**Lab 7: Program 2 Documentation**

1. **Problem Statement**

We are reading inputs from the main source code, and using the methods set & inherited by the two template based classes in order to process them. The main method will serve as a medium to take in the action from the user and to perform the action (by calling set functions) on the list of numbers accordingly. It will also log the action and information accordingly into a log text file.

1. **Requirements**
   1. **Assumptions**

* Input set through user input in main source code
* Output will be handled by the command line
  1. **Specifications**
* Template Classes (will contain or inherit):
  + Value Number of set template data type
  + Public function to return value
  + Public function to set value
  + Public function to make a copy
* Structure:
  + Integer variables for day, month, and year
* Source Code:
  + Print out all values by calling the template-class functions
    - Depending on how it is written in the main, it can read an “instance” as a different data type

1. **Decomposition Diagram**

* Source Code
  + Input
    - Program takes inputs from the main source code
  + Process
    - Functions listed under specifications can be called by user via source code
    - Program has checks in place to process request and follows them accordingly
  + Output
    - Program will print out the results onto the command line

|  |  |
| --- | --- |
| Source Code | |
| Responsibilities:  Template-Class A:  +T getValuea() const  +T setValuea(T x)  Template-Class B:  +B::B()  +T getValueb() const  +T setValueb(T x)  +B(const A &instance) | Collaborators: N/A |

|  |
| --- |
| Source Code |
| Responsibilities:  Template-Class A - T getValuea() const, void setValuea(T x)  Template-Class B - B::B(), T getValueb() const, void setValueb(T x), B(const A &instance) |
| Template Class A:   * T getValuea() const   + Return valuea when this method is called * void setValuea(T x)   + Set valuea to the value of the input parameter (given as x)   Template Class B:   * B::B()   + Constructor method used to create object * T getValueb() const   + Return valueb when this method is called * void setValueb(T x)   + Set valueb to the value of the input parameter (given as x) * B(const A &instance)   + Uses getValuea() function to get the value of a, and to copy it into valueb |

1. **Test Strategy**

* Valid Data
* Invalid Data

1. **Test Plan Version 1**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Strategy | Test Number | Description | Input | Expected Output | Actual Output | Pass/Fail |
| Valid Data | 1 | Object of type float, setValuea & print |  |  |  |  |
| Valid Data | 2 | Object of type float, setValueb & print |  |  |  |  |
| Valid Data | 3 | Object of type int, setValuea & print |  |  |  |  |
| Valid Data | 4 | Object of type int, setValueb & print |  |  |  |  |
| Valid Data | 5 | Object of type int, setValuea & print |  |  |  |  |
| Valid Data | 6 | Object of type int, setValueb & print |  |  |  |  |
| Valid Data | 7 | Object of type int, setValuea & print |  |  |  |  |
| Valid Data | 8 | Object of type int, setValueb & print |  |  |  |  |
| Valid Data | 9 | Object with struct input, setValuea for day, month, and year |  |  |  |  |
| Valid Data | 10 | Object with struct input, setValuea for day, month, and year |  |  |  |  |

1. **Initial Algorithm**

Template-Class A:

* T getValuea() const
  + Return valuea when this method is called
* void setValuea(T x)
  + Set valuea to the value of the input parameter (given as x)

Template-Class B:

* B::B()
  + Constructor method used to create object
* T getValueb() const
  + Return valueb when this method is called
* void setValueb(T x)
  + Set valueb to the value of the input parameter (given as x)
* B(const A &instance)
  + Uses getValuea() function to get the value of a, and to copy it into valueb

Struct Data

* Set int data types variables: day, month, year

main()

* instance1 (input float)
  + setValuea
  + getValuea
  + setValueb
  + getValueb
* instance2 (input int)
  + setValuea
  + getValuea
  + setValueb
  + getValueb
* instance3 (input char)
  + setValuea
  + getValuea
  + setValueb
  + getValueb
* instance4 (input string)
  + setValuea
  + getValuea
  + setValueb
  + getValueb
* instance5 (input class)
  + setValuea
  + getValuea.day
  + getValuea.month
  + getValuea.year
  + setValueb
  + getValueb.day
  + getValueb.month
  + getValueb.year

