শিক্ষা নিয়ে গড়বো দেশ তথ্য–প্রযুক্তির বাংলাদেশ

Bangabandhu Sheikh Mujibur Rahman Digital University, Bangladesh

Software Development Project-01

COURSE NO. PROG 112: Object Oriented Programming Sessional

SUBMITTED BY

Name-Mobashira Mehajabin Arpita(2101008)
Srabonee Paul Tuli(2101015)
Mehrin Farzana(2101013)
Department of IRE
Session :2021-2022
Bangabandhu Sheikh Mujibur Rahman Digital University, Bangladesh

SUBMITTED TO

Md.Toukir Ahmed
Lecturer
Department of IRE, BDU
Bangabandhu Sheikh Mujibur Rahman Digital University, Bangladesh

Date of Submission: 24 August,2023

Project title: Scientific calculator.

Name of team members	Portion of code in the project
1. Mobashira Mehajabin Arpita	(10-265) & (698-714)
2. Srabonee Paul Tuli	(266-326), (572-611) & (764-882)
3. Mehrin Farzana	(327-571), (613-695), (758-763) & (883-888)

Code:

```
// Include necessary header files
#include <iostream>
#include <vector>
#include<cmath>
#include <string>
#include <ctype.h>
#include <windows.h>
#include <bits/stdc++.h>
// Define macros for sleep and PI
#define sleep(x)Sleep(x*1000)
#define PI 3.141592654

using namespace std;
// Define a class for basic arithmetic operations
class arithmetic{
private:
   vector<float> numbers;
   double a;
protected:
   // Function to input numbers and operations
   void getdata(){
   char choice;
   cout << "For final result enter '='" << endl;
   cout << endl;
   cout << "Enter numbers and operations: " << endl;
   while (choice != '=') {
      float num;
      cin >> num;
      addNumber(num);
      cin >> choice;
   }
   }
   // Function to add a number to the list
   void addNumber(float num) {
      numbers.push_back(num);
   }

   // Function to calculate the sum of numbers
```


^

```
    void sumNumbers() {
        float sum = 0;
        for (int i = 0; i < numbers.size(); i++) {
            sum += numbers[i];
        }
        cout << "Sum of the entered numbers: " << sum << endl;
    }
    // Function to subtract numbers
    void subtractNumbers() {
        float result = numbers[0];
        for (int i = 1; i < numbers.size(); i++) {
            result -= numbers[i];
        }
        cout << "Result of subtraction: " << result << endl;
    }

     // Function to multiply numbers
    void multiplyNumbers() {
        float result = 1;
        for (int i = 0; i < numbers.size(); i++) {
            result *= numbers[i];
        }
        cout << "Result of multiplication: " << result << endl;
    }

     // Function to divide numbers
    void divideNumbers() {
        float result = numbers[0];
        for (int i = 1; i < numbers.size(); i++) {
            if (numbers[i] == 0) {
                cout << "Error: Division by zero!" << endl;
                return;
            }
            result /= numbers[i];
        }
        cout << "Result of division: " << result << endl;
    }
};

// Define a class for arithmetic calculations with user input
class Arith : public arithmetic{
public:
// Constructor to handle user input and perform calculations
     Arith(){
    int operation;
    cout << "1. Addition" << endl;
    cout << "2. Subtraction" << endl;
    cout << "3. Multiplication" << endl;
    cout << "4. Division" << endl;
    cout << "Type the initial number of your selection:";
    cin >> operation;
```


^

```
    switch (operation) {
        case 1:
            {getdata();
            sumNumbers();
            break;}
        case 2:
            {getdata();
            subtractNumbers();
            break;}
        case 3:
            {getdata();
            multiplyNumbers();
            break;}
        case 4:
            {getdata();
            divideNumbers();
            break;}
    }
}
};
// Define a class for trigonometric operations
class trigonometric{
private :
    double n,n1,result;
    long long n2,n3;
protected:
    void get_angle(){
    cout << "Enter angle in degrees : ";
    cin >> n1;
    }
    void COS(){
      double x = n1 * PI/180;
      result = cos(n1);;
    cout <<" cos(" << n1 << ") = " << result << endl;
    }
    void SIN(){
      double x = n1 * PI/180;
      result = sin(x);;
    cout <<" sin(" << n1 << ") = " << result << endl;
    }
    void TAN(){
    if(n1 ==90 || n1 ==270 || n1 ==450){
    printf("\nMath error!\n");}
    else{
      double x = n1 * PI/180;
      result = tan(x);;
    cout <<" tan(" << n1 << ") = " << result << endl;
    }
    }
    void arc_COS(){
        if(n1>1 || n1<-1){ // Parameter not in Range
```


