#include <iostream>

#include <fstream>

using namespace std;

class student {

public:

struct stu {

char name[20];

int roll;

} s;

void put\_data();

void get\_data();

};

void student::put\_data() {

cout << "Enter name: ";

cin >> s.name;

cout << "Enter roll: ";

cin >> s.roll;

ofstream file;

file.open("hit.txt", ios::out | ios::app);

file.write((char\*)this, sizeof(student));

file.close();

get\_data();

}

void student::get\_data() {

int temp;

cout << "Enter roll no: ";

cin >> temp;

ifstream file;

file.open("hit.txt", ios::in);

file.seekg(0, ios::beg);

while (file.read((char\*)this, sizeof(student))) {

if (temp == s.roll) {

cout << "Student name: " << s.name << "\n";

cout << "Student roll: " << s.roll << "\n";

}

}

file.close();

}

int main() {

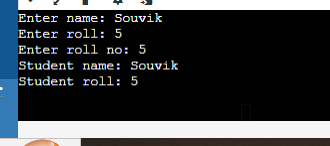
student st;

st.put\_data();

return 0;

}

Output:



#include <fstream>

#include <iostream>

#include <string>

using namespace std;

// Function to create a text file

void createTextFile(const string& filename) {

ofstream outfile(filename);

if (outfile.is\_open()) {

outfile << "This is a sample text file.\n";

outfile << "You can add more content here.\n";

cout << "Text file " << filename << " created successfully!" << endl;

} else {

cerr << "Error creating file: " << filename << endl;

}

outfile.close(); // Close the file even on errors

}

// Function to read from a text file

void readTextFile(const string& filename) {

ifstream infile(filename);

if (infile.is\_open()) {

string line;

while (getline(infile, line)) {

cout << line << endl;

}

} else {

cerr << "Error opening file: " << filename << endl;

}

infile.close(); // Close the file even on errors

}

// Function to write to a binary file

void writeBinaryFile(const string& filename, const char\* data, int size) {

ofstream outfile(filename, ios::binary);

if (outfile.is\_open()) {

outfile.write(data, size);

cout << "Binary data written to file " << filename << endl;

} else {

cerr << "Error creating binary file: " << filename << endl;

}

outfile.close(); // Close the file even on errors

}

// Function to read from a binary file

void readBinaryFile(const string& filename, int size) {

char buffer[size];

ifstream infile(filename, ios::binary);

if (infile.is\_open()) {

infile.read(buffer, size);

cout << "Binary data from file " << filename << ":" << endl;

for (int i = 0; i < size; ++i) {

cout << hex << static\_cast<int>(buffer[i]) << " ";

}

cout << endl;

} else {

cerr << "Error opening binary file: " << filename << endl;

}

infile.close(); // Close the file even on errors

}

int main() {

string textFilename = "example.txt";

string binaryFilename = "data.bin";

// Create a text file

createTextFile(textFilename);

// Read from the text file

readTextFile(textFilename);

// Sample data for binary file

char binaryData[] = "This is binary data";

// Write to a binary file

writeBinaryFile(binaryFilename, binaryData, sizeof(binaryData));

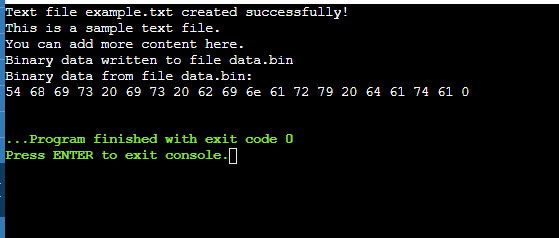
// Read from the binary file (adjust size based on written data)

readBinaryFile(binaryFilename, sizeof(binaryData));

return 0;

}

Output:



#include <iostream>

using namespace std;

float divison(int x , int y)

{

if(y==0)

{

throw "Attempted to divide by zero!";

}

return (x/y);

}

int main()

{

int i = 25;

int j = 0;

float k = 0;

try{

k = divison(i,j);

cout<< k <<endl;

}

catch(const char\* e)

{

cerr<< e << endl;

}

return 0;

}

Output:

