PILATHARA CO-OPERATIVE ARTS & SCIENCE COLLEGE

PILATHARA KANNUR- 670504
(AFFILIATED TO KANNUR UNIVERSITY)

PRACTICAL RECORD

PROGRAMMING IN C LANGUAGE

NAME : ....

REG.NO :...

SEMESTER :..

SUBJECT :...

PILATHARA CO-OPERATIVE ARTS & SCIENCE COLLEGE

PILATHARA KANNUR- 670504
(AFFILIATED TO KANNUR UNIVERSITY)

PRACTICAL RECORD

CERTIFICATE

CERTIFIED THAT THIS IS A BONAFIDE RECORD OF THE ORIGINAL WORK DONE BY ....REG.NO...OF FIRST BCA IN THE C PROGRAMMING DURING THE YEAR 2020-2023

EXAMINERS: **LECTURER IN CHARGE:**

1

PRINCIPAL

2

PROGRAME : 1

THE SIZE AND RANGE OFALL THE DATA TYPE IN C

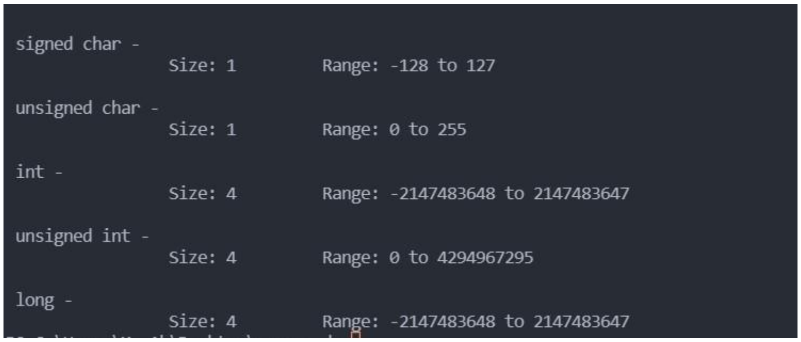
ALGORITHM

Step 1: Start
Step 2: Display "Size and Range of Five Data Types in C:-"
Step 3: Display the size and range of 'signed char'
Step 4: Display the size and range of 'unsigned char'
Step 5: Display the size and range of 'int'
Step 6: Display the size and range of 'unsigned int'
Step 7: Display the size and range of 'long'
Step 8: Stop

PROGRAME

```
#include <stdio.h>
#include <limits.h>
void main()
{
   printf("Size and Range of Five Data Types in C:-\n");
   printf("\n signed char -");
   printf("\n \t\t Size: %u \t Range: %d to %d \n", sizeof(signed char), SCHAR_MIN, SCHAR_MAX);
   printf("\n unsigned char -");
   printf("\n \t\t Size: %u \t Range: %u to %u \n", sizeof(unsigned char), 0, UCHAR_MAX);
   printf("\n int -");
   printf("\n \t\t Size: %u \t Range: %d to %d \n", sizeof(int), INT_MIN, INT_MAX);
   printf("\n unsigned int -");
   printf("\n \t\t Size: %u \t Range: %u to %u \n", sizeof(unsigned int), 0, UINT_MAX);
   printf("\n long -");
   printf("\n \t\t Size: %u \t Range: %li to %li \n", sizeof(long), LONG_MIN, LONG_MAX);
}
```

OUTPUT



```
signed char -
                Size: 1         Range: -128 to 127

unsigned char -
                Size: 1         Range: 0 to 255

int -
                Size: 4         Range: -2147483648 to 2147483647

unsigned int -
                Size: 4         Range: 0 to 4294967295

long -
                Size: 4         Range: -2147483648 to 2147483647
```

PROGRAME : 2

CONVERT FARANHEIT TO CELSIUS

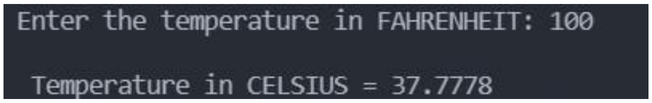
ALGORITHM

```
Step 1: Start
Step 2: Read the temperature in Fahrenheit f
Step 3: c ← (f - 32) * 5 / 9
Step 4: Display the temperature in Celsius c
Step 5: Stop
```

PROGRAME

```c
#include <stdio.h>
int main()
{
         float f, c;
        printf("Enter the temperature in FAHRENHEIT: ");
        scanf("%f", &f);
        c = (f - 32) * 5 / 9;
        printf("\n Temperature in CELSIUS = %g", c);
}
```

OUTPUT



```
Enter the temperature in FAHRENHEIT: 100

 Temperature in CELSIUS = 37.7778
```

PROGRAM : 3

PROGRAME TO FIND LARGEST AND SECOND LARGEST AMONG THREE

ALGORITHM

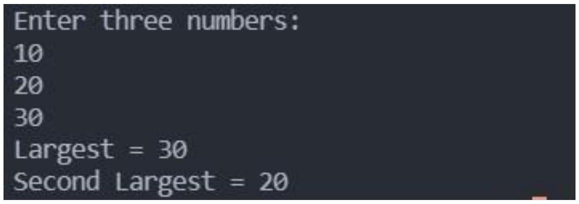
```
Step 1: Start
Step 2: Read three numbers a, b, c
Step 3: If a > b
If a > c
max1 ← a
If b > c
max2 ← b
Else
max2 ← c
Else
max1 ← c
max2 ← a
Else
If b > c
max1 ← b
If a>c
max2 ← a
Else
max2 ← c
Else
max1 ← c
max2 ← b
Step 4: Display the largest number max1
Step 5: Display the second largest number max2
Step 6: Stop
```