^

```
        cout << "Math Error!!" <<endl;}
    else{
        result = acos(n1);
        double x=180/PI;
    cout <<" cos^-1(" << n1 << ") = " << result * x << endl;}
    }
    void arc_SIN(){
        if(n1>1 || n1<-1){// Parameter not in Range
        cout << "Math Error!!" <<endl;}
    else{
        result = asin(n1);
        double x=180/PI;
    cout <<"sin^-1(" << n1 << ") = " << result * x << endl;}
    }
    void arc_TAN(){
         if(n1>1 || n1<-1){ // Parameter not in Range
        cout << "Math Error!!" <<endl;}
    else{
        result = atan(n1);
        double x=180/PI;
    cout <<"tan^-1(" << n1 << ") = " << result * x << endl;}
    }
    long long fact_return(long long n){
    long long f=1;
    for(int i=1; i<=n; i++){
        f=f*i;
    }
    return f;
}

public:
    void get_value(){
    cout << "Enter number : ";
    cin >> n1;
    }

//Square Root
    void squareRoot(){
    double result=sqrt(n1);
    cout << "Square root of " << n1 << " is : " << result << endl;
    }

//Cube Root
    void CubeRoot(){
    result=cbrt(n1);
    cout << "Cube root of " << n1 << " is : " << result << endl;
    }

//log base 10
    void LOG_base(){
    if(n1 > 0){
```

```
    result = log10(n1);
    cout << "log("<< n1 <<") = "<< result << endl;}
    else{
      cout << "Math Error! " << endl;
    }
}

//Permutation: P(n,r) = n! ÷ (n-r)!
void npr(){
    cout << "Enter n :";
    cin >> n2;
    cout << "Enter r :";
    cin >> n3;
    long long per = fact_return(n2) / fact_return(n2-n3);
    cout << "\nnPr = " << per << endl;

}

//Combination: nCr = n!/[r! (n-r)!]
void ncr(){
    cout << "Enter n :";
    cin >> n2;
    cout << "Enter r :";
    cin >> n3;
    long long comb = fact_return(n2) / (fact_return(n3) * fact_return(n2-n3));
    cout << "\nnCr = " << comb << endl;

}

//Power
void power(){
    cout << "Enter base :";
    cin >> n1;
    cout << "Enter power :";
    cin >> n;
    result = pow(n1,n);
    cout << "\n(" << n1 <<")^" << n <<" = " << result << endl;

}

};
// Define a class for trigonometric calculations with user input
class Trigono : public trigonometric{
public:
    Trigono(){
    int b;
    cout << "1. Sine" << endl;
    cout << "2. Cosine" << endl;
    cout << "3. Tangent" << endl;
    cout << "4. Sine Inverse" << endl;
    cout << "5. Cosine Inverse" << endl;
```

^

```
    cout << "6. Tangent Inverse" << endl;
    cout << "Type the initial number of your selection:";
    cin >> b;
    switch(b){
    case 1:{
         get_angle();
         SIN();
        break;}
     case 2:{
        get_angle();
        COS();
        break;}
     case 3:{
        get_angle();
        TAN();
        break;}
     case 4:{
        get_value();
        arc_SIN();
        break;}
     case 5:{
        get_value();
        arc_COS();
        break;}
     case 6:{
       get_value();
       arc_TAN();
        break;}
     default:
        cout << "Invalid Choice !"<< endl;
        break;
    }
}

};
// Define a class for matrix operations
class Matrix {
private:
    vector<vector<double>> data;

public:
    Matrix(int rows, int cols) : data(rows, vector<double>(cols)) {}

    void setData() {
        cout << "Enter matrix elements:" << endl;
        for (int i = 0; i < data.size(); i++) {
            for (int j = 0; j < data[i].size(); j++) {
                cin >> data[i][j];
            }
        }
    }
```

```
    Matrix operator+(const Matrix& other) const {
        if (data.size() != other.data.size() || data[0].size() != other.data[0].size()) {
            throw runtime_error("Matrix dimensions don't match for addition");
        }

        Matrix result(data.size(), data[0].size());

        for (int i = 0; i < data.size(); i++) {
            for (int j = 0; j < data[i].size(); j++) {
                result.data[i][j] = data[i][j] + other.data[i][j];
            }
        }

        return result;
    }

    Matrix operator*(const Matrix& other) const {
        if (data[0].size() != other.data.size()) {
            throw runtime_error("Matrix dimensions don't match for multiplication");
        }

        Matrix result(data.size(), other.data[0].size());

        for (int i = 0; i < data.size(); i++) {
            for (int j = 0; j < other.data[0].size(); j++) {
                double sum = 0;
                for (int k = 0; k < data[0].size(); k++) {
                    sum += data[i][k] * other.data[k][j];
                }
                result.data[i][j] = sum;
            }
        }
        return result;
    }

    void print() const {
        for (const auto& row : data) {
            for (const double element : row) {
                cout << element << " ";
            }
            cout << endl;
        }
    }
};
// Define a class for cryptography operations
class Cryptography{

        char message[101];
        short int key;
```

```
public:
    Cryptography(){
        int b;
        cout << "1. Encrypt" << endl;
        cout << "2. Decrypt" << endl;
        cout << "Type the initial number of your selection:";
        cin >> b;
        switch(b){
        case 1:{
            encrypt();
            break;}
        case 2:{
            decrypt();
            break;}
        default:
            cout << "Invalid Choice !"<< endl;
            break;
        }
    }
    void encrypt(){
        printf("Enter non-spaced-msg<ENTER>integer key\n-------------------------------------\n");
        scanf(" %[^\n]s", message);
        scanf("%hd", &key);

        for(int i=0; message[i]!='\0'; i++){
            //if u also wish to encrypt the space then comment this & the below 'if' out
            if(message[i]==' ' && message[i+1]!='\0') i++;
            if(message[i]>=97)
                message[i] = ((((message[i]%97)+key)%26)+97-32);
            else if(message[i]<97)
                message[i] = ((((message[i]%65)+key)%26)+97-32);
        }

        printf("Encrypted: %s\n", message);
    }

    void decrypt(){
        printf("Enter non-spaced-msg<ENTER>integer key\n-------------------------------------\n");
        scanf(" %[^\n]s", message);
        scanf("%hd", &key);
        for(int i=0; message[i]!='\0'; i++){
            message[i]=toupper(message[i]);
            //if u also wish to encrypt the space then comment this and the above 'if' out
            if(message[i]==' ' && message[i+1]!='\0') i++;
            if(((message[i]%65)-key)<0)
                message[i] = (((message[i]%65)-key)+26)+97;
            else if(((message[i]%65)-key)>=0)
                message[i] = ((message[i]%65)-key)+97;
        }
        printf("Decrypted: %s\n", message);
    }
```

```
};

// Define a class for number system conversions
class NumberSystem{
   protected:
      string num;

   public:

      NumberSystem(){
         int b;
         cout << "1. Binary to Decimal" << endl;
         cout << "2. Decimal to Binary" << endl;
         cout << "3. Decimal to Octal" << endl;
         cout << "4. Decimal to Hexadecimal" << endl;
         cout << "5. Octal to Binary" << endl;
         cout << "6. Hexadecimal to Binary" << endl;
         cout << "7. Octal to Hexadecimal" << endl;
         cout << "8. Hexadecimal to Octal" << endl;
         cout << "9. Binary to Hexadecimal" << endl;
         cout << "10. Binary to Octal" << endl;
         cout << "Type the initial number of your selection:";
         cin >> b;
         switch(b){
         case 1:{
            BinToDec();
            break;}
         case 2:{
            DecToBin();
            break;}
         case 3:{
            DecToOc();
            break;}
         case 4:{
            DecToHex();
            break;}
         case 5:{
            OcToBin();
            break;}
         case 6:{
            HexToBin();
            break;}
         case 7:{
            OcToHex();
            break;}
         case 8:{
            HexToOc();
            break;}
         case 9:{
            BinToHex();
            break;}
```

```
    case 10:{
        BinToOc();
        break;}
    default:
        cout << "Invalid Choice !"<< endl;
        break;
    }
}

void BinToDec(){
    string b;
    cout<<"Binary: ";
    cin>>b;
    int d = stoi(b, nullptr, 2);
    cout<<"Decimal: "<<d;
}

void DecToBin(){
    int num;
    cout<<"Decimal = ";
    cin>>num;
    string s;
    int count=0;

    for(int i=num; i>0; i/=2){
        if(i%2)
            s+='1';
        else if(!(i%2) && i!=0)
            s+='0';
        count++;
    }

    cout<<"Binary: ";
    for(int i=count; i>=0; i--)
        cout<<s[i];
}

void DecToOc(){
    int n;
    cout<<"Decimal: ";
    cin>>n;
    cout<<"Octal: ";
    cout << oct << n;
}

void DecToHex(){
    int n ;
    cout<<"Decimal: ";
    cin>>n;
    cout<<"Hexadecimal: ";
    cout << hex << n;
```

```
}

void OcToDec(){
    string o;
    cout<<"Octal: ";
    cin>>o;
    int d = stoi(o, nullptr, 8);
    cout<<"Decimal: "<<d;
}

void HexToDec(){
    string h;
    cout<<"Hexadecimal: ";
    cin>>h;
    int d = stoi(h, nullptr, 16);
    cout<<"Decimal: "<<d;
}

void OcToBin() {
    string octal;
    cout<<"Octal: ";
    cin>>octal;
    int d = stoi(octal, nullptr, 8);

    string s;
    int count=0;

    for(int i=d; i>0; i/=2){
        if(i%2)
            s+='1';
        else if(!(i%2) && i!=0)
            s+='0';
        count++;
    }

    cout<<"Binary: ";
    for(int i=count; i>=0; i--)
