ptr[i] = i + 1;
    }
    printf("The elements of the array are: ");
    for (i = 0; i < n; ++i)
    {
        printf("%d, ", ptr[i]);
    }
    n = 10;
    printf("\n\nEnter the new size of the array: %d\n", n);
    ptr = realloc(ptr, n * sizeof(int));
    printf("Memory successfully re-allocated using realloc.\n");
    for (i = 5; i < n; ++i)
    {
        ptr[i] = i + 1;
    }
}
```

```

        printf("The elements of the array are: ");
        for (i = 0; i < n; ++i)
        {
            printf("%d, ", ptr[i]);
        }
        free(ptr);
    }

    return 0;
}

```

OUTPUT:

Enter number of elements: 5

Memory successfully allocated using calloc.

The elements of the array are: 1, 2, 3, 4, 5,

Enter the new size of the array: 10

Memory successfully re-allocated using realloc.

The elements of the array are: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,

RESULT:

Thus the following program is successfully run verified.

11. USE STRUCTURES AND UNIONS FOR SOLVING VARIOUS APPLICATION

EX.NO:

DATE:

GENERATE STUDENT MARKSHEET WITH SUBJECT DETAILS AND GRADES

AIM:

To write a C program to Generate student marksheet with subject details and Grades (Total and Average)

ALGORITHM:

Step 1: Start

Step 2: Define a structure named student that contains name,marks of 5 subjects, roll no and average marks

Step 3: Create a structure object s1 of type student.

Step 4: Input structure object's name and subject marks.

Step 5: Compute the total marks by summing up marks of subjects. Step 6:Compute the average marks by dividing by 5

Step 7: Generate student mark sheet with subject details and Grades

Step 8: End

PROGRAM:

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

```
struct student
```

```
{
```

```
    char name[20];
```

```
    int marks[5];
```

```
    int tot;
```

```
    float avg;
```

```
};
```

```

void main()
{
    struct student s1;
    int total=0,i;
    printf("Enter the name of the student : ");
    scanf("%s",s1.name);
    printf("\n Enter the marks of student :");
    for(i=0;i<5;i++)
    {
        printf("\n Mark%d: ",i+1); scanf("%d",&s1.marks[i]); total+=s1.marks[i];
    }
    s1.tot=total;
    s1.avg=total/5;
    printf("\n");
    printf("Student Name      :%s\n",s1.name);
    printf("English Marks      :%d\n",s1.marks[0] );
    printf("Maths Marks        :%d\n", s1.marks[1]);
    printf("Physics Marks       :%d\n",s1.marks[2]);
    printf("Chemistry Marks     :%d\n",s1.marks[3]);
    printf("Biology Marks       :%d\n\n",s1.marks[4]);
    printf("\n Total marks    : %d\n",s1.tot);
    printf("\n Average          : %f\n",s1.avg);
}

```

OUTPUT:

Enter the name of the student Ravikumar

Enter the marks of
student : Mark1:75

Mark2:85

Mark3:84

Mark4:84

Mark5:78

Student Name	:Ravikumar
English Marks	:75
Maths Marks	:85
Physics Marks	:84
Chemistry Marks	:84
Biology Marks	:78
Total marks	:406
Average	: 81.000000

RESULT:

Thus a C program to Generate student mark sheet with subject details and Grades (Total and Average) was done successfully

12. IMPLEMENT VARIOUS OPERATIONS OF SEQUENTIAL FILE

EX.NO:

DATE:

WRITE A C PROGRAM TO DISPLAY THE CONTENTS OF A FILE

AIM:

To display the contents of the file on the output screen.

ALGORITHM:

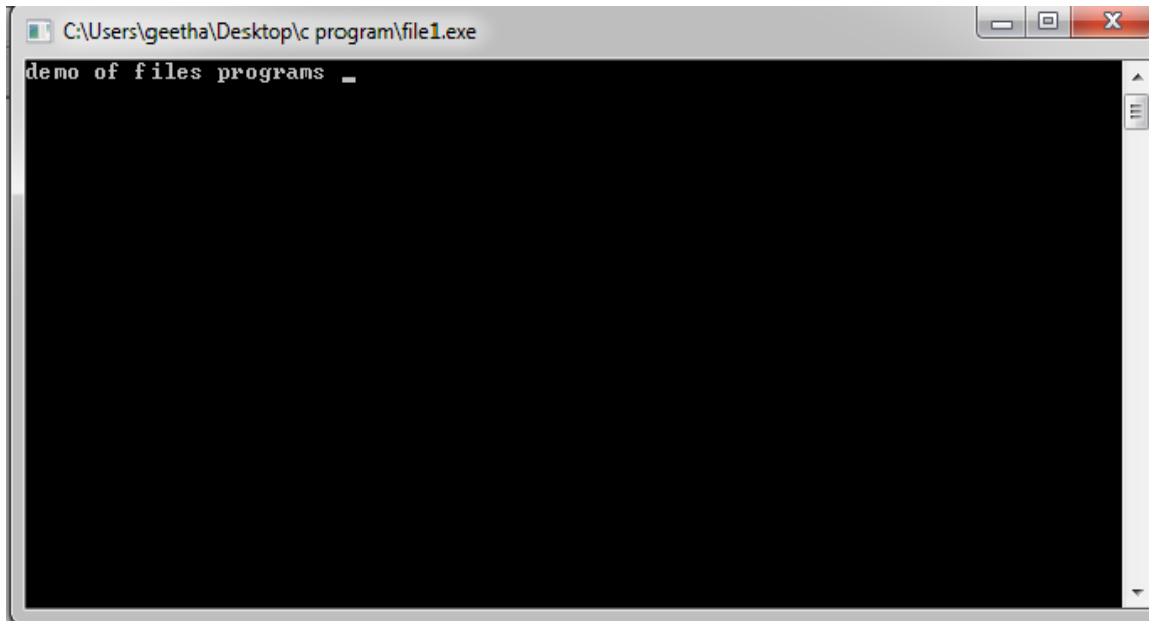
1. Start
2. Read file name
3. Open file in read mode
4. Repeat until end of the file and print the contents if the file
5. Close the file
6. Stop

PROGRAM:

```
#include<stdio.h>
void main()
{
    FILE *fp; char ch;
    fp=fopen("file.txt","r");

    if(fp==NULL)
    {
        printf("file opening problem"); return;
    }
    while(!feof(fp))
    {
        ch=fgetc(fp); printf("%c",ch);
    }
    fclose(fp);
}
```

OUTPUT:



A screenshot of a Windows command prompt window. The title bar at the top reads "C:\Users\geetha\Desktop\c program\file1.exe". The window contains a single line of text: "demo of files programs _". The rest of the window is black, indicating that the program has finished execution and is waiting for input.

RESULT:

Thus the program on displaying the file contents on the output screen is executed

13. EXECUTE THE FUNCTIONS OF RANDOM ACCESS FILE

EX.NO.:

DATE:

TELEPHONE DIRECTORY

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\tPhonebook Menu");
        printf("\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\n(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 a C Program was executed and the output was obtained.

CALCULATOR IN C

(MINI PROJECT IN C PROGRAMMING)

DONE BY [KRISHNA KUMAR.V,

LAVAN J V,

MARIYAPPAN.S]

Abstract:

Calculator is basic thing that is to be used for solving larger problems. Calculator performs as time saver instead of doing problems for a long time, it makes easier to solve problems.

Calculators in the elementary grades serve as aids in advancing student understanding without replacing the need for other calculation methods it allows students solve complicated problems quickly and in an efficient manner. Additionally, it can reduce the problem to simpler tasks and allows the student to devote more time in understanding the problem

So, this calculator is created to do required tasks with ease.

objective of the calculator:

i) Performs simple arithmetic operations

- (a) Addition
- (b) Subtraction

- (c) Multiplication
- (d) Division
- (e) Modulus

ii) Performs scientific calculations

- a) Power
- b) Factorial
- c) square
- d) cube
- e) square root
- f) Sin
- g) Cos
- h) Tan
- i) Log
- j) log10

Source code:

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

void calculator_choice();
void simple_calc();
void scientific_clac();
void addition();
void subtraction();
void multiplication();
void division();
void modulus();
void factorial();
void power();
void square();
```

```

void cube();
void squareroot();
void sinfn();
void cosfn();
void tanfn();
void logfn();
void log10fn();
void clear();

void main(){
    calculator_choice();
}

void calculator_choice(){
    clear();
    int calc;
    char c;
    printf("\t\tWelcome to the Normal & Scientific calculator!!\n\n");
    printf("\n***** Press e to quit the program *****\n");
    printf("Enter a for Normal Calculator \n");
    printf("Enter b for Scientific Calculator \n");

    while(1){
        printf("\n\nEnter the Calculator of your Choice: ");

        scanf("%c",&c);
        calc=c;

        switch(calc)
        {
            case 65:
                simple_calc();
                break;
            case 97:
                simple_calc();
                break;
            case 66:
                scientific_clac();
                break;
            case 98:
                scientific_clac();
                break;
            case 69:
                exit(0);
            case 101:
                exit(0);
            default:
                clear();
        }
    }
}

