Sri Shridevi Charitable Trust (R.)

SHRIDEVI INSTITUTE OF ENGINEERING AND TECHNOLOGY

Sira Road, Tumkur - 572 106, Karnataka, India.

Phone: 0816 - 2212629 | Principal: 0816 - 2212627, 9686114899 | Telefax: 0816 - 2212628

Email: info@shrideviengineering.org, principal@shrideviengineering.org | Website: www.shrideviengineering.org

(Approved by AICTE, New Delhi, Recognised by Govt. of Karnataka and Affiliated to Visvesvaraya Technological University, Belagavi)

ESTD:2002

LAB MANUAL

(As per CBCS Scheme 2022)

OOPS with C++ lab Manual

(BIS306B)

DEPARTMENT OF ISE

(Dept.of.ISE)

Prepared By:

Shamsiya parveen

Assistant Professor

Practical Component

Sl.NO	Experiments
1	Develop a C++ program to find the largest of three numbers
2	Develop a C++ program to sort the elements in ascending and descending order.
3	Develop a C++ program using classes to display student name, roll number, marks obtained in two subjects and total score of student
4	Develop a C++ program for a bank empolyee to print name of the employee, account_no. & balance. Print invalid balance if amount<500, Display the same, also display the balance after withdraw and deposit.
5	Develop a C++ program to demonstrate function overloading for the following prototypes. add(int a, int b) add(double a, double b
6	Develop a C++ program using Operator Overloading for overloading Unary minus operator.
7	Develop a C++ program to implement Multiple inheritance for performing arithmetic operation of two numbers
8	Develop a C++ program using Constructor in Derived classes to initialize alpha, beta and gamma and display corresponding values.
9	Develop a C++ program to create a text file, check file created or not, if created it will write some text into the file and then read the text from the file.
10	Develop a C++ program to write and read time in/from binary file using fstream
11	Develop a function which throws a division by zero exception and catch it in catch block. Write a C++ program to demonstrate usage of try, catch and throw to handle exception.
12	Develop a C++ program that handles array out of bounds exception using C++.

CIE for the practical component of the IPCC

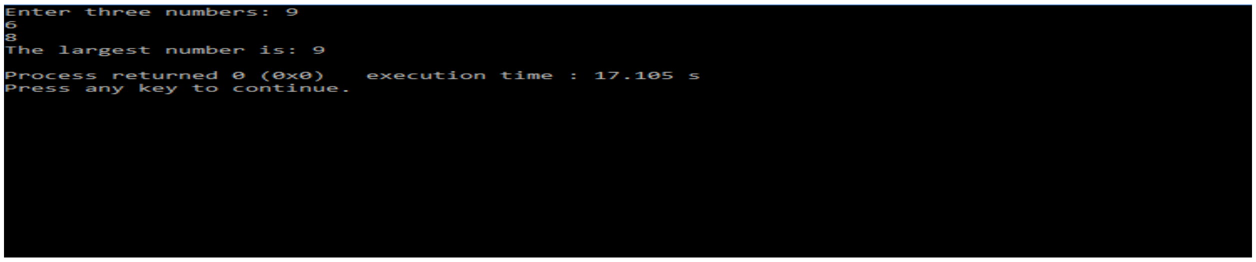
- **15 marks** for the conduction of the experiment and preparation of laboratory record, and **10 marks** for the test to be conducted after the completion of all the laboratory sessions.
- On completion of every experiment/program in the laboratory, the students shall be evaluated including viva-voce and marks shall be awarded on the same day.
- The CIE marks awarded in the case of the Practical component shall be based on the continuous evaluation of the laboratory report. Each experiment report can be evaluated for 10 marks. Marks of all experiments' write-ups are added and scaled down to **15 marks.**
- The laboratory test **(duration 02/03 hours)** after completion of all the experiments shall be conducted for 50 marks and scaled down to **10 marks.**
- Scaled-down marks of write-up evaluations and tests added will be CIE marks for the laboratory component of IPCC for **25 marks**.
- The student has to secure 40% of 25 marks to qualify in the CIE of the practical component of the IPCC.

The practical portion will have a CIE component only. Questions mentioned in the SEE paper may include questions from the practical component.

1.Develop a C++ program that finds the largest among three numbers:

```
#include <iostream>
using namespace std;
int main() {
int num1, num2, num3;
cout << "Enter three numbers: ";
cin >> num1 >> num2 >> num3;
 int largest = num1;
  if (num2 > largest) {
  largest = num2;
    }
    if (num3 > largest) {
        largest = num3;
    }
    cout << "The largest number is: " << largest << endl;
    return 0;
}
```

Output:

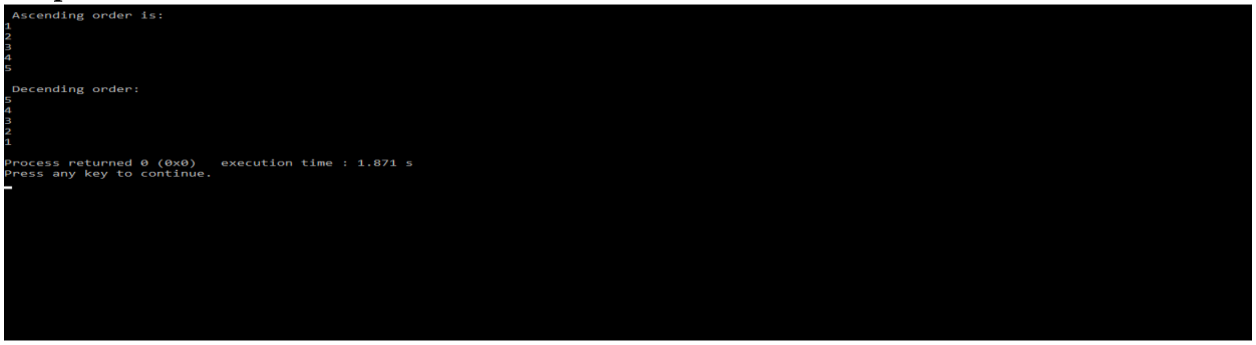


```
Enter three numbers: 9
6
8
The largest number is: 9

Process returned 0 (0x0)   execution time : 17.105 s
Press any key to continue.
```

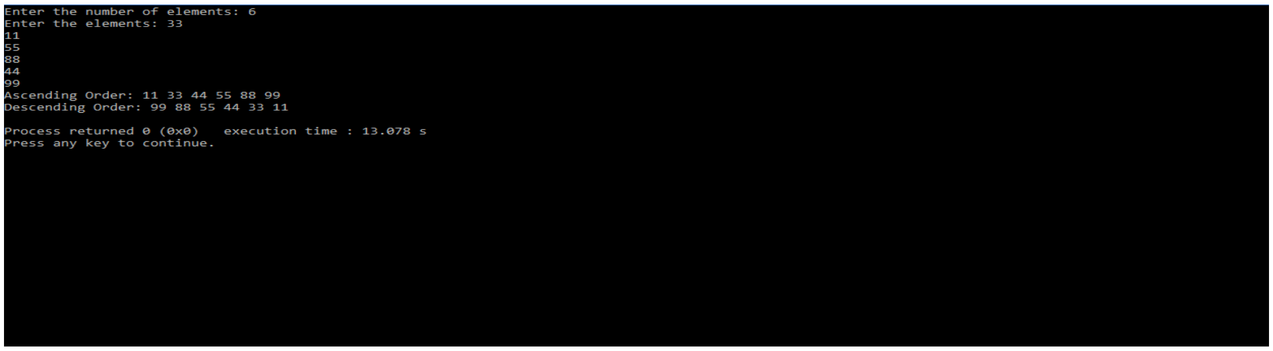
2. Develop a C++ program to sort the elements in ascending and descending order.

```
include<iostream>
using namespace std;
 int main()
 {
 int i,j,temp,a[5]={2,4,3,1,5};
 cout<<"\n Ascending order is: "<<endl;
 for(i=0;i<5;i++)
 {
     j=i+1;
     while(j<5)
     {
         while(a[i]>a[j])
     {
         temp=a[i];
         a[i]=a[j];
         a[j]=temp;
         j++;
     }
     j++;
     }
 }
 for(i=0;i<5;i++)
 cout<<a[i]<<endl;
 cout<<"\n Decending order: "<<endl;
 for(i=0;i<5;i++)
 {
j=i+1;
while(j<5)
 {
     while(a[i]<a[j])
     {
         temp=a[i];
         a[i]=a[j];
         a[j]=temp;
         j++;
     }
     j++;
     }
 }
 for(i=0;i<5;i++)
 cout<<a[i]<<endl;
  return 0;
 }
```

Output:



```
 Ascending order is:
1
2
3
4
5

 Decending order:
5
4
3
2
1

Process returned 0 (0x0)   execution time : 1.871 s
Press any key to continue.
```