        cout<<s[i];
}

void HexToBin()
{
    string hex;
    cout<<"Hexadecimal: ";
    cin>>hex;
    string result = bitset<16>(stoul(hex, nullptr, 16)).to_string();
    cout<<"Binary: "<< result;
}

void OcToHex(){
    string o;
```

```
        cout<<"Octal: ";
        cin>>o;
        int d = stoi(o, nullptr, 8);
        cout<<"Hexadecimal: ";
        cout << hex << d;
    }

    void HexToOc(){
        string h;
        cout<<"Hexadecimal: ";
        cin>>h;
        int d = stoi(h, nullptr, 16);
        cout<<"Octal: ";
        cout << oct << d;
    }

    void BinToHex(){
        string b;
        cout<<"Binary: ";
        cin>>b;
        int d = stoi(b, nullptr, 2);
        cout<<"Hexadecimal: ";
        cout << hex << d;
    }

    void BinToOc(){
        string b;
        cout<<"Binary: ";
        cin>>b;
        int d = stoi(b, nullptr, 2);
        cout<<"Octal: ";
        cout << oct << d;
    }

};
// Define a class for complex number operations
class ComplexNumber {
private:
    double real;
    double imaginary;

public:
    ComplexNumber(double r = 0, double i = 0) : real(r), imaginary(i) {}

    ComplexNumber operator+(const ComplexNumber& other) const {
        return ComplexNumber(real + other.real, imaginary + other.imaginary);
    }

    ComplexNumber operator-(const ComplexNumber& other) const {
        return ComplexNumber(real - other.real, imaginary - other.imaginary);
    }
```

```
    ComplexNumber operator*(const ComplexNumber& other) const {
        double newReal = real * other.real - imaginary * other.imaginary;
        double newImaginary = real * other.imaginary + imaginary * other.real;
        return ComplexNumber(newReal, newImaginary);
    }

    void display() const {
        cout << real;
        if (imaginary >= 0) {
            cout << " + " << imaginary << "i";
        } else {
            cout << " - " << -imaginary << "i";
        }
    }
};
// Define a class for complex number calculations with user input
class ComplexCalculator {
    // ... Existing ComplexCalculator class code ...
};
// Define a class for complex arithmetic calculations
class ComplexArith : public ComplexCalculator, public arithmetic {
    // ... Existing ComplexArith class code ...
};

// Function to perform bitwise operations
void BitWiseOper(){
    int operation,j,n;
    char s[5];
    printf("enter operation:\n '1' for Shifting,\n '2' for Adding,\n '3' for Removing,\n '4' Checking,\n '5' for
Toggling,\n '6' for finding LSB (least significant bit)\n '7' for finding MSB (most significant bit)\n");
    printf("Enter: ");
    scanf("%d", &operation);
    if(operation==1){
        printf("To Left shift, , input format: 'your number' << '#rooms'\nTo Right shift, , input format: 'your
number' >> '#rooms'\n");
    scanf(" %d%[<> ]%d",&n, s, &j);
    if(s[0]=='<' & s[1]=='<' || s[1]=='<' & s[2]=='<')
        printf("%d\n", n<<j);
    else if(s[0]=='>' & s[1]=='>' || s[1]=='>' & s[2]=='>')
        printf("%d\n", n>>j);
    else
         printf("Invalid!");


    }

    else if(operation==2){
        printf("Enter the number & the position of which bit to add(from right to left, starts at 0):\n");
        scanf("%d%d", &n, &j);
        printf("%d\n", n|(1<<(j)));
```

```
    }
    else if(operation==3){
        printf("Enter the number & the position of which bit to remove(from right to left, starts at 0):\n");
        scanf("%d%d", &n, &j);
        printf("%d\n", n&(~(1<<(j))));
    }

    else if(operation==4){
        printf("Enter the number & the position of which bit to check(from right to left, starts at 0):\n");
        scanf("%d%d", &n, &j);
        printf("%d\n", n&(1<<(j)));
    }

    else if(operation==5){
        printf("Enter the number & the position of which bit to toggle(from right to left, starts at 0):\n");
        scanf("%d%d", &n, &j);
        printf("%d\n", n^(1<<(j)));
    }

    else if(operation==6){
        printf("Enter the number: ");
        scanf("%d", &n);
        printf("%d\n", n&(-n));
    }
    else if(operation==7){
        long int n, msb;
        printf("Enter the number: ");
        scanf("%ld", &n);
        msb = 1<<((int)(log(n)/log(2)));
        printf("%d", msb);
    }
    else
     printf("Invalid!");

}
// Function to check if a number is prime
void prime(){
    int n;
    cout<<"Enter 0 to break out.\n";
    while(1){
    cout<<"Enter number: ";
    scanf(" %d", &n);
    if(n==0) break;
    else if(n==1)
        printf("Not Prime.\n");
    else if(n==2)
        printf("Prime!\n");
    for(int i=2; i<n; i++){
        if((n/i)<i && n%i!=0){
            printf("Prime!\n");
            break;
```

```
        }
        else if(n%i==0){
            printf("Not Prime. The least divisor: %d\n", i);
            break;
        }
    }
    }
}

// Main function
int main() {
    trigonometric c;
    int f=1;
    cout <<"\t\t**SCIENTIFIC CALCULATOR**\n\n" << endl;
    while(f!=0){
    int choice;

    sleep(1);
    cout << "\n\n0- !EXIT!\n" << endl;
    cout << "1- ARITHMETIC CALCULATIONS\n" << endl;
    cout << "2- TRIGONOMETRIC FUNCTIONS\n" << endl;
    cout << "3- Permutation (nPr)\n" << endl;
    cout << "4- Combination (nCr)\n" << endl;
    cout << "5- Power function\n" << endl;
    cout << "6- log(base 10)\n" << endl;
    cout << "7- Square Root\n" << endl;
    cout << "8- Cube Root\n" << endl;
    cout << "9- CRYPTOGRAPHY\n" << endl;
    cout << "10- Number conversion\n" << endl;
    cout << "11- Matrix Addition\n" << endl;
    cout << "12- Matrix Multiplication\n" << endl;
    cout << "13- Complex Number Addition\n" << endl;
    cout << "14- Complex Number Subtraction\n" << endl;
    cout << "15- Complex Number Multiplication\n" << endl;
    cout << "16- Bit-wise Operations\n" << endl;
    cout << "17- Is-it-Prime?\n" << endl;
    cout << "Type the initial number of your selection:";
    cin >> choice;
    switch(choice){
    case 0:{
     f=0;
     break;
    }
    case 1:{
       Arith s;
       break;}
    case 2:{
     Trigono t;
     break;}
     case 3:{
      c.npr();
```

```
break;}
case 4:{
 c.ncr();
break;}
case 5:{
 c.power();
break;}
case 6:{
 c.get_value();
 c.LOG_base();
break;}
case 7:{
 c.get_value();
 c.squareRoot();
break;}
case 8:{
c.get_value();
c.CubeRoot();
break;}
case 9:{
Cryptography c;
break;}
case 10:{
NumberSystem n;
break;}
 case 11: {
      int rows, cols;
      cout << "Enter number of rows and columns for matrices:" << endl;
      cin >> rows >> cols;

      Matrix mat1(rows, cols);
      Matrix mat2(rows, cols);

      cout << "Enter elements for Matrix 1:" << endl;
      mat1.setData();

      cout << "Enter elements for Matrix 2:" << endl;
      mat2.setData();

      cout << "Matrix 1:" << endl;
      mat1.print();

      cout << "Matrix 2:" << endl;
      mat2.print();

      try {
         Matrix result = mat1 + mat2;
         cout << "Result of Matrix Addition:" << endl;
         result.print();
      } catch (const runtime_error& e) {
         cout << e.what() << endl;
```

