NED University Of Engineering And Technology

Department Of Telecommunication Engineering

Programming Languages (PL-105)

Topic:

- Scientific Calculator.

Presented to:

- Dr.Amir

Presented by:

- Anas Tauseef: TC-063
- Muhammad Owais (TC-045)

MOTIVATION: -

The selection of a scientific calculator for our programming language project is rooted in its capacity for advanced mathematical operations, precision, and user-friendly interface. Scientific Calculator helped us in every subject and through which we solve are various complex problem.it is become a based need in every stage of study.

SUMMARY: -

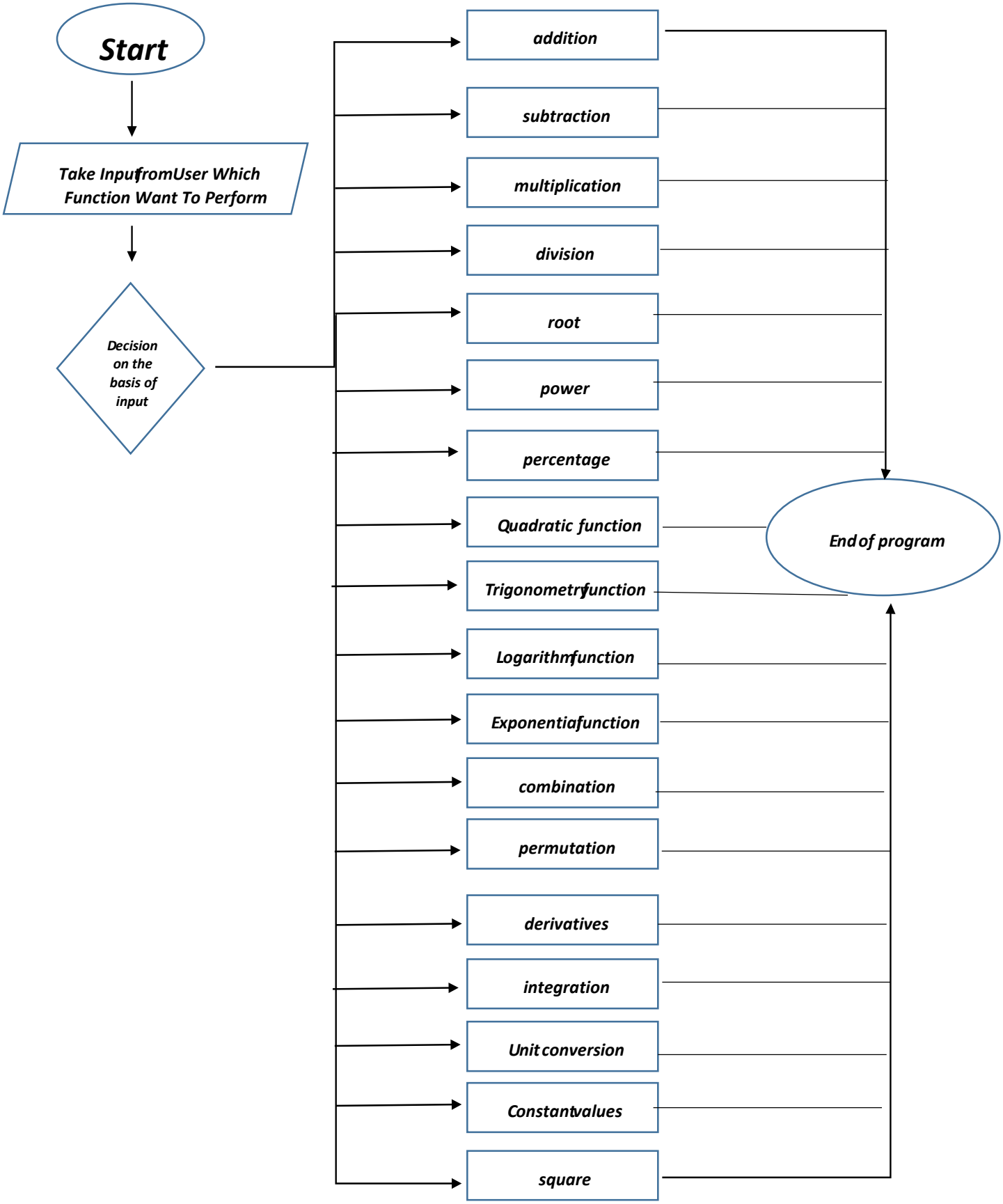
In this CEP assignment we focused on developing a scientific calculator, a tool widely used in scientific and engineering applications. The goal was to create a versatile and user-friendly calculator that could handle complex mathematical operations, making it suitable for various programming tasks we take input from user and give its choice to perform single function just like a calculator.