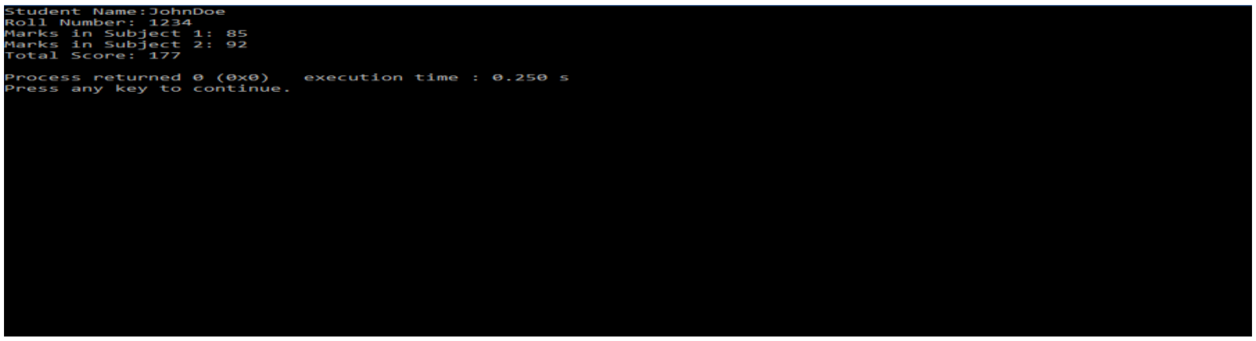
```
Enter the number of elements: 6
Enter the elements: 33
11
55
88
44
99
Ascending Order: 11 33 44 55 88 99
Descending Order: 99 88 55 44 33 11

Process returned 0 (0x0)   execution time : 13.078 s
Press any key to continue.
```

3.Develop a C++ program using classes to display student name, roll number, marks obtained in two subjects and total score of student

```
#include <iostream>
#include <string>
using namespace std;


class Student {
private:
    string name;
    int rollNumber;
    int marksSubject1;
    int marksSubject2;
public:
    void setDetails(const string& studentName, int studentRollNumber, int subject1Marks, int
subject2Marks) {
        name = studentName;
        rollNumber = studentRollNumber;
        marksSubject1 = subject1Marks;
        marksSubject2 = subject2Marks;
    }
    void displayDetails() {
        cout << "Student Name:"  << name << endl;
        cout << "Roll Number: " << rollNumber << endl;
        cout <<"Marks in Subject 1: " << marksSubject1 << endl;
        cout << "Marks in Subject 2: " << marksSubject2 << endl;
        int totalScore = marksSubject1 + marksSubject2;
        cout << "Total Score: " << totalScore << endl;
    }
};
```

```
int main() {
   // Create an instance of the Student class
   Student student;
   // Set student details
   student.setDetails("JohnDoe", 1234, 85, 92);
   // Display student details
   student.displayDetails();
   return 0;
}
}
```

Output:

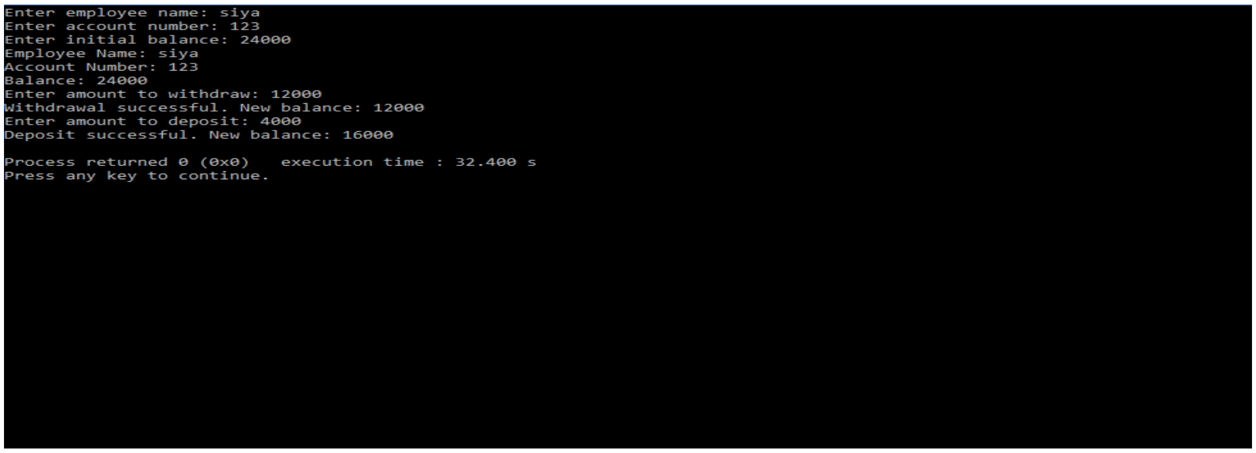


```
Student Name:JohnDoe
Roll Number: 1234
Marks in Subject 1: 85
Marks in Subject 2: 92
Total Score: 177

Process returned 0 (0x0)   execution time : 0.250 s
Press any key to continue.
```