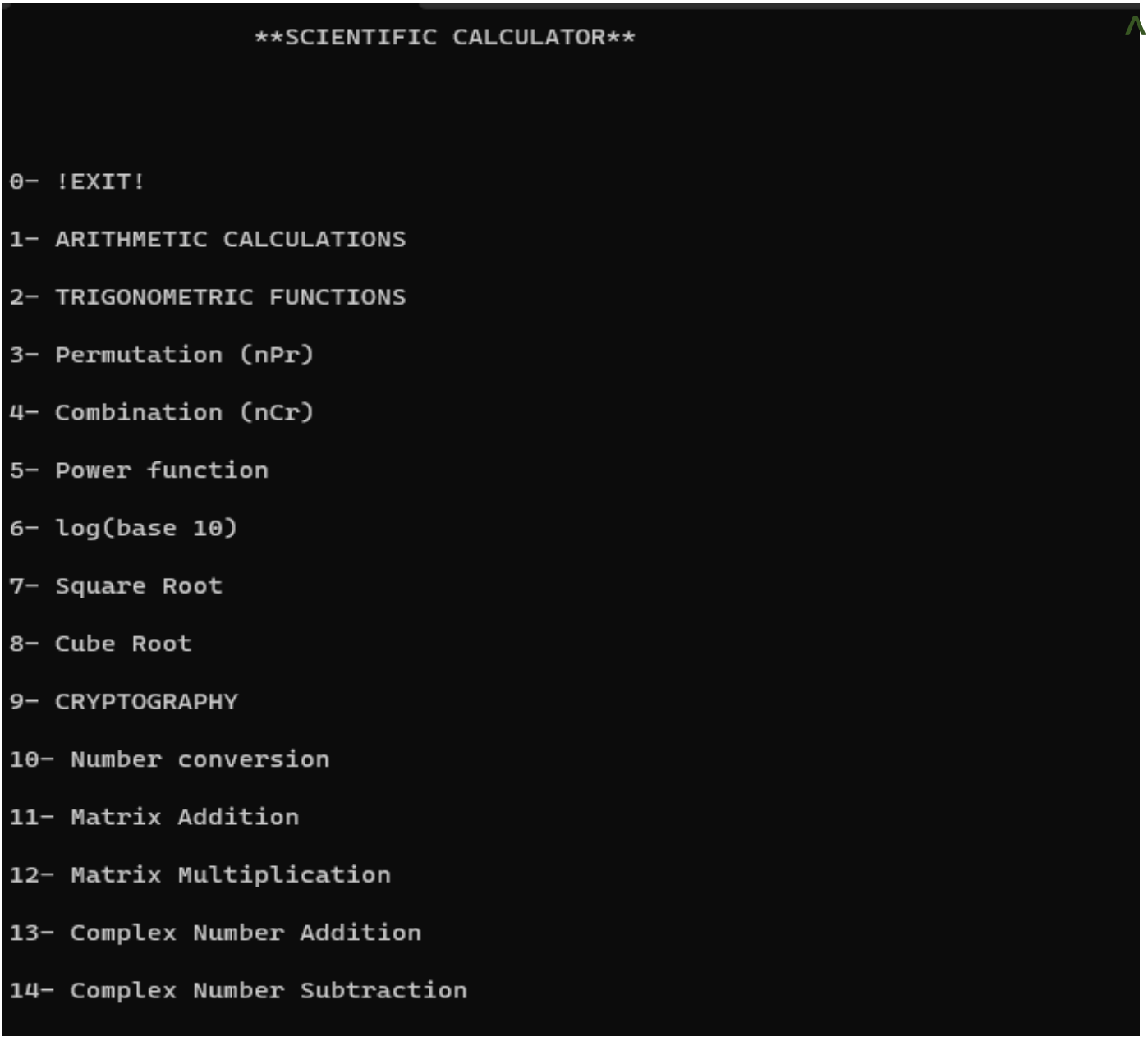
^

```
    }

    break;
}

case 12: {
    int rows1, cols1, rows2, cols2;
    cout << "Enter number of rows and columns for Matrix 1:" << endl;
    cin >> rows1 >> cols1;

    cout << "Enter number of rows and columns for Matrix 2:" << endl;
    cin >> rows2 >> cols2;

    Matrix mat1(rows1, cols1);
    Matrix mat2(rows2, cols2);

    cout << "Enter elements for Matrix 1:" << endl;
    mat1.setData();

    cout << "Enter elements for Matrix 2:" << endl;
    mat2.setData();

    cout << "Matrix 1:" << endl;
    mat1.print();

    cout << "Matrix 2:" << endl;
    mat2.print();

    try {
        Matrix result = mat1 * mat2;
        cout << "Result of Matrix Multiplication:" << endl;
        result.print();
    } catch (const runtime_error& e) {
        cout << e.what() << endl;
    }

    break;
}
case 13: {
    double real1, imag1, real2, imag2;
    cout << "Enter real and imaginary parts of Complex Number 1:" << endl;
    cin >> real1 >> imag1;
    cout << "Enter real and imaginary parts of Complex Number 2:" << endl;
    cin >> real2 >> imag2;

    ComplexNumber num1(real1, imag1);
    ComplexNumber num2(real2, imag2);

    ComplexArith ca;
    ComplexNumber result = num1 + num2;
```

```
        cout << "Result of Complex Number Addition: ";
        result.display();
        cout << endl;
        break;
    }

    case 14: {
        double real1, imag1, real2, imag2;
        cout << "Enter real and imaginary parts of Complex Number 1:" << endl;
        cin >> real1 >> imag1;
        cout << "Enter real and imaginary parts of Complex Number 2:" << endl;
        cin >> real2 >> imag2;

        ComplexNumber num1(real1, imag1);
        ComplexNumber num2(real2, imag2);

        ComplexArith ca;
        ComplexNumber result = num1 - num2;

        cout << "Result of Complex Number Subtraction: ";
        result.display();
        cout << endl;
        break;
    }
    case 15: {
        double real1, imag1, real2, imag2;
        cout << "Enter real and imaginary parts of Complex Number 1:" << endl;
        cin >> real1 >> imag1;
        cout << "Enter real and imaginary parts of Complex Number 2:" << endl;
        cin >> real2 >> imag2;

        ComplexNumber num1(real1, imag1);
        ComplexNumber num2(real2, imag2);

        ComplexArith ca;
        ComplexNumber result = num1 * num2;

        cout << "Result of Complex Number Multiplication: ";
        result.display();
        cout << endl;
        break;
    }
    case 16:{
    BitWiseOper();
    break;}
    case 17:{
    prime();
    break;}
default:
cout << "Invalid Choice !"<< endl;
break;
```

```
    }
   }
   return 0;
}
```