PROJECT DETAIL'S: -

- `Switch Case`: A control structure in programming that allows the selection of a branch of code based on the value of an expression.
- `If-Else Statement`: A conditional statement that executes specific blocks of code based on whether a given condition is true or false.
- `Function Prototype`: A declaration specifying a function's name, return type, and parameters, providing a blueprint for its implementation.
- `Getche`: A function in C programming that reads a single character from the console without waiting for the Enter key.
- `Clrscr`: A function used to clear the console screen in C programming.
- `Stdio.h Library`: A standard input/output library in C providing functions like printf and scanf.
- `Define PI Library`: A symbolic constant definition often used for representing the mathematical constant Pi in C programming.

- `Math.h Library`: A library in C programming providing mathematical functions and constants.
- `Main()`: The main function in C, serving as the program's entry point where execution begins.

GENERALIZED FLOWCHART: -



Source Code:

```
#include<stdio.h>
#include<math.h>
#define PI 3.14159265

void add(void);
void sub(void);
void mult(void);
void div(void);
void root(void);
void square(void);
void cube(void);
void power(void);
void quadratic(void);
void
trignometry(void);
void
combination(void);
void
permutation(void);
int derivative();
void complex();
void
addComplex(int,int,int
,int);
void
subtractComplex(int,in
t,int,int);
void
multiplyComplex(int,in
t,int,int);
int
definite_integration()
;
int percentage();
int Logarithm();
int exponential();
int
unit_conversions_physi
cs();
int constants();

void main(void){
    int a,b;
    int operation;
    float
num1,mun2,answer;
         clrscr();
        printf("
Welcome To Scientific
Calculator\n");
        printf("
Created by Anas and
Owais\n");
    //Gives User
Choice To Enter
Operations
    printf("\nEnter an
Operation:");

    //Calculator
Functions
    printf("\n(1)-For
Addition :");
    printf("\n(2)-For
Subtruction :");
    printf("\n(3)-For
Multiplication :");
    printf("\n(4)-For
Division :");
    printf("\n(5)-For
Root :");
    printf("\n(6)-For
Square :");
    printf("\n(7)-For
Cube :");
    printf("\n(8)-For
Power :");
    printf("\n(9)-For
Percentage :");
    printf("\n(10)-For
Quadratic Equation
:");
    printf("\n(11)-For
Trignometry Function
:");
    printf("\n(12)-For
logaritam Function
:");
    printf("\n(13)-For
Exponential Function
:");
    printf("\n(14)-For
Combination :");
    printf("\n(15)-For
Permutation :");
    printf("\n(16)-For
Derivatives :");
    printf("\n(17)-For
Complex Numbers :");

    printf("\n(18)-For
definite_Integration
:");
    printf("\n(19)-
Unit-Conversions :");
    printf("\n(20)-
Constant Values :");

scanf("%d",&operation)
;

    //By Using swith
Case
    switch(operation){
    case 1 : add();
break;
    case 2 : sub();
break;
    case 3 : mult();
break;
    case 4 : div();
break;
    case 5 : root();
break;
    case 6 : square();
break;
    case 7 : cube();
break;
    case 8 : power();
break;
    case 9 :
percentage(); break;
    case 10 :
quadratic(); break;
    case 11  :
trignometry(); break;
    case 12 :
Logarithm(); break;
    case 13 :
exponential(); break;
    case 14 :
combination(); break;
    case 15 :
permutation(); break;
    case 16 :
derivative(); break;
    case 17 :
complex(); break;
    case 18 :
definite_integration()
; break;
    case 19 :
unit_conversions_physi
cs(); break;
    case 20 :
constants(); break;
    default:
printf("enter a valid
number for
operation");
    }
    getche();
}

//Addition

void add(void){
    int a,b;
    printf("\n\nEnter
a Number a : ");
    scanf("%d",&a);
    printf("\nEnter a
Number b : ");
    scanf("%d",&b);
```

```
    printf("\nThe
Addition  of %d and %d
is %d",a,b,a+b);
   // getche();
}

//Subtruction

void sub(void){
    int a,b;
    printf("\n\nEnter
a Number a : ");
    scanf("%d",&a);
    printf("\nEnter a
Number b : ");
    scanf("%d",&b);
    printf("\nThe
subtruction  of %d and
%d is %d",a,b,a-b);
  //  getche();
}

//MultiPlication

void mult(void){
    int a,b;
    printf("\n\nEnter
a Number a : ");
    scanf("%d",&a);
    printf("\nEnter a
Number b : ");
    scanf("%d",&b);
    printf("\nThe
Multiplication  of %d
and %d is
%d",a,b,a*b);
    //getche();
}

//Division

void div(void){
    int a,b;
    printf("\n\nEnter
a Number a : ");
    scanf("%d",&a);
    printf("\nEnter a
Number b : ");
    scanf("%d",&b);
    printf("\nThe
divison  of %d and %d
is %d",a,b,a/b);
   // getche();
}

//Root

void root(void){
    float a,root;
    printf("\n\nEnter
a Number a : ");
    scanf("%2f",&a);
    root=sqrt(a);
    printf("\nThe
Square Root of %2f is
%2f",a,root);
     //getche();
}

//Square

void square(void){
    int a;
    printf("\n\nEnter
a Number a : ");
    scanf("%d",&a);
    printf("\nThe
square of %d is
%d",a,a*a);
    //getche();
}

//Cube

void cube(void){
    int a;
    printf("\n\nEnter
a Number a : ");
    scanf("%d",&a);
    printf("\nThe cube
of %d is %d",a,a*a*a);
    //getche();
}

//Power

void power(void){
    int a,n,power;
    printf("\n\nEnter
a Number a : ");
    scanf("%d",&a);
    printf("\nEnter
The Power of a : ");
    scanf("%d",&n);
    power=pow(a,n);
    printf("\nThe
power of %d is
%d",a,power);
    //getche();
}

int percentage() {
    int n, i;
    float
subjectGPA,totalMarks
= 0,totalSubjectMarks,
obtainedMarks,
percentage, gpa = 0;

    //number Of
Subjects
    printf("Enter the
number of subjects:
");
    scanf("%d", &n);

    //Details Of
    printf("Enter the
details for each
subject:\n");

    //loppp for
calculate each subject
    for (i = 1; i <=
n; i++) {
     printf("Enter
obtained marks for
subject %d: ", i);
     scanf("%f",
&obtainedMarks);
     printf("Enter
total marks for
subject %d: ", i);
     scanf("%f",
&totalSubjectMarks);

     totalMarks +=
obtainedMarks;

     // Calculate GPA
for each subject
     subjectGPA =
(obtainedMarks /
totalSubjectMarks) *
4.0;
     gpa +=
subjectGPA;

     // Print grade
for each subject
     printf("Grade for
subject %d: ", i);

     if (subjectGPA >=
3.67) {

printf("A\n");
     }
     else if
(subjectGPA >= 3.33) {
          printf("A-
\n");
     }
     else if
(subjectGPA >= 3.0) {

printf("B+\n");
     }
      else if
(subjectGPA >= 2.67) {

printf("B\n");
```

```
    }
    else if
(subjectGPA >= 2.33) {
        printf("B-
\n");
    }
    else if
(subjectGPA >= 2.0) {

printf("C+\n");
    }
    else if
(subjectGPA >= 1.67) {

printf("C\n");
    }
    else if
(subjectGPA >= 1.33) {
        printf("C-
\n");
    }
    else if
(subjectGPA >= 1.0) {

printf("D\n");
    }
    else {

printf("F\n");
    }
   }

    // Calculate
overall percentage
    percentage =
(totalMarks / (n *
100)) * 100;

    // Calculate
overall GPA
    gpa /= n;

    printf("\nOverall
Percentage: %.2f%%\n",
percentage);
    printf("Overall
GPA: %.2f\n", gpa);

    return 0;
    //getche();
}


//Quadratic Equation

void quadratic(void){
    float a, b, c,
discriminant, root1,
root2;

    // Input
coefficients
    printf("Enter
coefficients (a, b, c)
of the quadratic
equation (ax^2 + bx +
c): ");
    scanf("%lf %lf
%lf", &a, &b, &c);

    // Calculate
discriminant
    discriminant = b *