4.Develop a C++ program for a bank empolyee to print name of the employee, account_no. & balance. Print invalid balance if amount<500, Display the same, also display the balance after withdraw and deposit.

```
#include <iostream>
#include <string>
using namespace std;
class BankEmployee {
private:
    string name;
    int accountNo;
    double balance;
public:
    BankEmployee(string empName, int accNo, double initialBalance) {
        name = empName;
        accountNo = accNo;
        balance = initialBalance;
    }
    void printDetails() {
        cout << "Employee Name: " << name << endl;
        cout << "Account Number: " << accountNo << endl;
        cout << "Balance: " << balance << endl;
    }
    void withdraw(double amount) {
        if (amount > balance) {
            cout << "Invalid balance. Insufficient funds." << endl;
        } else {
            balance -= amount;
            cout << "Withdrawal successful. New balance: " << balance << endl;
        }
    }

    void deposit(double amount) {
        balance += amount;
        cout << "Deposit successful. New balance: " << balance << endl;
    }
};
int main() {
    string employeeName;
    int accountNumber;
    double initialBalance;
    cout << "Enter employee name: ";
    getline(cin, employeeName);
    cout << "Enter account number: ";
    cin >> accountNumber;
    cout << "Enter initial balance: ";
    cin >> initialBalance;
    BankEmployee employee(employeeName, accountNumber, initialBalance);
    employee.printDetails();
    double withdrawAmount, depositAmount;
    cout << "Enter amount to withdraw: ";
    cin >> withdrawAmount;
    employee.withdraw(withdrawAmount);
    cout << "Enter amount to deposit: ";
    cin >> depositAmount;
    employee.deposit(depositAmount);
return 0; }
```

Output:

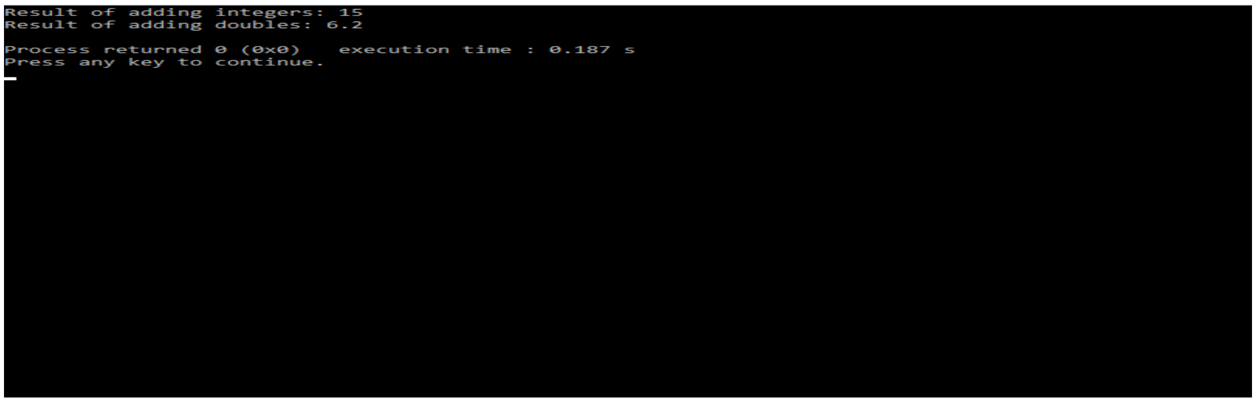


```
Enter employee name: siya
Enter account number: 123
Enter initial balance: 24000
Employee Name: siya
Account Number: 123
Balance: 24000
Enter amount to withdraw: 12000
Withdrawal successful. New balance: 12000
Enter amount to deposit: 4000
Deposit successful. New balance: 16000

Process returned 0 (0x0)   execution time : 32.400 s
Press any key to continue.
```

5.Develop a C++ program to demonstrate function overloading for the following prototypes. add(int a, int b) add(double a, double b

```
#include <iostream>
using namespace std;
int add(int a, int b) {
    return a + b;
}
double add(double a, double b) {
    return a + b;
}
int main() {
    int x = 5, y = 10;
    double p = 2.5, q = 3.7;
    int result1 = add(x, y);
    double result2 = add(p, q);
    cout << "Result of adding integers: " << result1 << endl;
    cout << "Result of adding doubles: " << result2 << endl;
    return 0;
}
```

Output:

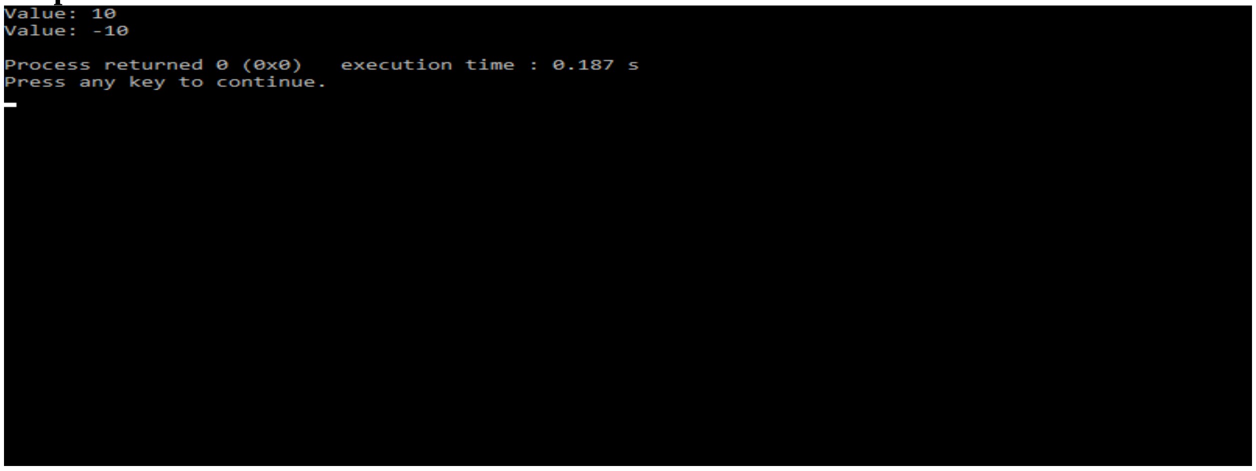


```
Result of adding integers: 15
Result of adding doubles: 6.2

Process returned 0 (0x0)   execution time : 0.187 s
Press any key to continue.
```

6.Develop a C++ program using Operator Overloading for overloading Unary minus operator.

```
#include <iostream>
using namespace std;

class Number {
private:
    int value;
public:
    Number(int val) {
        value = val;
    }

    Number operator-() {
        return Number(-value);
    }

    void display() {
        cout << "Value: " << value << endl;
    }
};

int main() {
    Number num1(10);
    Number num2 = -num1;

    num1.display(); // Output: Value: 10
    num2.display(); // Output: Value: -10

    return 0;
}
```

Output:

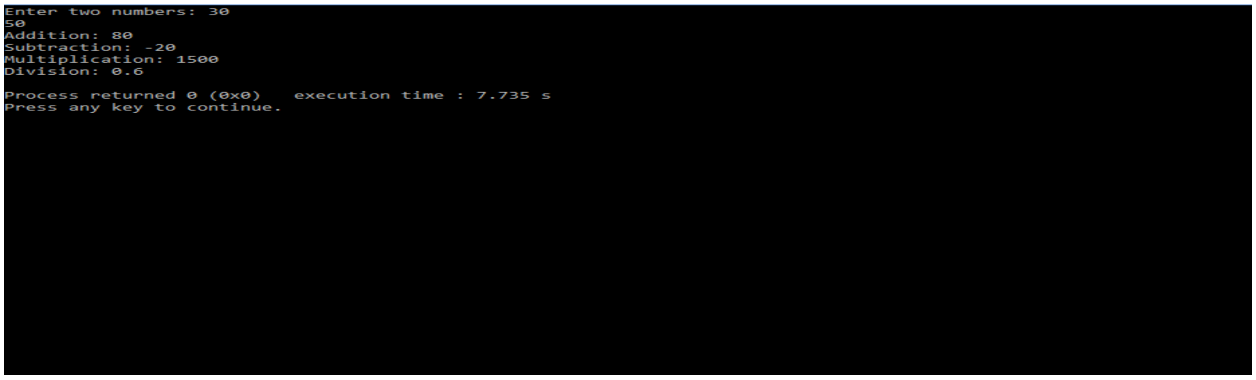


```
Value: 10
Value: -10

Process returned 0 (0x0)   execution time : 0.187 s
Press any key to continue.
```

7.Here is a C++ program that demonstrates multiple inheritance for performing arithmetic operations on two numbers:

```
#include <iostream>
using namespace std;

// Base class for arithmetic operations
class Arithmetic {
public:
    int add(int a, int b) {
        return a + b;
    }

    int subtract(int a, int b) {
        return a - b;
    }
};
// Derived class for multiplication
class Multiplication {
public:
    int multiply(int a, int b) {
        return a * b;
    }
};
// Derived class for division
class Division {
public:
    float divide(float a, float b) {
        return a / b;
    }
};
// Derived class that inherits from Arithmetic, Multiplication, and Division
class Calculator : public Arithmetic, public Multiplication, public Division {
public:
    void performOperations(int a, int b) {
        cout << "Addition: " << add(a, b) << endl;
        cout << "Subtraction: " << subtract(a, b) << endl;
        cout << "Multiplication: " << multiply(a, b) << endl;
       cout << "Division: " << divide(a, b) << endl;
    }
};
int main() {
    int num1, num2;
    cout << "Enter two numbers: ";
    cin >> num1 >> num2;
    Calculator calculator;
    calculator.performOperations(num1, num2);
    return 0;
}
```

Output:

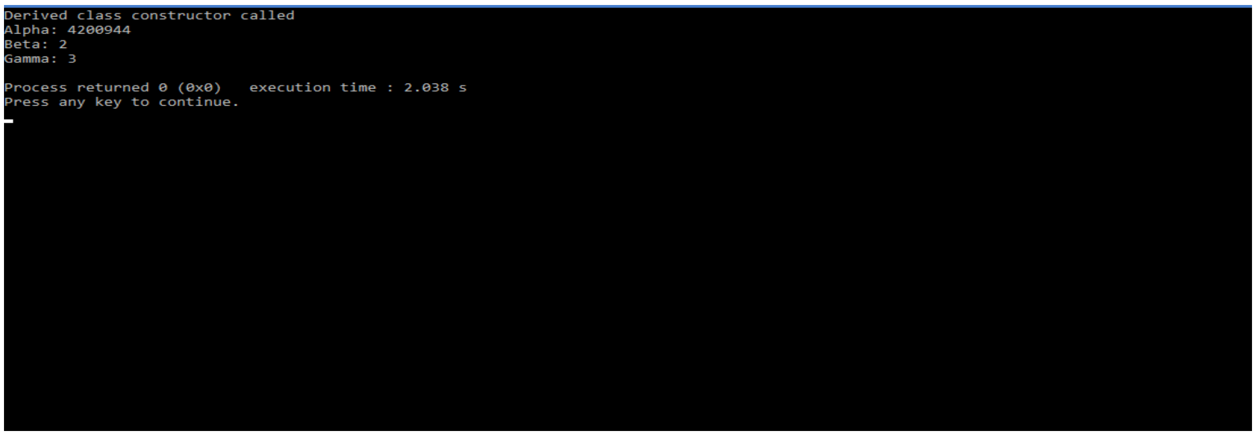


```
Enter two numbers: 30
50
Addition: 80
Subtraction: -20
Multiplication: 1500
Division: 0.6

Process returned 0 (0x0)   execution time : 7.735 s
Press any key to continue.
```

8.Develop a C++ program using Constructor in Derived classes to initialize alpha, beta and gamma and display corresponding values.

```
#include <iostream>
using namespace std;
class Base {
protected:
int alpha;
public:
Base(int a) {
alpha = a;
cout << "Base class constructor called" << endl;
}
};
class Derived : public Base {
protected:
int beta;
int gamma;
public:
Derived(int a, int b, int c)  {
beta = b;
gamma = c;
cout << "Derived class constructor called" << endl;
}
void displayValues() {
cout << "Alpha: " << alpha << endl;
cout << "Beta: " << beta << endl;
cout << "Gamma: " << gamma << endl;
}
};
int main() {
Derived obj(1, 2, 3);
obj.displayValues();
return 0;
}
```

Output:

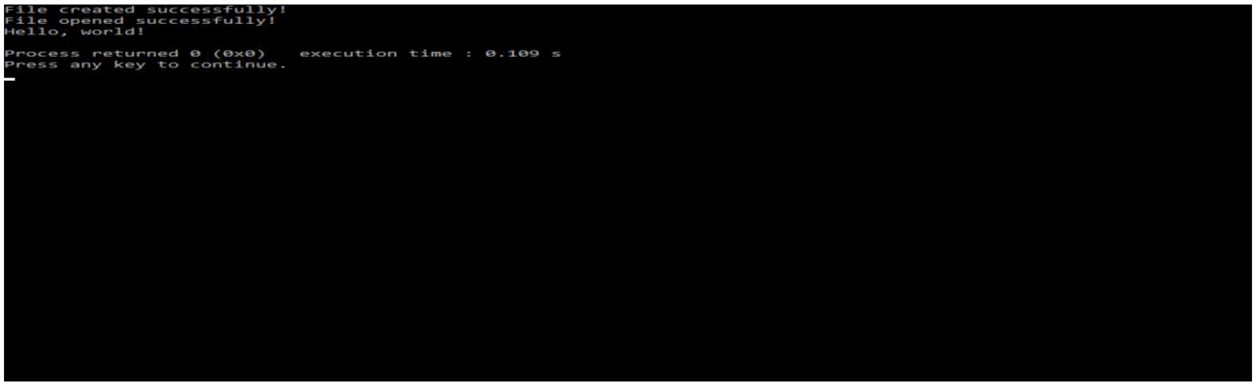


```
Derived class constructor called
Alpha: 4200944
Beta: 2
Gamma: 3

Process returned 0 (0x0)   execution time : 2.038 s
Press any key to continue.
```

9.Develop a C++ program to create a text file, check file created or not, if created it will write some text into the file and then read the text from the file.

```
#include <iostream>

#include <fstream>

using namespace std;

int main() {

// Creating and opening a file

ofstream outfile("example.txt");

// Checking if file was created successfully

if (outfile) {

cout << "File created successfully!" << std::endl;

// Writing some text into the file

outfile << "Hello, world!";

outfile.close();

// Reading the text from the file

ifstream infile("example.txt");

string line;

if (infile) {

cout << "File opened successfully!" << endl;

// Reading the text line by line

while (getline(infile, line)) {

cout << line << endl;

}

infile.close();

} else {

cout << "Error opening file!" << std::endl;

} } else {

cout << "Error creating file!" << std::endl;

} return 0;  }
```