Output:



```
15- Complex Number Multiplication

16- Bit-wise Operations

17- Is-it-Prime?

Type the initial number of your selection:
```

ARITHMETIC CALCULATIONS:

```
Type the initial number of your selection:1
1. Addition
2. Subtraction
3. Multiplication
4. Division
Type the initial number of your selection:3
For final result enter '='

Enter numbers and operations:
3*4*5*6*2=
Result of multiplication: 720
0- !EXIT!

1- ARITHMETIC CALCULATIONS

2- TRIGONOMETRIC FUNCTIONS

3- Permutation (nPr)

4- Combination (nCr)

5- Power function

6- log(base 10)

7- Square Root

8- Cube Root

Type the initial number of your selection:
```

TRIGONOMETRIC FUNCTIONS:

```
Type the initial number of your selection:2
1. Sine
2. Cosine
3. Tangent
4. Sine Inverse
5. Cosine Inverse
6. Tangent Inverse
Type the initial number of your selection:2
Enter angle in degrees : 45
 cos(45) = 0.525322
```

Permutation (nPr):

```
0- !EXIT!
1- ARITHMETIC CALCULATIONS
2- TRIGONOMETRIC FUNCTIONS
3- Permutation (nPr)
4- Combination (nCr)
5- Power function
6- log(base 10)
7- Square Root
8- Cube Root
Type the initial number of your selection:3
Enter n :5
Enter r :3
nPr = 60
```

Combination (nCr):

```
0- !EXIT!
1- ARITHMETIC CALCULATIONS
2- TRIGONOMETRIC FUNCTIONS
3- Permutation (nPr)
4- Combination (nCr)
5- Power function
6- log(base 10)
7- Square Root
8- Cube Root
Type the initial number of your selection:4
Enter n :5
Enter r :3
```

Power function:

```
0- !EXIT!

1- ARITHMETIC CALCULATIONS

2- TRIGONOMETRIC FUNCTIONS

3- Permutation (nPr)

4- Combination (nCr)

5- Power function

6- log(base 10)

7- Square Root

8- Cube Root

Type the initial number of your selection:5
Enter base :23
Enter power :4

(23)^4 = 279841
```

log(base 10):

```
0- !EXIT!

1- ARITHMETIC CALCULATIONS

2- TRIGONOMETRIC FUNCTIONS

3- Permutation (nPr)

4- Combination (nCr)

5- Power function

6- log(base 10)

7- Square Root

8- Cube Root

Type the initial number of your selection:6
Enter number : 2
log(2) = 0.30103
```