b - 4 * a * c;

    root1=(-
b+sqrt(discriminant))/
(2*a);
    root2=(-b-
sqrt(discriminant))/(2
*a);
    //Roots
    printf("%f and %f
are The Roots Of The
Quadratic
Equation\n",root1,root
2);
    // Check the
nature of roots
    if (discriminant >
0) {
     // Real and
distinct roots

     printf("The Roots
of the Quadratic
equation are Real");
    } else if
(discriminant == 0) {
     // Real and equal
roots
     printf("The Roots
of the Quadratic
equation are Real");

    } else {
     // Complex roots
     printf("The Roots
of the Quadratic
equation are
Complex");
          // getche();
    }
}

//trignomatery

void trignometry(void)
{
    double angle,
radians;
    int choice;

    // Input angle in
degrees
    printf("Enter the
angle in degrees: ");
    scanf("%lf",
&angle);

    // Convert angle
to radians
    radians = angle *
PI / 180.0;

    // Display menu
for trigonometric
functions
    printf("\nChoose
the trigonometric
function to
calculate:\n");
    printf("1. Sin\n2.
Cos\n3. Tan\n4.
Sec\n5. Cosec\n6.
Cot\n");
    printf("Enter your
choice (1-6): ");
    scanf("%d",
&choice);

    // Calculate and
display selected
trigonometric function
    switch (choice) {
     case 1:

printf("Sin(%lf) =
%lf\n", angle,
sin(radians));
         break;
     case 2:

printf("Cos(%lf) =
%lf\n", angle,
cos(radians));
         break;
     case 3:

printf("Tan(%lf) =
%lf\n", angle,
tan(radians));
         break;
     case 4:

printf("Sec(%lf) =
%lf\n", angle, 1.0 /
cos(radians));
         break;
     case 5:

printf("Cosec(%lf) =
```

```
%lf\n", angle, 1.0 /
sin(radians));
         break;
      case 6:

printf("Cot(%lf) =
%lf\n", angle, 1.0 /
tan(radians));
         break;
      default:

printf("Invalid
choice\n");
    }
     //getche();
    }

int Logarithm() {
    double num,
result;
    int choice;

    // Take user input
    printf("Enter a
number: ");
    scanf("%lf",
&num);

    // Display options
    printf("Choose an
option:\n");
    printf("1. Common
Logarithm (log10)\n");
    printf("2.
Exponential Logarithm
(exp)\n");
    printf("Enter your
choice (1 or 2): ");
    scanf("%d",
&choice);

    // Perform
calculation based on
user choice
    switch (choice) {
      case 1:
         result =
log10(num);

printf("Common
Logarithm (log10) of
%.2lf is %.2lf\n",
num, result);
         break;

      case 2:
         result =
log(num);

printf("Exponential
Logarithm (exp) of
%.2lf is %.6lf\n",
num, result);
         break;

      default:

printf("Invalid
choice\n");
    }
    return 0;
    //getche();
}

    //Exponential
Function

    int exponential()
{
    double x,result;

    printf("Enter a
number x: ");
    scanf("%lf", &x);

    result = exp(x);


printf("Exponential
function (e^%.2lf):
%.6lf\n", x, result);

    return 0;
    //getche();
    }

    //Combination

    void
combination(void){
    int
i,n,r,nfac=1,rfac=1,nr
fac=1,ncr;
    printf("Enter n :
");
    scanf("%d",&n);
    printf("Enter r :
");
    scanf("%d",&r);
    for(i=2;i<=n;i++){
      nfac=nfac*i;
    }
    for(i=2;i<=r;i++){
      rfac=rfac*i;
    }
    for(i=2;i<=n-
r;i++){
      nrfac=nrfac*i;
    }

ncr=nfac/(rfac*nrfac);
    printf(" The
Combination when n=%d
and r=%d is equal to
%d",n,r,ncr);
    //getche();
    }
    void
permutation(void){
      int
i,r,n,nfac=1,nrfac=1,n
pr;
      printf("Enter n :
");
      scanf("%d",&n);
      printf("Enter r :
");
      scanf("%d",&r);
      for(i=2;i<=n;i++)
{
      nfac=nfac*i;
      }
      for(i=2;i<=n-
r;i++){

nrfac=nrfac*i;
      }
      npr=nfac/(nrfac);
      printf(" The
Combination when n=%d
and r=%d is equal to
%d",n,r,npr);
      //getche();
    }