PROGRAME

```
#include <stdio.h>

void main()

{

   int a, b, c;

   int max1, max2;
```

```
    printf("Enter three numbers:\n");

   scanf("%d %d %d", &a, &b, &c);
   if (a > b)
    {
        if (a > c)
    {
    max1 = a;
    if (b > c)
    {
        max2 = b;
    }
    else
    {
        max2 = c;
    }
}
else
{
    max1 = c;
    max2 = a;
}
}
else
{
    if (b > c)
{
max1 = b;
if (a > c)
{
    max2 = a;
}
else
{
    max2 = c;
}
}
else
{
    max1 = c;
    max2 = b;
}
}
    printf("Largest = %d\n", max1);
    printf("Second Largest = %d\n", max2);
}
```

OUTPUT



```
Enter three numbers:
10
20
30
Largest = 30
Second Largest = 20
```

PROGRAM : 4

ROOTS OF A QUADRATIC EQUATION

ALGORITHM

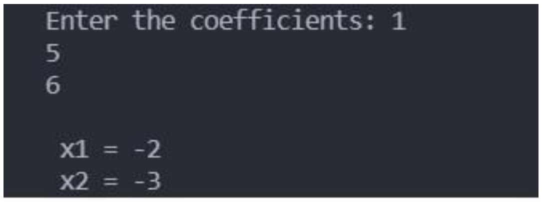
```
Step 1: Start
Step 2: Read the coefficients a, b, c
Step 3: D ← b2 – 4ac
Step 4: If D < 0
    Display “IMAGINARY ROOTS”
Else if D = 0
    x1 ← (-b) / (2 * a)
    Display the root x1
Else
    x1 ← (-b + √D) / (2 * a)
    x1 ← (-b - √D) / (2 * a)
    Display the roots x1 and x2
Step 5: Stop
```

PROGRAME

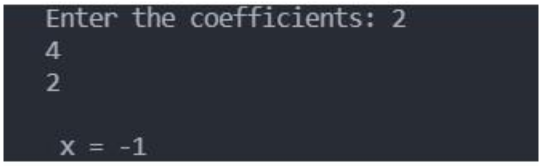
```c
#include <stdio.h>
#include <math.h>
void main()
{
    float a, b, c;
    float D, x1, x2;
    printf("Enter the coefficients: ");
    scanf("%f %f %f", &a, &b, &c);
    D = pow(b, 2) - (4 * a * c);
    if (D < 0.0f)
    {
        printf("\n IMAGINARY ROOTS");
    }
    else if (D == 0.0f)
    {
        x1 = (-b) / (2 * a);
        printf("\n x = %g", x1);
    }
```

```
     else
     {
          x1 = (-b + sqrt(D)) / (2 * a);
          x2 = (-b - sqrt(D)) / (2 * a);
          printf("\n x1 = %g", x1);
          printf("\n x2 = %g", x2);
      }
   }
```

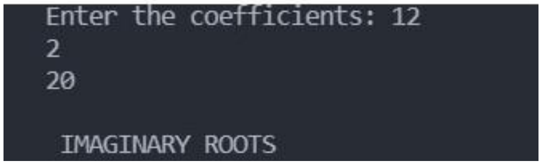
OUTPUT



```
Enter the coefficients: 1
5
6

 x1 = -2
 x2 = -3
```



```
Enter the coefficients: 2
4
2

 x = -1
```



```
Enter the coefficients: 12
2
20

 IMAGINARY ROOTS
```