Square Root:

```
0- !EXIT!

1- ARITHMETIC CALCULATIONS

2- TRIGONOMETRIC FUNCTIONS

3- Permutation (nPr)

4- Combination (nCr)

5- Power function

6- log(base 10)

7- Square Root

8- Cube Root

Type the initial number of your selection:7
Enter number : 88
Square root of 88 is : 9.38083
```

Cube Root:

```
0- !EXIT!

1- ARITHMETIC CALCULATIONS

2- TRIGONOMETRIC FUNCTIONS

3- Permutation (nPr)

4- Combination (nCr)

5- Power function

6- log(base 10)

7- Square Root

8- Cube Root

Type the initial number of your selection:8
Enter number : 58
Cube root of 58 is : 3.87088
```

Matrix Operations:

```
14- Complex Number Subtraction

15- Complex Number Multiplication

16- Bit-wise Operations

17- Is-it-Prime?

Type the initial number of your selection:11
Enter number of rows and columns for matrices:
3 2
Enter elements for Matrix 1:
Enter matrix elements:
3 6
9 2
1 5
Enter elements for Matrix 2:
Enter matrix elements:
7 10
5 7
3 8
Matrix 1:
3 6
9 2
1 5
Matrix 2:
7 10
5 7
3 8
Result of Matrix Addition:
10 16
14 9
4 13
```

```
16- Bit-wise Operations

17- Is-it-Prime?

Type the initial number of your selection:12
Enter number of rows and columns for Matrix 1:
2 3
Enter number of rows and columns for Matrix 2:
3 1
Enter elements for Matrix 1:
Enter matrix elements:
2 4 8
1 9 6
Enter elements for Matrix 2:
Enter matrix elements:
3
7
5
Matrix 1:
2 4 8
1 9 6
Matrix 2:
3
7
5
Result of Matrix Multiplication:
74
96
```

Complex number Operations:

```
11- Matrix Addition

12- Matrix Multiplication

13- Complex Number Addition

14- Complex Number Subtraction

15- Complex Number Multiplication

16- Bit-wise Operations

17- Is-it-Prime?

Type the initial number of your selection:15
Enter real and imaginary parts of Complex Number 1:
25 8
Enter real and imaginary parts of Complex Number 2:
5 2
Result of Complex Number Multiplication: 109 + 90i
```

```
11- Matrix Addition

12- Matrix Multiplication

13- Complex Number Addition

14- Complex Number Subtraction

15- Complex Number Multiplication

16- Bit-wise Operations

17- Is-it-Prime?

Type the initial number of your selection:13
Enter real and imaginary parts of Complex Number 1:
3 7
Enter real and imaginary parts of Complex Number 2:
9 12
Result of Complex Number Addition: 12 + 19i
```

Cryptography:

```
1. Encrypt
2. Decrypt
Type the initial number of your selection:1
Enter non-spaced-msg<ENTER>integer key
----------------------------------------
Salam
13
Encrypted: FNYNZ
```

```
1. Encrypt
2. Decrypt
Type the initial number of your selection:2
Enter non-spaced-msg<ENTER>integer key
----------------------------------------
fnynz
13
Decrypted: salam
```

Number System:

```
1. Binary to Decimal
2. Decimal to Binary
3. Decimal to Octal
4. Decimal to Hexadecimal
5. Octal to Binary
6. Hexadecimal to Binary
7. Octal to Hexadecimal
8. Hexadecimal to Octal
9. Binary to Hexadecimal
10. Binary to Octal
Type the initial number of your selection:1
Binary: 10101
Decimal: 21
```

```
1. Binary to Decimal
2. Decimal to Binary
3. Decimal to Octal
4. Decimal to Hexadecimal
5. Octal to Binary
6. Hexadecimal to Binary
7. Octal to Hexadecimal
8. Hexadecimal to Octal
9. Binary to Hexadecimal
10. Binary to Octal
Type the initial number of your selection:2
Decimal = 23
Binary: 10111
```

```
1. Binary to Decimal
2. Decimal to Binary
3. Decimal to Octal
4. Decimal to Hexadecimal
5. Octal to Binary
6. Hexadecimal to Binary
7. Octal to Hexadecimal
8. Hexadecimal to Octal
9. Binary to Hexadecimal
10. Binary to Octal
Type the initial number of your selection:3
Decimal: 21
Octal: 25
```

```
1. Binary to Decimal
2. Decimal to Binary
3. Decimal to Octal
4. Decimal to Hexadecimal
5. Octal to Binary
6. Hexadecimal to Binary
7. Octal to Hexadecimal
8. Hexadecimal to Octal
9. Binary to Hexadecimal
10. Binary to Octal
Type the initial number of your selection:4
Decimal: 21
Hexadecimal: 15
```

```
1. Binary to Decimal
2. Decimal to Binary
3. Decimal to Octal
4. Decimal to Hexadecimal
5. Octal to Binary
6. Hexadecimal to Binary
7. Octal to Hexadecimal
8. Hexadecimal to Octal
9. Binary to Hexadecimal
10. Binary to Octal
Type the initial number of your selection:5
Octal: 25
Binary: 10101
```

```
1. Binary to Decimal
2. Decimal to Binary
3. Decimal to Octal
4. Decimal to Hexadecimal
5. Octal to Binary
6. Hexadecimal to Binary
7. Octal to Hexadecimal
8. Hexadecimal to Octal
9. Binary to Hexadecimal
10. Binary to Octal
Type the initial number of your selection:6
Hexadecimal: a4f
Binary: 0000101001001111
```