1. **Test Plan Version 2**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Strategy | Test Number | Description | Input | Expected Output | Actual Output | Pass/Fail |
| Valid Data | 1 | Object of type float, setValuea & print | instance1.setValuea(1.34);  cout << instance1.getValuea() << endl; | 1.34 |  |  |
| Valid Data | 2 | Object of type float, setValueb & print | instance1.setValueb(3.14);  cout << instance1.getValueb() << endl; | 3.14 |  |  |
| Valid Data | 3 | Object of type int, setValuea & print | instance2.setValuea(1);  cout << instance2.getValuea() << endl; | 1 |  |  |
| Valid Data | 4 | Object of type int, setValueb & print | instance2.setValueb(3);  cout << instance2.getValueb() << endl; | 3 |  |  |
| Valid Data | 5 | Object of type int, setValuea & print | instance3.setValuea('a');  cout << instance3.getValuea() << endl; | a |  |  |
| Valid Data | 6 | Object of type int, setValueb & print | instance3.setValueb('c');  cout << instance3.getValueb() << endl; | c |  |  |
| Valid Data | 7 | Object of type int, setValuea & print | instance4.setValuea("good");  cout << instance4.getValuea() << endl; | good |  |  |
| Valid Data | 8 | Object of type int, setValueb & print | instance4.setValueb("bad");  cout << instance4.getValueb() << endl; | bad |  |  |
| Valid Data | 9 | Object with struct input, setValuea for day, month, and year | instance5.setValuea({ 27, 10, 2015 });  cout << instance5.getValuea().day << endl;  cout << instance5.getValuea().month << endl;  cout << instance5.getValuea().year << endl; | 27  10  2015 |  |  |
| Valid Data | 10 | Object with struct input, setValuea for day, month, and year | instance5.setValueb({ 2, 11, 2015 });  cout << instance5.getValueb().day << endl;  cout << instance5.getValueb().month << endl;  cout << instance5.getValueb().year << endl; | 2  11  2015 |  |  |
| Invalid Data | 11 | NA: Due to all test cases being given | NA: Due to all test cases being given | NA |  |  |

1. **Code**
   1. **Source Code**

//Program Name: Lab 7 - Program 2

//Programmer Name: Srinivas Simhan

//Description: We are using classes set via templates and learning how to implement them

//Due Date: 11/8/2017

#include <iostream>

#include <string>

using namespace std;

template <class T>

class A

{

T valuea;

public:

T getValuea() const

{

return valuea;

}

void setValuea(T x)

{

valuea = x;

}

};

template <class T>

class B : public A <T>

{

T valueb;

public:

B::B()

{

}

T getValueb() const

{

return valueb;

}

void setValueb(T x)

{

valueb = x;

}

B(const A &instance)

{

valueb = getValuea();

}

};

struct Date

{

int day;

int month;

int year;

};

int main()

{

//Float

B<float> instance1;

instance1.setValuea(1.34);

cout << instance1.getValuea() << endl;

instance1.setValueb(3.14);

cout << instance1.getValueb() << endl;

//Integer

B<int> instance2;

instance2.setValuea(1);

cout << instance2.getValuea() << endl;

instance2.setValueb(3);

cout << instance2.getValueb() << endl;

//Char

B<char> instance3;

instance3.setValuea('a');

cout << instance3.getValuea() << endl;

instance3.setValueb('c');

cout << instance3.getValueb() << endl;

//String

B<string> instance4;

instance4.setValuea("good");

cout << instance4.getValuea() << endl;

instance4.setValueb("bad");

cout << instance4.getValueb() << endl;

//Struct

B<Date> instance5;

instance5.setValuea({ 27, 10, 2015 });

cout << instance5.getValuea().day << endl;

cout << instance5.getValuea().month << endl;

cout << instance5.getValuea().year << endl;

instance5.setValueb({ 2, 11, 2015 });

cout << instance5.getValueb().day << endl;

cout << instance5.getValueb().month << endl;

cout << instance5.getValueb().year << endl;

// system("pause");

}

1. **Updated Algorithm**

Template-Class A:

* Set variable valuea using template
  + Within class A
    - T valuea;
* T getValuea() const
  + Return valuea when this method is called
    - return valuea;
* void setValuea(T x)
  + Set valuea to the value of the input parameter (given as x)
    - valueb = x;

Template-Class B:

* Set variable valueb using template
  + Within class B : public A (class B inherits public A methods)
    - T valueb;
* B::B()
  + Constructor method used to create object
* T getValueb() const
  + Return valueb when this method is called
    - return valueb;
* void setValueb(T x)
  + Set valueb to the value of the input parameter (given as x)
    - valueb = x;
* B(const A &instance)
  + Uses getValuea() function to get the value of a, and to copy it into valueb
    - valueb = getValuea()