PROGRAM : 5

PRIME NUMBERS BETWEEN TWO NUMBERS

ALGORITHM

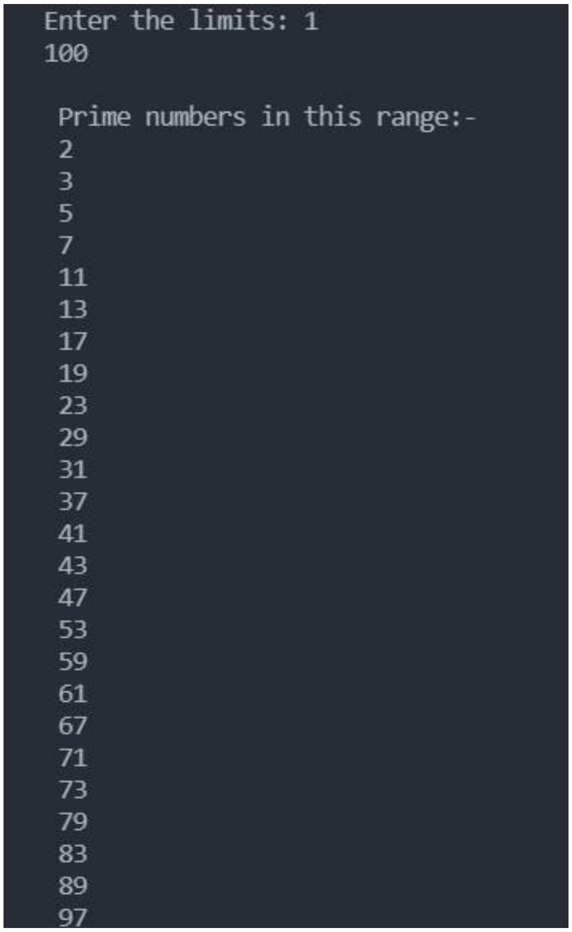
```
Step 1: Start
Step 2: Read the lower limit l and upper limit u
Step 3: Display “Prime numbers in this range:-“
Step 4: If l < 2
l ← 2
Step 5: Repeat the step until l ≤ u
Step 5.1: prime ← 1
Step 5.2: i ← 2
Step 5.3: Repeat the step until i < l
Step 5.3.1: If l%i = 0
Step 5.3.1.1: prime ← 0
Step 5.3.1.2: Go to Step 5.4
Step 5.3.2: i ← i+1
Step 5.4: If prime = 1
Display l
Step 5.5: l ← l+1
Step 6: Stop
```

PROGRAME

```
#include <stdio.h>
int main()
{
   int l, u;
   int i, prime;
   printf("Enter the limits: ");
   scanf("%d %d", &l, &u);
   printf("\n Prime numbers in this range:-");
   if (l < 2) l = 2;
   while (l <= u)
   {
      prime = 1;
      for (i = 2; i < l; ++i)
   {
   if (l % i == 0)
   {
      prime = 0;
      break;
   }
   }
   if (prime) printf("\n %d", l);
```

```
    ++l;
    }
}
```

OUTPUT



```
  Enter the limits: 1
  100

   Prime numbers in this range:-
   2
   3
   5
   7
   11
   13
   17
   19
   23
   29
   31
   37
   41
   43
   47
   53
   59
   61
   67
   71
   73
   79
   83
   89
   97
```

PROGRAM : 6

PROGRAME TO CHECK WHETHER THE GIVEN MATRIX IS IDENTITY METRIX OR NOT

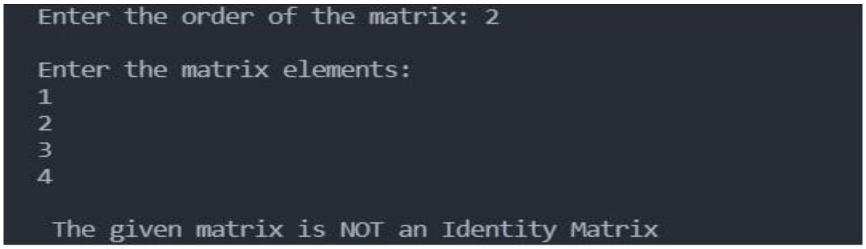
ALGORITHM

```
Step 1: Start
Step 2: identityMatrix ← 1
Step 3: Read the order of the matrix n
Step 4: i ← 0
Step 5: Repeat the step until i < n
Step 5.1: j ← 0
Step 5.2: Repeat the step until j < n
Step 5.2.1: Read M[i][j]
Step 5.2.2: If (i = j and M[i][j] ≠ 1) or (i ≠ j and M[i][j] ≠ 0)
identityMatrix ← 0
Step 5.2.3: j ← j+1
Step 5.3: i ← i+1
Step 6: If identityMatrix = 1
Display "The given matrix is an IDENTITY MATRIX"
Else
Display "The given matrix is NOT an Identity Matrix"
Step 7: Stop
```

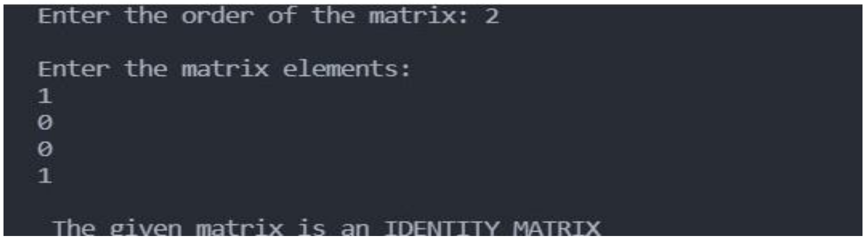
PROGRAME

```
#include <stdio.h>
#define SIZE 3
void main()
{
    int m[SIZE][SIZE];
    int n, i, j;
    int identityMatrix = 1;
    printf("Enter the order of the matrix: ");
    scanf("%d", &n);
    printf("\nEnter the matrix elements:\n");
    for (i = 0; i < n; ++i)
    {
        for (j = 0; j < n; ++j)
        {
            scanf("%d", &m[i][j]);
            if ((((i == j) && (m[i][j] != 1)) || ((i != j) && (m[i][j] != 0))))
            {
                identityMatrix = 0;
            }
        }
    }
    if (identityMatrix)
    {
        printf("\n The given matrix is an IDENTITY MATRIX");
    }
    else
    {
        printf("\n The given matrix is NOT an Identity Matrix");
    }

}
```

