**Task 1:**

#include <iostream>

#include <string>

#include <algorithm>

using namespace std;

const int MAXROW = 100;

template<class T> T maxfunc(T[MAXROW]);

template<class T>

T maxfunc(T info[MAXROW]) {

T MAX;

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

if (i == 0) {

MAX = info[i];

}

if (info[i] > MAX) {

MAX = info[i];

}

}

return MAX;

}

int main() {

int infoNumber[MAXROW] = { 4, 1, 13, 3, 2 };

double infoNumber2[MAXROW] = { 1.1, 4.1, 8.1, 5.2, 2.3 };

string infoString[MAXROW] = {"the", "student", "is", "in", "class"};

int dataNumber = maxfunc<int>(infoNumber);

double dataNumber2 = maxfunc<double>(infoNumber2);

string dataString = maxfunc<string>(infoString);

cout << "Integer number output: " << dataNumber << endl;

cout << "Double number output: " << dataNumber2 << endl;

cout << "String output: " << dataString << endl;

return 0;

}

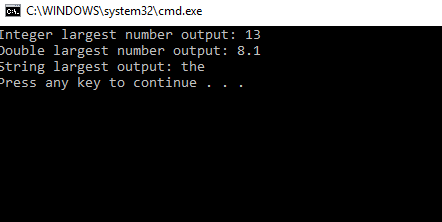
**Executable module instructions:**

1. **Compile**
2. **Run**

**Test data and expected results:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Largest Type** | **Input Data** | **Expected Output** |
| 1 | Integer | 4, 1, 13, 3, 2 | 13 |
| 2 | Double | 1.1, 4.1, 8.1, 5.2, 2.3 | 8.1 |
| 3 | string | "the", "student", "is", "in", "class" | “the” |

**Output:**



**Task 2:**

#include <iostream>

#include <string>

using namespace std;

struct Date {

int day;

int month;

int year;

};

template <class a>

class A {

a valuea;

public:

A() {

}

a getValuea() const {

return valuea;

}

void setValuea(a x) {

valuea = x;

}

A(A & a) {

valuea = a.getValuea();

}

};

template <class b>

class B :public A <b> {

b valueb;

public:

B() : A() {

}

b getValueb() const {

return valueb;

}

void setValueb(b x) {

valueb = x;

};

B(B & b) {

valueb = b.getValueb();

}

};

ostream & operator<<( ostream& out, const Date& rhs){

out << rhs.day << "/" << rhs.month << "/" << rhs.year;

return out;

}

int main() {

B <float> dataFloat1;

B <float> dataFloat2;

dataFloat1.setValuea(1.34);

dataFloat2.setValueb(3.14);

cout << "Value A: "<<dataFloat1.getValuea() << " Value B: " << dataFloat2.getValueb() << endl;

B <int> datainteger1;

B <int> datainteger2;

datainteger1.setValuea(1);

datainteger2.setValueb(3);

cout << "Value A: " << datainteger1.getValuea() << " Value B: " << datainteger2.getValueb() << endl;

B <char> datachar1;

B <char> datachar2;

datachar1.setValuea('a');

datachar2.setValueb('c');

cout << "Value A: " << datachar1.getValuea() << " Value B: " << datachar2.getValueb() << endl;

B <string> datastring1;

B <string> datastring2;

datastring1.setValuea("good");

datastring2.setValueb("morning");

cout << "Value A: " << datastring1.getValuea() << " Value B: " << datastring2.getValueb() << endl;

B<Date> a;

B<Date> b;

Date date1 = { 27,10,2014 };

Date date2 = {2,11,2014};

a.setValuea(date1);

b.setValueb(date2);

cout << "Value A: " << a.getValuea() << " Value B: " << b.getValueb() << endl;

return 0;

}

**Executable module instructions:**

1. **Compile**
2. **Run**

**Test data and expected results:**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Type** | **Input Data** | **Expected Output** |
| 1 | Float | Value A: 1.34  Value B: 3.14 | Value A: 1.34  Value B: 3.14 |
| 2 | Integer | Value A: 1  Value B: 3 | Value A: 1  Value B: 3 |
| 3 | char | Value A: a  Value B: c | Value A: a  Value B: c |
| 4 | String | Value A: good  Value B: morning | Value A: good  Value B: morning |
| 5 | Date | Value A: 27/10/2014  Value B: 2/11/2014 | Value A: 27/10/2014  Value B: 2/11/2014 |

**Output:**