```

```

        printf("\n***** you violated ! exiting
*****\n");
        exit(0);
    }

}
}

```

```

void simple_calc(){
    clear();
    int choice;
    printf("\t\tWelcome to the simple calculator!!\n\n");
    printf("\n***** Press 0 to quit the program *****\n");
    printf("Enter 1 for Addition \n");
    printf("Enter 2 for Subtraction \n");
    printf("Enter 3 for Multiplication \n");
    printf("Enter 4 for Division \n");
    printf("Enter 5 for Modulus\n");

    while(1){
        printf("\nEnter the operation you want to do: ");

        scanf("%d",&choice);

        switch(choice)
        {
            case 1:
                addition();
                break;
            case 2:
                subtraction();
                break;
            case 3:
                multiplication();
                break;
            case 4:
                division();
                break;
            case 5:
                modulus();
                break;

            case 0:
                exit(0);
            default:
                clear();
        }
    }
}

```

```

        printf("\n***** you violated ! exiting
*****\n");
        exit(0);
    }

}
}

```

```

void scientific_clac(){
    clear();
    int choice;
    printf("\t\tWelcome to the Scientific calculator!!\n\n");
    printf("\n***** Press 0 to quit the program *****\n");
    printf("Enter 1 for Power \n");
    printf("Enter 2 for Factorial \n");
    printf("Enter 3 for square \n");
    printf("Enter 4 for cube \n");
    printf("Enter 5 for squareroot\n");
    printf("Enter 6 for Sin \n");
    printf("Enter 7 for Cos \n");
    printf("Enter 8 for Tan \n");
    printf("Enter 9 for log \n");
    printf("Enter 10 for log10\n");
}

```

```

while(1){
    printf("\nEnter the operation you want to do: ");
}

```

```

scanf("%d",&choice);

```

```

switch(choice)
{
    case 1:
        power();
        break;
    case 2:
        factorial();
        break;
    case 3:
        square();
        break;
    case 4:
        cube();
        break;
    case 5:
        squareroot();
        break;
    case 6:

```

```

        sinfn();
        break;
    case 7:
        cosfn();
        break;
    case 8:
        tanfn();
        break;
    case 9:
        logfn();
        break;
    case 10:
        log10fn();
        break;

        break;
    case 0:
        exit(0);
    default:
        clear();
        printf("\n***** you violated !
exiting *****\n");
        exit(0);
    }

}

}

void addition(){
    printf("Enter the numbers you want to add: ");
    int a,b;
    scanf("%d%d",&a,&b);
    printf("The sum of a and b is %d\n",a+b);

}

void subtraction(){
    printf("Enter the numbers you want to subtract: ");
    int a,b;
    scanf("%d%d",&a,&b);
    printf("The subtraction of a and b is %d\n",a-b);
}

void multiplication(){
    printf("Enter the numbers you want to multiply: ");
    int a,b;
    scanf("%d%d",&a,&b);
    printf("The multiplication of a and b is %d\n",a*b);
}

void division(){

```

```

    printf("Enter the numbers you want to divide: ");
    int a,b;
    scanf("%d%d",&a,&b);
    printf("The division of a and b is %f\n",(float)a/(float)b);
}
void modulus(){
    printf("Enter the numbers you want to find modulus of: ");
    int a,b;
    scanf("%d%d",&a,&b);
    printf("The modulus of a and b is %d\n",a%b);
}
void factorial(){
    int n,factorial;
    printf("Enter the number you want the factorial of: ");
    scanf("%d",&n);
    factorial=1;
    for(int i=1;i<=n;i++){
        factorial=factorial*i; // factorial*=i;
    }
    printf("Factorial of %d is %d\n",n,factorial);
}
void power(){
    double b;
    double p;
    printf("Enter the base and the power: ");
    scanf("%lf%lf",&b,&p);
    double e=pow(b,p);
    printf("The power is %lf\n",e);
}
void square(){
    double b;
    printf("Enter the number you want the square of: ");
    scanf("%lf",&b);
    double p=pow(b,2);
    printf("The square of %lf is %lf\n",b,p);
}
void cube(){
    double b;
    printf("Enter the number you want the cube of: ");
    scanf("%lf",&b);
    double p=pow(b,3);
    printf("The cube of %lf is %lf\n",b,p);
}
void squareroot(){
    double b;
    printf("Enter the number you want the square root of : ");
    scanf("%lf",&b);
    double s = sqrt(b);