OUTPUT



```
Enter the order of the matrix: 2

Enter the matrix elements:
1
2
3
4

 The given matrix is NOT an Identity Matrix
```



```
Enter the order of the matrix: 2

Enter the matrix elements:
1
0
0
1

 The given matrix is an IDENTITY MATRIX
```

PROGRAM : 7

PROGRAME TO MULTIPLY MATRICES

ALGORITHM

```
Step 1: Start
Step 2: Read the row size r1 and column size c1 of matrix A
Step 3: Read the row size r2 and column size c2 of matrix B
Step 4: If c1 ≠ r2
  Step 4.1: Display "Matrix multiplication is NOT POSSIBLE in this case!"
  Step 4.2: Stop
Step 5: Read matrix A
Step 6: Read matrix B
Step 7: i ← 0
Step 8: Repeat the step until i < r1
  Step 8.1: j ← 0
  Step 8.2: Repeat the step until j < c2
    Step 8.2.1: P[i][j] ← 0
    Step 8.2.2: k ← 0
    Step 8.2.3: Repeat the step until k < c1
      Step 8.2.3.1: P[i][j] ← P[i][j] + (A[i][k] * B[k][j])
      Step 8.2.3.2: k ← k+1
    Step 8.2.4: j ← j+1
  Step 8.3: i ← i+1
Step 9: Display the product P
Step 10: Stop
```

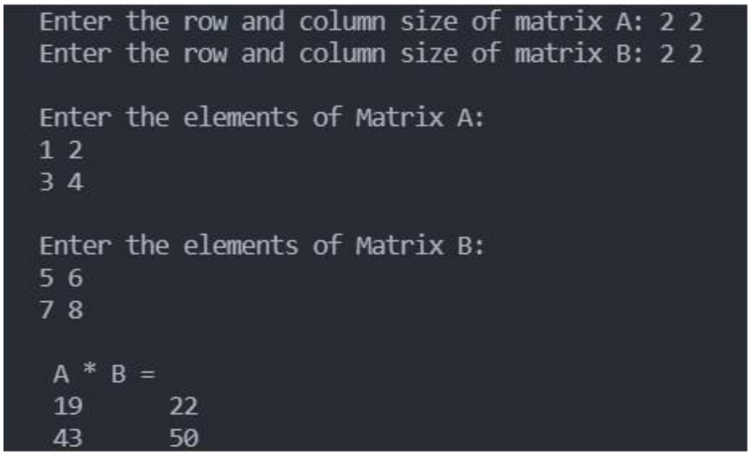
PROGRAME

```c
#include <stdio.h>

#define SIZE 3

void main()

{

   int a[SIZE][SIZE], b[SIZE][SIZE], p[SIZE][SIZE];

   int r1, c1, r2, c2;

   int i, j, k;
```

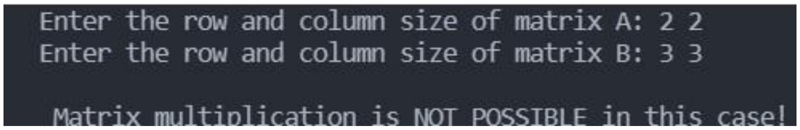
```
printf("Enter the row and column size of matrix A: ");

scanf("%d %d", &r1, &c1);

printf("Enter the row and column size of matrix B: ");

scanf("%d %d", &r2, &c2);

if (c1 != r2)

{

    printf("\n Matrix multiplication is NOT POSSIBLE in this case!");

}

else

{

    printf("\nEnter the elements of Matrix A:\n");

    for (i = 0; i < r1; ++i) for (j = 0; j < c1; ++j) scanf("%d", &a[i][j]);

    printf("\nEnter the elements of Matrix B:\n");

    for (i = 0; i < r2; ++i) for (j = 0; j < c2; ++j) scanf("%d", &b[i][j]);

    for (i = 0; i < r1; ++i)

    {

        for (j = 0; j < c2; ++j)

        {

            p[i][j] = 0;

            for (k = 0; k < c1; ++k)

            {

                p[i][j] += a[i][k] * b[k][j];

            }

        }

    }

    printf("\n A * B =");

    for (i = 0; i < r1; ++i)
```

```
      {
        printf("\n");
        for (j = 0; j < c2; ++j)
        {
          printf(" %d \t", p[i][j]);
        }
      }
    }
  }
```

OUTPUT



```
Enter the row and column size of matrix A: 2 2
Enter the row and column size of matrix B: 2 2

Enter the elements of Matrix A:
1 2
3 4

Enter the elements of Matrix B:
5 6
7 8

 A * B =
 19      22
 43      50
```



```
Enter the row and column size of matrix A: 2 2
Enter the row and column size of matrix B: 3 3

 Matrix multiplication is NOT POSSIBLE in this case!
```

PROGRAM : 8

PROGRAME TO ACCEPT TWO NUMBERS AND DO ARITHMATIC OPERATION (USING SWITCH)