//derivatives
int derivative()
 {
    int
choice,degree,i,x, a,
b, c, d;

    printf("Choose the
type of equation:\n");

    printf("1.
Linear\n");
    printf("2.
Quadratic\n");
    printf("3.
Cubic\n");
    printf("Enter your
choice (1, 2, or 3):
");
    scanf("%d",
&choice);

    switch (choice) {
      case 1:
          printf("Enter
the coefficient 'a'
```

```
for the linear
equation (ax + b): ");
          scanf("%d",
&a);
          printf("The
derivative of the
linear equation is:
%d\n",a);
          break;
      case 2:
          printf("Enter
the coefficients 'a',
'b', and 'c' for the
quadratic equation
(ax^2 + bx + c): ");
          scanf("%d %d
%d", &a, &b, &c);
          printf("The
derivative of the
quadratic equation is:
%dx+%d\n",a*2,b);
          break;

      case 3:
          printf("Enter
the coefficients 'a',
'b', 'c', and 'd' for
the cubic equation
(ax^3 + bx^2 + cx +
d): ");
          scanf("%d %d
%d %d", &a, &b, &c,
&d);
          printf("The
derivative of the
cubic equation is:
%dx^2+%dx+%d\n",a*3,b*
2,c);
          break;
          printf("\n");
    break;
      default:
        printf("Invalid
choice! Please choose
1, 2, or 3.\n");
        return 1;  //
Return an error code
    }

    return 0;
    //getche();
}


//  complex number
void complex()
{
int a,b,c,d;
char operation;
printf("enter the two
real no :");
scanf("%d,%d",&a,&c);
printf("\nenter the
two Imaginary no :");
scanf("%d,%d",&b,&d);
printf("Choose
operation (+, -, *):
");
scanf(" %c",
&operation);
switch (operation)
{
case
'+':addComplex(a,b,c,d
);break;
case '-
':subtractComplex(a,b,
c,d);break;
case
'*':multiplyComplex(a,
b,c,d);break;
default:printf("Invali
d operation\n");
}
}
void addComplex(int
a,int b,int c,int d)
{
printf("\nfirst
complex no is
%d+%di",a,b);
printf("\n second
complex no is
%d+%di",c,d);
printf("\nthe addition
of two complex no is
%d+%di",a+c,b+d);
}
void
subtractComplex(int
a,int b,int c,int d)
{
printf("\nfirst
complex no is
%d+%di",a,b);
printf("\n second
complex no is
%d+%di",c,d);
printf("\nthe
subtraction of two
complex no is
%d+%di",a-c,b-d);
}
void
multiplyComplex(int
a,int b,int c,int d)
{
printf("\nfirst
complex no is
%d+%di",a,b);
printf("\n second
complex no is
%d+%di",c,d);
printf("\nthe multiply
of two complex no is
%d+%di",a*c,b*d);
//getche();
}
//definite_integration
int
definite_integration()
{
    int choice;
    float a, b, c, d,
result;

    printf("Select the
type of equation:\n");
    printf("1.
Quadratic equation
(ax^2 + bx + c)\n");
    printf("2. Linear
equation (ax + b)\n");
    printf("3. Cubic
equation (ax^3 + bx^2
+ cx + d)\n");
    printf("Enter your
choice (1/2/3): ");
    scanf("%d",
&choice);

    printf("Enter the
coefficients:\n");
    printf("Enter 'a':
");
    scanf("%f", &a);
    printf("Enter 'b':
");
    scanf("%f", &b);

    if (choice == 1 ||
choice == 3) {
     printf("Enter
'c': ");
     scanf("%f", &c);
    }

    if (choice == 3) {
     printf("Enter
'd': ");
     scanf("%f", &d);
    }

    // Calculate the
definite integral
based on the user's
choice
    switch (choice) {
     case 1: //
Quadratic equation
```

```
            result = (a /
3) * (1 * 1 * 1 - 0 *
0 * 0) + (b / 2) * (1
* 1 - 0 * 0) + c * (1
- 0);
            break;
        case 2: // Linear
equation
            result = (a /
2) * (1 * 1 - 0 * 0) +
b * (1 - 0);
            break;
        case 3: // Cubic
equation
            result = (a /
4) * (1 * 1 * 1 * 1 -
0 * 0 * 0 * 0) + (b /
3) * (1 * 1 * 1 - 0 *
0 * 0) + (c / 2) * (1
* 1 - 0 * 0) + d * (1
- 0);
            break;
        default:

printf("Invalid
choice.\n");
            return 1; //
Exit with an error
code
    }

    printf("Definite
integral result:
%.2f\n", result);

    return 0;
    //getche();
}
int
unit_conversions_physi
cs() {
    int choice;
    double length,
mass, temperatureC,
time, force, velocity,
acceleration;
    printf("Select the
type of unit
conversion:\n");
    printf("1.
Length\n");
    printf("2.
Mass\n");
    printf("3.
Temperature\n");
    printf("4.
Time\n");
    printf("5.
Force\n");
    printf("6.
Velocity\n");
    printf("7.
Acceleration\n");
    printf("Enter your
choice (1-7): ");
    scanf("%d",
&choice);
    switch (choice) {
        case 1: // Length
            printf("Enter
length in meters: ");
            scanf("%lf",
&length);


printf("Converted
lengths:\n");
            printf("In
centimeters: %.2lf
cm\n", length * 100);
            printf("In
kilometers: %.6lf
km\n", length / 1000);
            break;

        case 2: // Mass
            printf("Enter
mass in kilograms: ");
            scanf("%lf",
&mass);


printf("Converted
masses:\n");
            printf("In
grams: %.2lf g\n",
mass * 1000);
            printf("In
pounds: %.6lf lbs\n",
mass * 2.20462);
            break;

        case 3: //
Temperature
            printf("Enter
temperature in
Celsius: ");
            scanf("%lf",
&temperatureC);


printf("Converted
temperatures:\n");
            printf("In
Fahrenheit: %.2lf
Â°F\n", (temperatureC
* 9 / 5) + 32);
            printf("In
Kelvin: %.2lf K\n",
temperatureC +
273.15);
            break;

        case 4: // Time
            printf("Enter
time in seconds: ");
            scanf("%lf",
&time);


printf("Converted
times:\n");
            printf("In
minutes: %.2lf min\n",
time / 60);
            printf("In
hours: %.4lf hours\n",
time / 3600);
            break;

        case 5: // Force
            printf("Enter
force in Newtons: ");
            scanf("%lf",
&force);


printf("Converted
forces:\n");
            printf("In
dynes: %.2lf dyn\n",
force * 100000);
            printf("In
pounds-force: %.6lf
lbf\n", force *
0.224809);
            break;

        case 6: //
Velocity
            printf("Enter
velocity in meters per
second: ");
            scanf("%lf",
&velocity);


printf("Converted
velocities:\n");
            printf("In
kilometers per hour:
%.2lf km/h\n",
velocity * 3.6);
            printf("In
miles per hour: %.6lf
mph\n", velocity *
2.23694);
            break;

        case 7: //
Acceleration
            printf("Enter
acceleration in meters
```

```
per second squared:
");
          scanf("%lf",
&acceleration);


printf("Converted
accelerations:\n");
          printf("In
centimeters per second
squared: %.2lf
cm/s^2\n",
acceleration * 100);
          printf("In
gravity: %.6lf g\n",
acceleration / 9.81);
          break;

      default:

printf("Invalid
choice.\n");
          return 1; //
Exit with an error
code
    }

    return 0;
   // getche();
}

//Constants

int constants() {
    int a;
    printf("Enter
Choice For Constants
:");
    printf("1-For
Physics Constant");
    printf("2-For
chemistry Constant");
    scanf("%d",&a);
    switch(a){
    case 1:
    printf("Physics
Constants and
Units:\n");
    printf("1. Speed
of light in a vacuum:
c = 3.00 x 10^8
m/s\n");
    printf("2.
Planck's constant: h =
6.63 x 10^-34 J*s\n");
    printf("3.
Gravitational
constant: G = 6.67 x
10^-11 N*m^2/kg^2\n");
    printf("4.
Elementary charge: e =
1.60 x 10^-19 C\n");
    printf("5.
Boltzmann constant: k
= 1.38 x 10^-23
J/K\n");
    printf("6.
Avogadro's number: N_A
= 6.02 x 10^23 mol^-
1\n");
    printf("7. Gas
constant: R = 8.31
J/(mol*K)\n");
    printf("8. Mass of
electron: m_e = 9.11 x
10^-31 kg\n");
    printf("9. Mass of