Struct Data

* Set int data types variables: day, month, year

main()

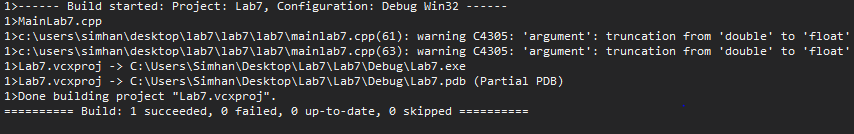
* instance1 (input float)
  + setValuea
    - instance set at 1.34
  + getValuea
    - printed via cout
  + setValueb
    - instance set at 3.14
  + getValueb
    - printed via cout
* instance2 (input int)
  + setValuea
    - instance set at 1
  + getValuea
    - printed via cout
  + setValueb
    - instance set at 3
  + getValueb
    - printed via cout
* instance3 (input char)
  + setValuea
    - instance set at ‘a’
  + getValuea
    - printed via cout
  + setValueb
    - instance set at ‘c’
  + getValueb
    - printed via cout
* instance4 (input string)
  + setValuea
    - instance set at “good”
  + getValuea
    - printed via cout
  + setValueb
    - instance set at “bad”
  + getValueb
    - printed via cout
* instance5 (input class)
  + setValuea
    - instance set at {27, 10, 2015}
  + getValuea.day
    - printed via cout
  + getValuea.month
    - printed via cout
  + getValuea.year
    - printed via cout
  + setValueb
    - instance set at {2, 11, 2015}
  + getValueb.day
    - printed via cout
  + getValueb.month
    - printed via cout
  + getValueb.year
    - printed via cout

1. **Test Plan Version 3**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Test Strategy | Test Number | Description | Input | Expected Output | Actual Output | Pass/Fail |
| Valid Data | 1 | Object of type float, setValuea & print | instance1.setValuea(1.34);  cout << instance1.getValuea() << endl; | 1.34 | \*see screenshot below\* | Pass |
| Valid Data | 2 | Object of type float, setValueb & print | instance1.setValueb(3.14);  cout << instance1.getValueb() << endl; | 3.14 | \*see screenshot below\* | Pass |
| Valid Data | 3 | Object of type int, setValuea & print | instance2.setValuea(1);  cout << instance2.getValuea() << endl; | 1 | \*see screenshot below\* | Pass |
| Valid Data | 4 | Object of type int, setValueb & print | instance2.setValueb(3);  cout << instance2.getValueb() << endl; | 3 | \*see screenshot below\* | Pass |
| Valid Data | 5 | Object of type int, setValuea & print | instance3.setValuea('a');  cout << instance3.getValuea() << endl; | a | \*see screenshot below\* | Pass |
| Valid Data | 6 | Object of type int, setValueb & print | instance3.setValueb('c');  cout << instance3.getValueb() << endl; | c | \*see screenshot below\* | Pass |
| Valid Data | 7 | Object of type int, setValuea & print | instance4.setValuea("good");  cout << instance4.getValuea() << endl; | good | \*see screenshot below\* | Pass |
| Valid Data | 8 | Object of type int, setValueb & print | instance4.setValueb("bad");  cout << instance4.getValueb() << endl; | bad | \*see screenshot below\* | Pass |
| Valid Data | 9 | Object with struct input, setValuea for day, month, and year | instance5.setValuea({ 27, 10, 2015 });  cout << instance5.getValuea().day << endl;  cout << instance5.getValuea().month << endl;  cout << \*see screenshot below\*instance5.getValuea().year << endl; | 27  10  2015 | \*see screenshot below\* | Pass |
| Valid Data | 10 | Object with struct input, setValuea for day, month, and year | instance5.setValueb({ 2, 11, 2015 });  cout << instance5.getValueb().day << endl;  cout << instance5.getValueb().month << endl;  cout << instance5.getValueb().year << endl; | 2  11  2015 | \*see screenshot below\* | Pass |
| Invalid Data | 11 | NA: Due to all test cases being given | NA: Due to all test cases being given | NA | NA | Pass |

1. **Screenshots**

Sceenshots for test cases (1-11) & build.





1. **Error Log**
   1. No errors found
2. **Status**
   1. The final status of my program was that it worked 100%. If any wrong data type inputs were given, it would not compile due to the fact that the compiler recognized that the template data type used was not the same data type that you were implementing, therefore you had to change the code to the data type that would work.