ALGORITHM

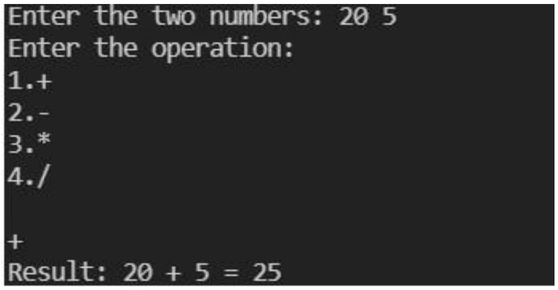
```
Step 1: Start
Step 2: Read the two numbers a, b
Step 3: Read the operation
Step 4: switch (operation)
Case '+': Display a + b
Case '-': Display a - b
Case '*': Display a * b
Case '/': Display a / b
Default: Display "INVALID Operation!"
Step 5: Stop
```

PROGRAME

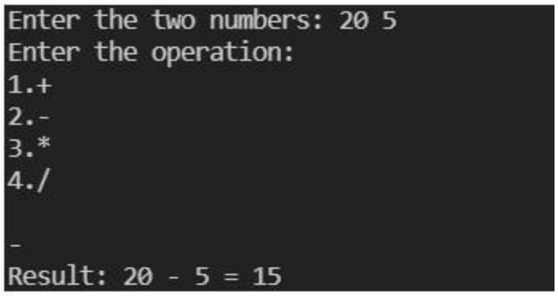
```
#include <stdio.h>
void main()
{
   float a, b;
   int n ;
   printf("Enter the two numbers: ");
   scanf("%f %f", &a, &b);
   printf("Enter the operation:\n1.Addition\n2.Substraction\n3.multiplication\n4.Division\n");
   scanf(" %c", &n);
   switch (n)
   {
      case '+' :
      {
         printf("Result: %g + %g = %g\n", a, b, a + b);
         break;
      }
      case '-' :
      {
         printf("Result: %g - %g = %g\n", a, b, a - b);
         break;
      }
      case '*':
      {
         printf("Result: %g * %g = %g\n", a, b, a * b);
         break;
      }
```

```
    case '/':
    {
      printf("Result: %g / %g = %g\n", a, b, a / b);
      break;
    }
      default:
    {
      printf(" INVALID Operation!\n");
    }
  }
}
```

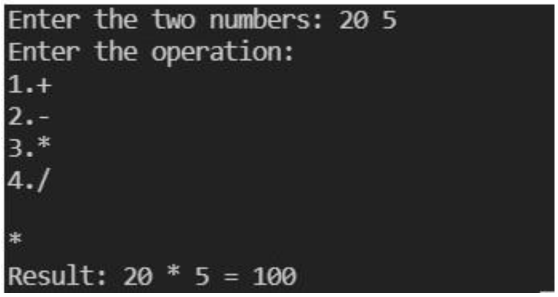
OUTPUT



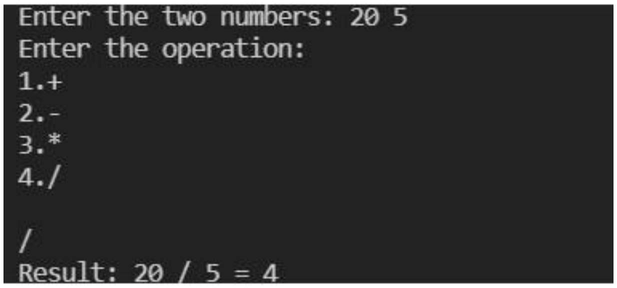
```
Enter the two numbers: 20 5
Enter the operation:
1.+
2.-
3.*
4./

+
Result: 20 + 5 = 25
```



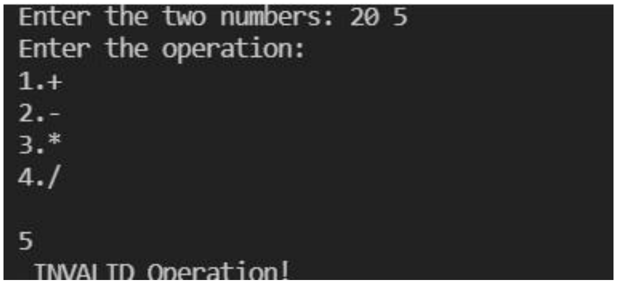
```
Enter the two numbers: 20 5
Enter the operation:
1.+
2.-
3.*
4./

-
Result: 20 - 5 = 15
```



```
Enter the two numbers: 20 5
Enter the operation:
1.+
2.-
3.*
4./

*
Result: 20 * 5 = 100
```



```
Enter the two numbers: 20 5
Enter the operation:
1.+
2.-
3.*
4./

/
Result: 20 / 5 = 4
```



```
Enter the two numbers: 20 5
Enter the operation:
1.+
2.-
3.*
4./

5
 INVALID Operation!
```

PROGRAM : 9

PROGRAME TO FIND FACTORIAL OF A NUMBER (RECURSIVE FUNCTION)

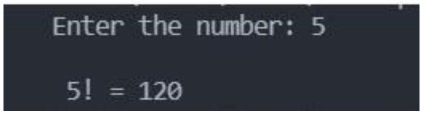
ALGORITHM

```
Step 1: Start
Step 2: Read the number n
Step 3: Display factorial(n)
Step 4: Stop
factorial(n):-
Step 1: Start
Step 2: If n ≤ 1
    Return 1
Else
    Return n * factorial(n - 1)
Step 3: Stop
```