proton: m_p = 1.67 x
10^-27 kg\n");
    printf("10.
Permittivity of free
space: Îµ_0 = 8.85 x
10^-12
C^2/(N*m^2)\n");
    printf("11.
Permeability of free
space: Î¼_0 = 4Ï€ x
10^-7 T*m/A\n");
    printf("12. Fine-
structure constant: Î±
â‰ˆ 1/137\n");
    printf("13.
Rydberg constant:
R_inf = 1.10 x 10^7
m^-1\n");
    printf("14.
Electron charge-to-
mass ratio: e/m_e â‰ˆ
1.76 x 10^11 C/kg\n");
    printf("15.
Stefan-Boltzmann
constant: Ïƒ = 5.67 x
10^-8 W/(m^2*K^4)\n");
    printf("16.
Universal
gravitational
constant: G â‰ˆ 6.67 x
10^-11 N*m^2/kg^2\n");
    printf("17. Bohr
radius: a_0 = 0.53 x
10^-10 m\n");
    printf("18.
Electronvolt (eV): 1
eV = 1.60 x 10^-19
J\n");
    printf("19.
Magnetic flux quantum:
Î¦_0 = 2.07 x 10^-15
Wb\n");
    printf("20.
Reduced Planck's
constant: Ä§ =
h/(2Ï€)\n");

    break;
//conversions

    case 2:
    printf("Chemistry
Constants and
Units:\n\n");

    // Avogadro's
Number
    printf("1.
Avogadro's Number:
6.022 x 10^23 mol^-
1\n");

    // Speed of Light
    printf("2. Speed
of Light: 3.00 x 10^8
m/s\n");

    // Planck's
Constant
    printf("3.
Planck's Constant:
6.626 x 10^-34
JÂ·s\n");

    // Boltzmann
Constant
    printf("4.
Boltzmann Constant:
1.38 x 10^-23 J/K\n");

    // Gas Constant
    printf("5. Gas
Constant: 8.314
J/(molÂ·K)\n");

    // Faraday's
Constant
    printf("6.
Faraday's Constant:
96,485 C/mol\n");

    // Elementary
Charge
    printf("7.
Elementary Charge:
1.602 x 10^-19 C\n");

    // Gravitational
Constant
    printf("8.
Gravitational
Constant: 6.674 x 10^-
11 NÂ·m^2/kg^2\n");
```

```
    // Rydberg
Constant
    printf("9. Rydberg
Constant: 1.097 x 10^7
m^-1\n");

    // Universal Gas
Constant
    printf("10.
Universal Gas
Constant: 8.314
J/(molÂ·K)\n");

    // Electron Mass
    printf("11.
Electron Mass: 9.109 x
10^-31 kg\n");

    // Proton Mass
    printf("12. Proton
Mass: 1.673 x 10^-27
kg\n");

    // Neutron Mass
    printf("13.
Neutron Mass: 1.675 x
10^-27 kg\n");

    // Avogadro's
Number
    printf("14.
Avogadro's Number:
6.022 x 10^23 mol^-
1\n");

    // Stefan-
Boltzmann Constant
    printf("15.
Stefan-Boltzmann
Constant: 5.67 x 10^-8
W/(m^2Â·K^4)\n");

    // Coulomb's
Constant
    printf("16.
Coulomb's Constant:
8.99 x 10^9
NÂ·m^2/C^2\n");

    // Bohr Radius
    printf("17. Bohr
Radius: 0.529 x 10^-10
m\n");

    // Electron Charge
to Mass Ratio
    printf("18.
Electron Charge to
Mass Ratio: -1.76 x
10^11 C/kg\n");

    // Faraday's
Constant
    printf("19.
Faraday's Constant:
96,485 C/mol\n");

    // Vacuum
Permittivity (Electric
Constant)
    printf("20. Vacuum
Permittivity: 8.85 x
10^-12
C^2/(NÂ·m^2)\n");
    break;
    }
    return 0;
   // getche();
}
```

Challenges:-

Through working on this program we face many challenges like bugs, error. We spend around 60-70 hours working on this project because when we started we did not perform such a lengthy code and we try to make this code as much easier to understand by the third-party by adding comments in every possible step.

CONCLUSION: -

In conclusion, the scientific calculator project represents a milestone in computational versatility, seamlessly incorporating advanced mathematical functions with precision. Its user-friendly interface caters to diverse users, while rigorous testing ensures reliability. This project not only meets academic and scientific needs but also contributes to a broader understanding of programming complexities, making it a valuable tool for various applications and scientific needs but also contributes to a broader understanding of programming complexities, making it a valuable tool for various applications.