Output:



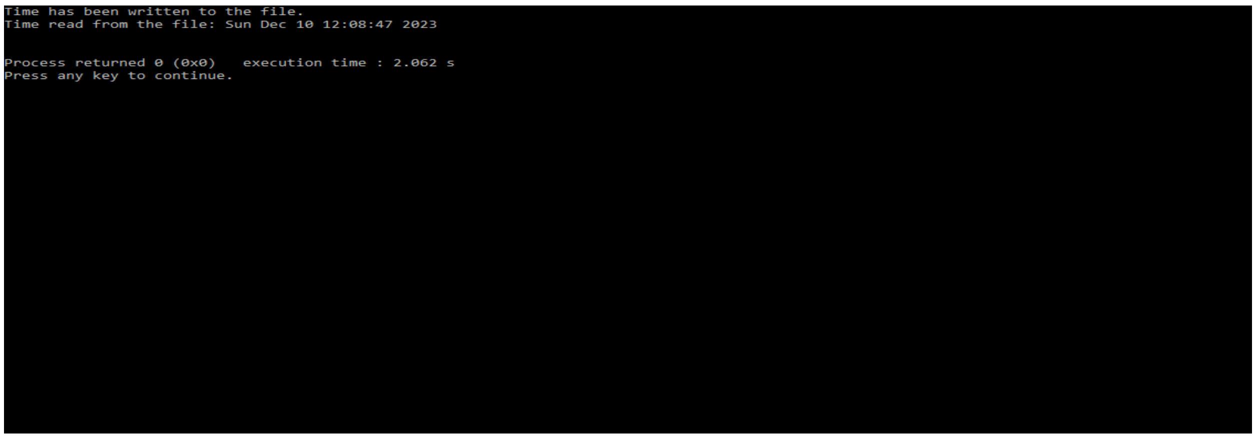
```
File created successfully!
File opened successfully!
Hello, world!

Process returned 0 (0x0)   execution time : 0.109 s
Press any key to continue.
```

10.Develop a C++ program to write and read time in/from binary file using fstream.

```
#include <iostream>
#include <fstream>
#include <ctime>
using namespace std;
void writeTimeToFile(const string& fileName) {
// Open the file in binary mode for writing
ofstream file(fileName, ios::binary);
if (file.is_open()) {
// Get the current time
time_t currentTime = time(nullptr);
// Write the time to the file
file.write((char*)&currentTime, sizeof(time_t));
// Close the file
file.close();
cout << "Time has been written to the file." << endl;
} else {
cout << "Failed to open the file for writing." << endl;
} }
void readTimeFromFile(const string& fileName) {
// Open the file in binary mode for reading
ifstream file(fileName, ios::binary);
if (file.is_open()) {
// Read the time from the file
time_t storedTime;
file.read((char*)&storedTime, sizeof(time_t));
// Check if reading was successful
if (file) {
```

```
// Convert the stored time to a readable string and display it
string timeString = ctime(&storedTime);
cout << "Time read from the file: " << timeString << endl;
} else {
cout << "Failed to read the time from the file." << endl;
}
// Close the file
file.close();
} else {
cout << "Failed to open the file for reading." << endl;
}
}
int main() {
string fileName = "time.bin";
// Write the current time to the file
writeTimeToFile(fileName);
// Read and display the time from the file
readTimeFromFile(fileName);
return 0;
}
```

Output:

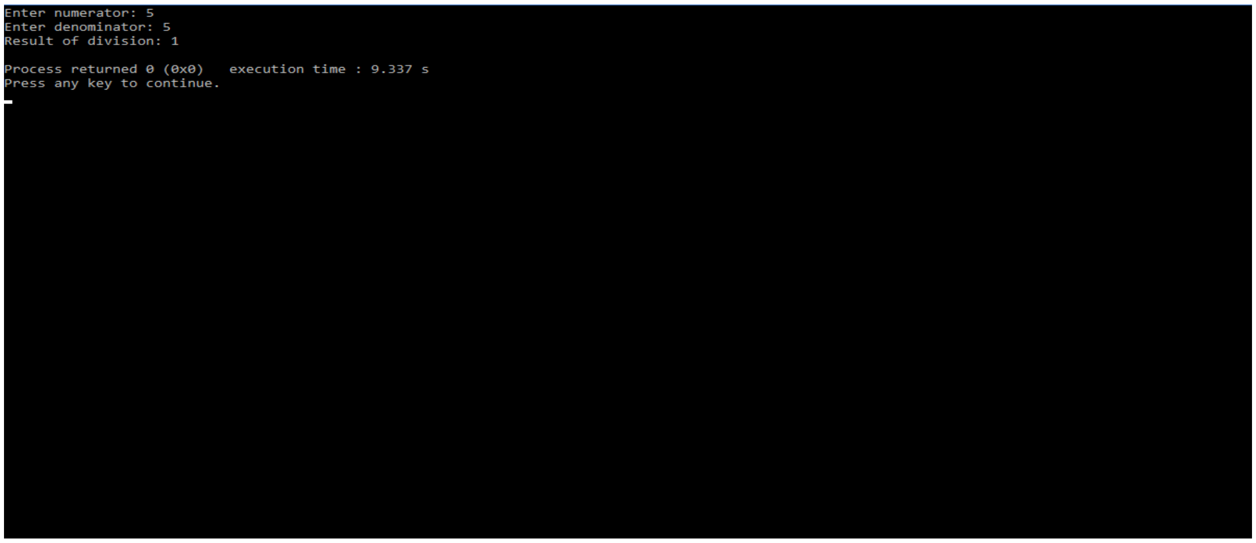


```
Time has been written to the file.
Time read from the file: Sun Dec 10 12:08:47 2023


Process returned 0 (0x0)   execution time : 2.062 s
Press any key to continue.
```

11. Develop a function which throws a division by zero exception and catch it in catch block. Write a C++ program to demonstrate usage of try, catch and throw to handle exception.

```
#include <iostream>
using namespace std;
double divide(int numerator, int denominator) {
if (denominator == 0) {
throw "Division by zero error";
}
return numerator / denominator;
}
int main() {
int numerator, denominator;
cout << "Enter numerator: ";
cin >> numerator;
cout << "Enter denominator: ";
cin >> denominator;
try {
double result = divide(numerator, denominator);
cout << "Result of division: " << result << endl;
} catch (const char* errorMessage) {
cerr << "Error:"  << errorMessage << endl;
}
return 0;
}
```

Output:

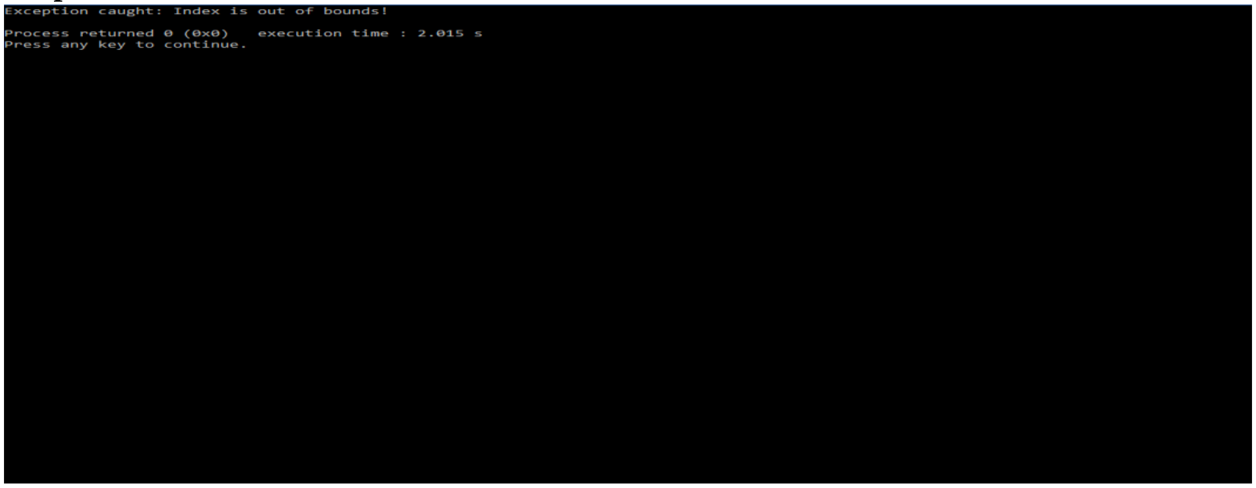


```
Enter numerator: 5
Enter denominator: 5
Result of division: 1

Process returned 0 (0x0)   execution time : 9.337 s
Press any key to continue.
```

12.Develop a C++ program that handles array out of bounds exception using C++.

```
#include <iostream>

#include <stdexcept>

using namespace std;

int main() {

    try {

        int arr[5] = {1, 2, 3, 4, 5};

        int index = 10; // Out-of-bounds index

        if (index >= 0 && index < 5) {

            cout << "Value at index " << index << ": " << arr[index] << endl;

        } else {

            throw out_of_range("Index is out of bounds!");

        }

    } catch (const out_of_range& e) {

        cout << "Exception caught: " << e.what() << endl;

    }

    return 0;

}
```

Output:



```
Exception caught: Index is out of bounds!

Process returned 0 (0x0)   execution time : 2.015 s
Press any key to continue.
```