PROGRAME

```c
#include <stdio.h>
unsigned long factorial(const unsigned short n)
{
   if (n <= 1) return 1;
   return n * factorial(n - 1);
}
void main()
{
   unsigned short n;
   unsigned long fact;
   printf("Enter the number: ");
   scanf("%hu", &n);
   fact = factorial(n);
   printf("\n %hu! = %lu", n, fact);
}
```

OUTPUT



```
Enter the number: 5

 5! = 120
```

PROGRAM : 10

A PROGRAME TO CHECK WHETHER A GIVEN STRING IS PALINDROME OR NOT (USING POINTER)

ALGORITHM

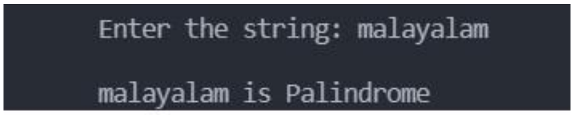
```
Step 1: Start
Step 2: palindrome ← 1
Step 3: Read the string str
Step 4: n ← length(str)
Step 5: i ← 0
Step 6: Repeat the step until i < n
Step 6.1: If str[i] ≠ str[n – 1 - i]
Step 6.1.1: palindrome ← 0
Step 6.1.2: Go to Step 7
Step 6.2: i ← i+1
Step 7: If palindrome = 1
Display “<str> is PALINDROME”
Else
Display “<str> is NOT Palindrome”
Step 8: Stop
```

PROGRAME

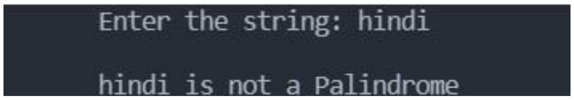
```
#include <stdio.h>
#include <string.h>
#define SIZE 30
void main()
{
    char str[SIZE];
    int n, i;
    int palindrome = 1;
    printf("\nEnter the string: ");
    gets(str);
    n = strlen(str);
    for (i = 0; i < n; ++i)
    {
        if (str[i] != str[n - 1 - i])
        {
            palindrome = 0;
            break;
        }
    }
    if (palindrome)
    {
        printf("\n%s is Palindrome", str);
    }
```

```
   else
 {
      printf("\n%s is not a Palindrome", str);
   }
}
```

OUTPUT



```
Enter the string: malayalam

malayalam is Palindrome
```



```
Enter the string: hindi

hindi is not a Palindrome
```

PROGRAM : 11

A PROGRAME TO CHECK THE NUMBER OF VOWELS IN A STRING

ALGORITHM

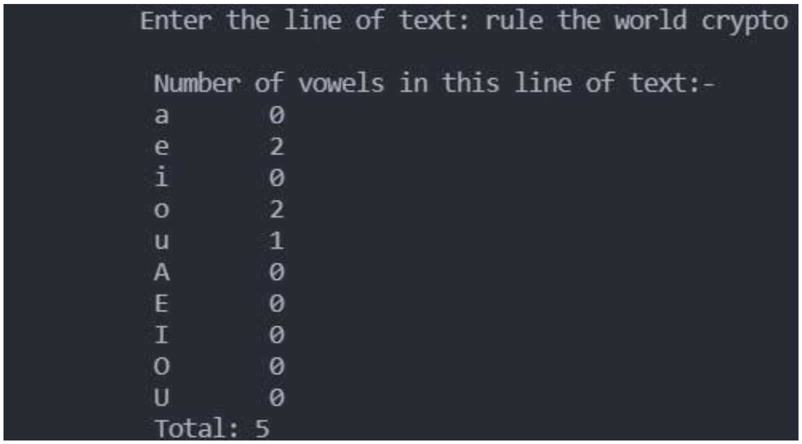
```
Step 1: Start
Step 2: na, ne, ni, no, nu, nA, nE, nI, nO, nU ← 0
Step 3: Read the text line
Step 4: n ← length(line)
Step 5: i ← 0
Step 6: Repeat the step until i < n
Step 6.1: ch ← line[i]
Step 6.2: If ch = a
    na ← na + 1
Else if ch = e
    ne ← ne + 1
Else if ch = i
    ni ← ni + 1
Else if ch = o
    no ← no + 1
Else if ch = u
    nu ← nu + 1
Else if ch = A
    nA ← nA + 1
Else if ch = E
    nE ← nE + 1
Else if ch = I
    nI ← nI + 1 36
Else if ch = O
    nO ← nO + 1
Else if ch = U
    nU ← nU + 1
Step 6.3: i ← i+1
Step 7: total ← na + ne + ni + no + nu + nA + nE + nI + nO + nU
Step 8: Display all the counts
Step 9: Stop
```