^

```
1. Binary to Decimal
2. Decimal to Binary
3. Decimal to Octal
4. Decimal to Hexadecimal
5. Octal to Binary
6. Hexadecimal to Binary
7. Octal to Hexadecimal
8. Hexadecimal to Octal
9. Binary to Hexadecimal
10. Binary to Octal
Type the initial number of your selection:7
Octal: 25
Hexadecimal: 15
```

```
1. Binary to Decimal
2. Decimal to Binary
3. Decimal to Octal
4. Decimal to Hexadecimal
5. Octal to Binary
6. Hexadecimal to Binary
7. Octal to Hexadecimal
8. Hexadecimal to Octal
9. Binary to Hexadecimal
10. Binary to Octal
Type the initial number of your selection:8
Hexadecimal: 25
Octal: 45
```

```
1. Binary to Decimal
2. Decimal to Binary
3. Decimal to Octal
4. Decimal to Hexadecimal
5. Octal to Binary
6. Hexadecimal to Binary
7. Octal to Hexadecimal
8. Hexadecimal to Octal
9. Binary to Hexadecimal
10. Binary to Octal
Type the initial number of your selection:9
Binary: 10101
Hexadecimal: 15
```

```
1. Binary to Decimal
2. Decimal to Binary
3. Decimal to Octal
4. Decimal to Hexadecimal
5. Octal to Binary
6. Hexadecimal to Binary
7. Octal to Hexadecimal
8. Hexadecimal to Octal
9. Binary to Hexadecimal
10. Binary to Octal
Type the initial number of your selection:10
Binary: 10101
Octal: 25
```

Bit wise Operations:

```
enter operation:
 '1' for Shifting,
 '2' for Adding,
 '3' for Removing,
 '4' Checking,
 '5' for Toggling,
 '6' for finding LSB (least significant bit)
 '7' for finding MSB (most significant bit)
Enter: 1
To Left shift, , input format: 'your number' << '#rooms'
To Right shift, , input format: 'your number' >> '#rooms'
2<<3
16
```

```
enter operation:
 '1' for Shifting,
 '2' for Adding,
 '3' for Removing,
 '4' Checking,
 '5' for Toggling,
 '6' for finding LSB (least significant bit)
 '7' for finding MSB (most significant bit)
Enter: 1
To Left shift, , input format: 'your number' << '#rooms'
To Right shift, , input format: 'your number' >> '#rooms'
3>>1
1
```

```
enter operation:
 '1' for Shifting,
 '2' for Adding,
 '3' for Removing,
 '4' Checking,
 '5' for Toggling,
 '6' for finding LSB (least significant bit)
 '7' for finding MSB (most significant bit)
Enter: 2
Enter the number & the position of which bit to add(from right to left, starts at 0):
12 2
12
```

```
enter operation:
 '1' for Shifting,
 '2' for Adding,
 '3' for Removing,
 '4' Checking,
 '5' for Toggling,
 '6' for finding LSB (least significant bit)
 '7' for finding MSB (most significant bit)
Enter: 3
Enter the number & the position of which bit to remove(from right to left, starts at 0):
12
2
8
```


^

```
enter operation:
 '1' for Shifting,
 '2' for Adding,
 '3' for Removing,
 '4' Checking,
 '5' for Toggling,
 '6' for finding LSB (least significant bit)
 '7' for finding MSB (most significant bit)
Enter: 4
Enter the number & the position of which bit to check(from right to left, starts at 0):
12
2
4
```

```
enter operation:
 '1' for Shifting,
 '2' for Adding,
 '3' for Removing,
 '4' Checking,
 '5' for Toggling,
 '6' for finding LSB (least significant bit)
 '7' for finding MSB (most significant bit)
Enter: 5
Enter the number & the position of which bit to toggle(from right to left, starts at 0):
12
2
8
```

```
enter operation:
 '1' for Shifting,
 '2' for Adding,
 '3' for Removing,
 '4' Checking,
 '5' for Toggling,
 '6' for finding LSB (least significant bit)
 '7' for finding MSB (most significant bit)
Enter: 6
Enter the number: 12
4
```

```
enter operation:
 '1' for Shifting,
 '2' for Adding,
 '3' for Removing,
 '4' Checking,
 '5' for Toggling,
 '6' for finding LSB (least significant bit)
 '7' for finding MSB (most significant bit)
Enter: 7
Enter the number: 12
8
```

Is it prime:

```
Enter 0 to break out.
Enter number: 1
Not Prime.
Enter number: 2
Prime!
Enter number: 3
Prime!
Enter number: 4
Not Prime. The least divisor: 2
Enter number: 23
Prime!
Enter number: 45
Not Prime. The least divisor: 3
Enter number: 78
Not Prime. The least divisor: 2
Enter number: 91
Not Prime. The least divisor: 7
Enter number: 119
Not Prime. The least divisor: 7
Enter number: 17
Prime!
Enter number: 897
Not Prime. The least divisor: 3
Enter number: 0
```

GitHub URL: MelyFaj/CPP-project-CALCULATOR: 1.2 mid-term project, OOP (github.com)