```

```

    printf("The square root of %lf is %lf\n",b,s);
}
void sinfn(){
    double b;
    printf("Enter the number you want the Sin of : ");
    scanf("%lf",&b);
    double s = sin(b);
    printf("The Sin of %lf is %lf\n",b,s);
}
void cosfn(){
    double b;
    printf("Enter the number you want the Cos of : ");
    scanf("%lf",&b);
    double s = cos(b);
    printf("The Cos of %lf is %lf\n",b,s);
}
void tanfn(){
    double b;
    printf("Enter the number you want the Tan of : ");
    scanf("%lf",&b);
    double s = tan(b);
    printf("The Tan of %lf is %lf\n",b,s);
}
void logfn(){
    double b;
    printf("Enter the number you want the Log to the base 2: ");
    scanf("%lf",&b);
    double s = log(b);
    printf("The log to the base 2 of %lf is %lf\n",b,s);
}
void log10fn(){
    double b;
    printf("Enter the number you want the Log to the base 10: ");
    scanf("%lf",&b);
    double s = log10(b);
    printf("The log to the base 10 of %lf is %lf\n",b,s);
}
void clear(){
    system("cls");
}

```

Output:

Normal calculator:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Welcome to the Normal & Scientific calculator!!

***** Press e to quit the program *****
Enter a for Normal Calculator
Enter b for Scientific Calculator

Enter the Calculator of your Choice: a

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Welcome to the simple calculator!!

***** Press 0 to quit the program *****
Enter 1 for Addition
Enter 2 for Subtraction
Enter 3 for Multiplication
Enter 4 for Division
Enter 5 for Modulus

Enter the operation you want to do: 1
Enter the numbers you want to add: 2 3
The sum of a and b is 5

Enter the operation you want to do: 2
Enter the numbers you want to subtract: 4 1
The subtraction of a and b is 3

Enter the operation you want to do: 3
Enter the numbers you want to multiply: 2 4
The multiplication of a and b is 8

Enter the operation you want to do: 4
Enter the numbers you want to divide: 4 2
The division of a and b is 2.000000

Enter the operation you want to do: 5
Enter the numbers you want to find modulus of: 4 4
The modulus of a and b is 0

Scientific calculator:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Welcome to the Normal & Scientific calculator!!

```
***** Press e to quit the program *****
Enter a for Normal Calculator
Enter b for Scientific Calculator
```

Enter the Calculator of your Choice: b

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Welcome to the Scientific calculator!!

```
***** Press 0 to quit the program *****
Enter 1 for Power
Enter 2 for Factorial
Enter 3 for square
Enter 4 for cube
Enter 5 for squareroot
Enter 6 for Sin
Enter 7 for Cos
Enter 8 for Tan
Enter 9 for log
Enter 10 for log10
```

```
Enter the operation you want to do: 1
Enter the base and the power: 2 3
The power is 8.000000
```

```
Enter the operation you want to do: 2
Enter the number you want the factorial of: 5
Factorial of 5 is 120
```

```
Enter the operation you want to do: 3
Enter the number you want the square of: 4
The square of 4.000000 is 16.000000
```

```
Enter the operation you want to do: 4
Enter the number you want the cube of: 4
The cube of 4.000000 is 64.000000
```

```
Enter the operation you want to do: 5
Enter the number you want the square root of : 9
The square root of 9.000000 is 3.000000
```

```
Enter the operation you want to do: 6
Enter the number you want the Sin of : 0
The Sin of 0.000000 is 0.000000
```

```
Enter the operation you want to do: 7
Enter the number you want the Cos of : 0
```

powershell + ▢ ▢ ^ X

Error handling:

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
Welcome to the Normal & Scientific calculator!!
```

```
***** Press e to quit the program *****
```

```
Enter a for Normal Calculator
```

```
Enter b for Scientific Calculator
```

```
Enter the Calculator of your Choice: g
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
***** you violated ! exiting *****
```

```
PS F:\c project>
```