PROGRAME

```
#include <stdio.h>
#include <string.h>
#define SIZE 100
void main()
```

```
{
   char line[SIZE], ch;
   int i, n;
   unsigned na, ne, ni, no, nu, nA, nE, nI, nO, nU, total;
   na = ne = ni = no = nu = nA = nE = nI = nO = nU = 0;
   printf("Enter the line of text: ");
   gets(line);
   n = strlen(line);
   for (i = 0; i < n; ++i)
   {
      ch = line[i];
      if (ch == 'a') ++na;
      else if (ch == 'e') ++ne;
      else if (ch == 'i') ++ni;
      else if (ch == 'o') ++no;
      else if (ch == 'u') ++nu;
      else if (ch == 'A') ++nA;
      else if (ch == 'E') ++nE;
      else if (ch == 'I') ++nI;
      else if (ch == 'O') ++nO;
      else if (ch == 'U') ++nU;
   }
   total = na + ne + ni + no + nu + nA + nE + nI + nO + nU;
   printf("\n Number of vowels in this line of text:-");
   printf("\n a \t %u", na);
   printf("\n e \t %u", ne);
   printf("\n i \t %u", ni);
   printf("\n o \t %u", no);
   printf("\n u \t %u", nu);
   printf("\n A \t %u", nA);
   printf("\n E \t %u", nE);
   printf("\n I \t %u", nI);
   printf("\n O \t %u", nO);
   printf("\n U \t %u", nU);
   printf("\n Total: %u", total);
}
```

OUTPUT



```
Enter the line of text: rule the world crypto

 Number of vowels in this line of text:-
 a       0
 e       2
 i       0
 o       2
 u       1
 A       0
 E       0
 I       0
 O       0
 U       0
 Total: 5
```

PROGRAM : 12

EMPLOYEE DETAILS USING STRUCTURE

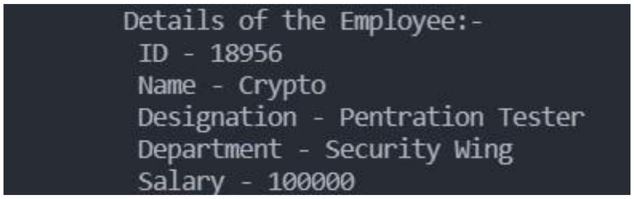
ALGORITHM

Step 1: Start
Step 2: Initialize an Employee structure
Step 3: Display the structure
Step 4: Stop

PROGRAME

```
#include <stdio.h>
#define SIZE 30
typedef struct
{
long id;
char name[SIZE];
char desg[SIZE];
char dept[SIZE];
float salary;
} Employee;
void main()
{
Employee e =
{
18956,
"Crypto",
"Pentration Tester",
"Security Wing",
100000.0f
};
printf("Details of the Employee:-");
printf("\n ID - %li", e.id);
printf("\n Name - %s", e.name);
printf("\n Designation - %s", e.desg);
printf("\n Department - %s", e.dept);
printf("\n Salary - %g", e.salary);
}
```

OUTPUT



```
Details of the Employee:-
 ID - 18956
 Name - Crypto
 Designation - Pentration Tester
 Department - Security Wing
 Salary - 100000
```

PROGRAM : 13

PROGRAME TO SWAP TWO NUMBER .

ALGORITHM

```
Step 1: Start
Step 2: Read two numbers a, b
Step 3: Display the two numbers before swap
Step 4: swap(&a, &b)
Step 5: Display the two numbers after swap
Step 6: Stop
```

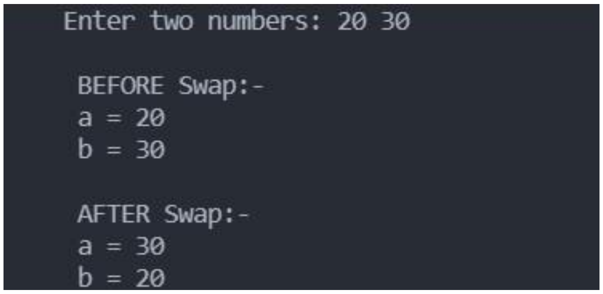
swap(*a, *b):-

```
Step 1: Start
Step 2: t ← *a
Step 3: *a ← *b
Step 4: *b ← t
Step 5: Stop
```

PROGRAME

```
#include <stdio.h>
void swap(int* a, int* b)
{
   int t;
   t = *a;
   *a = *b;
   *b = t;
}
void main()
{
   int a, b;
   printf("Enter two numbers: ");
   scanf("%d %d", &a, &b);
   printf("\n BEFORE Swap:-");
   printf("\n a = %d", a);
   printf("\n b = %d", b);
   swap(&a, &b);
   printf("\n");
   printf("\n AFTER Swap:-");
   printf("\n a = %d", a);
   printf("\n b = %d", b);
}
```

OUTPUT



```
Enter two numbers: 20 30

 BEFORE Swap:-
 a = 20
 b = 30

 AFTER Swap:-
 a = 30
 b = 20
```

PROGRAM : 14

ARRAY USING POINTERS

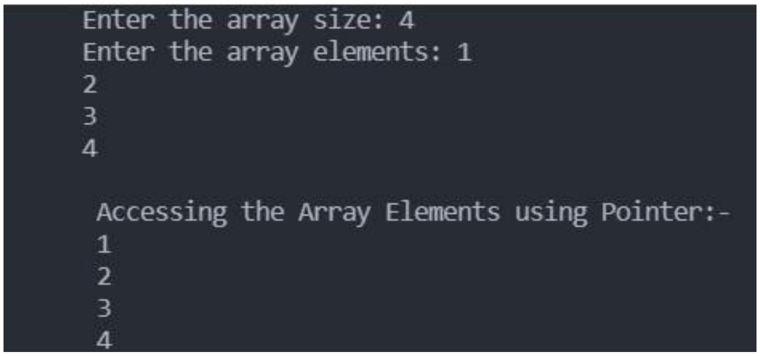
ALGORITHM

```
Step 1: Start
Step 2: Read the array size n
Step 3: Read the array a
Step 4: i ← 0
Step 5: Repeat the step until i < n
Step 5.1: ele ← *(a + i)
Step 5.2: Display ele
Step 5.3: i ← i+1
Step 6: Stop
```

PROGRAME

```
#include <stdio.h>
#define SIZE 10
void main()
{
   int a[SIZE];
   int *p = a;
   int n, i;
   int element;
   printf("Enter the array size: ");
   scanf("%d", &n);
   printf("Enter the array elements: ");
   for (i = 0; i < n; ++i) scanf("%d", &a[i]);
   {
      printf("\n Accessing the Array Elements using Pointer:-");
      for (i = 0; i < n; ++i)
      {
         element = *(p + i);
         printf("\n %d", element);
      }
   }
}
```

OUTPUT



```
Enter the array size: 4
Enter the array elements: 1
2
3
4

 Accessing the Array Elements using Pointer:-
 1
 2
 3
 4
```

PROGRAM : 15

CREATE A FILE , STORE RECORD AND DISPLAY IT

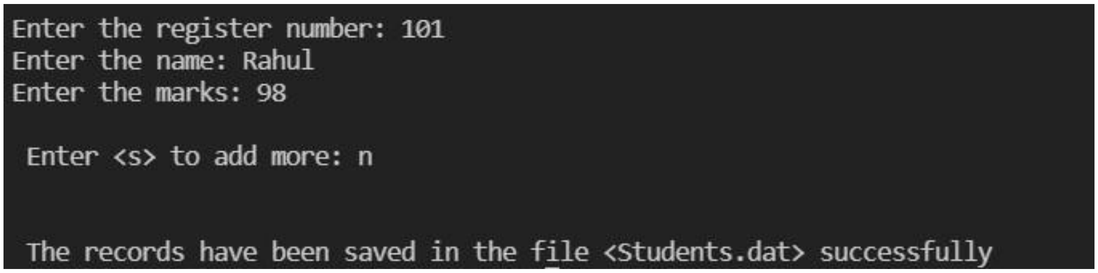
ALGORITHM

```
Step 1: Start
Step 2: filename ← "Students.dat"
Step 3: confirm ← ‘s’
Step 4: i ← 0
Step 5: file ← open a file to write in binary mode
Step 6: If error in creating the file
Step 6.1: Display “Cannot create the file!"
Step 6.2: Exit
Step 7: Repeat the step until confirm = ‘s’
Step 7.1: Read the student details s
Step 7.2: Write s into file
Step 7.3: Read confirm
Step 8: Close the file
Step 9: Display “The records have been saved in the file <filename> successfully"
Step 10: file ← open the file to read in binary mode
Step 11: If error in opening the file
Step 11.1: Display “Cannot open the file!"
Step 11.2: Exit
Step 12: Display “Here are the contents of this file:-“
Step 13: Repeat the step until all student details s have been read from the file
Step 13.1: Display the student details s of student i
Step 13.2: i ← i + 1
Step 14: Close the file 47
Step 15: Stop
```

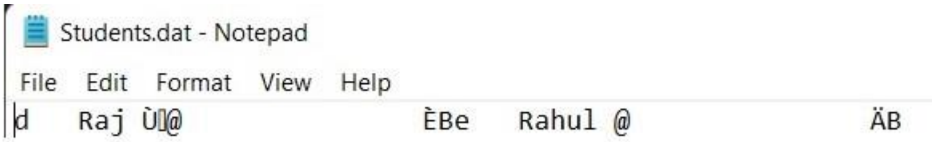
PROGRAME

```
#include <stdio.h>
#include <conio.h>
#include <stdlib.h>
const char* filename = "Students.dat";
typedef struct
{
long regno;
char name[20];
float marks;
} Student;
void main()
{
Student s;
char confirm = 's';
unsigned i = 0;
FILE* file = fopen(filename, "wb");
if (!file)
{
printf("Cannot create the file!");
getch();
exit(EXIT_FAILURE);
}
while (confirm == 's')
{
printf("Enter the register number: ");
scanf("%li", &s.regno);
printf("Enter the name: ");
scanf(" %[^\n]", s.name);
printf("Enter the marks: ");
scanf("%f", &s.marks);
fwrite(&s, sizeof(Student), 1, file);
printf("\n Enter <s> to add more: ");
scanf(" %c", &confirm);
printf("\n");
}
printf("\n The records have been saved in the file <%s> successfully", filename);
fclose(file);
file = fopen(filename, "rb");
if (!file)
{
printf("Cannot open the file!");
exit(EXIT_FAILURE);
}
}
```

OUTPUT



```
Enter the register number: 101
Enter the name: Rahul
Enter the marks: 98

 Enter <s> to add more: n


 The records have been saved in the file <Students.dat> successfully
```



Students.dat - Notepad

```
d    Raj Ù▯@                 ÈBe    Rahul @                ÄB
```