If this lab is an Individual assignment, you must do all coded programs on your own. You may ask others for help on the language syntax, but you must organize and present your own logical solution to the problem. No lab is complete until the MyClass submits the signed pledge form associated with that lab. I realize that no coded programs will be graded until I turn in the sign & pledge form associated with that program; any late penalties will continue to compound until the pledge form is submitted.

If this lab is a team assignment, both team members may share logic as they program side by side on their own computers. Each person must type all of his/her own code as part of the learning process. Team assignments are never to be “You do this portion and I’ll do that portion” or “You do this lab and I’ll do the next lab”.

Some of the lab assignments will have short answer questions. These short answer questions will be spot checked and graded for completion, but not checked for accuracy. Once these labs are graded and returned, I encourage you to compare answers with another class member who has also had the lab graded and returned.

I realize that no coded programs will be graded until I turn in the sign & pledge form associated with that program; any late penalties will continue to compound until the pledge form is submitted.

I/We realize that the penalty for turning in work that is not my own, or assisting others in doing so, can range from an "F" in the class to dismissal from Trinity University. I realize that it is a violation of academic integrity to share any portion of this lab with any person (outside my 2320 team & professor)!

Print Name _________________________________________ Time Required = ______.____ Hrs.
Signature _______________________________________________________________ (pledged)

OOP- 5 Stacks Print Only Pages 1-5
Individual Assignment
Answers To Questions 10 Points - Program 15 Points

Short Answer Questions – 5 Points
Electronic Solutions Of Short Answer Questions Will Not Be Accepted. Print A Copy Of Short Answer Question & Write Answers On Printed Copy

Our Stack Uses Dynamic Memory (Review This From OOP-4 If Necessary)

1] ____________________________________________________________________________ D_?_ pointers are ones that reference memory incorrectly.

2] ____________________________________________________________________________ In order to avoid dangling pointers, the constructor should either allocate the needed dynamic memory or set the pointer to _?_.

3] ____________________________________________________________________________ In order to avoid dangling pointers, the constructor should either _?_ or set the pointer to NULL.


5] ____________________________________________________________________________ _?_ {Shallow/Deep} makes an actual copy of the data to which the pointers are pointing.

Stacks

1] ____________________________________________________________________________ LIFO is an acronym for _?_.

2] ____________________________________________________________________________ FIFO is an acronym for _?_.

3] ____________________________________________________________________________ The _?_ is an ADT for FIFO.

4] ____________________________________________________________________________ The _?_ is an ADT for LIFO.
List the three primitive operations required of all stacks.

The primitive operation, and method, which adds new information the top of a Stack is called _?_.

The primitive operation, and method, which uses a reference variable to return a copy of the top info & remove that top item is called _?_.

```cpp
#define MAX 5
class Stack
{
    public:
    long int Info[MAX]
}
```

The design, illustrated with the block of code above, is considered poor. Why?

Why should the software engineer use C++ templates when designing the a great Stack class?

Why should the software engineer use a dynamic Info array, instead of a static Info array, when designing the templated Stack class?

Stack _?_ (Overflow/Underflow) is an attempt to push more information into the stack than it will hold; exceed the capacity.

Stack _?_ (Overflow/Underflow) is an attempt to pop more information from the stack than it contains.

What Is Stack Underflow?

What Is Stack Overflow?
# define MAX 5
class Stack
{
    public:
    long int Info[MAX];
}

15) {T/F} A stack defined with the code above must overload the = operator to use a deep copy. (try it if not 100% sure)

class Stack
{
    public:
    long int * Info;
}

16) {T/F} A stack defined with the code below must overload the = operator to use a deep copy .(try it if not 100% sure)

class EmpName
{
    public:
    char
        *First,
        * Last;
};

17) Write the complete C++ code for an Employee constructor which will accept the following:

EmpName
    Nancy("Doe", "Jane");
class EmpName {
  public:
  char
    *First,
    *Last;
};

18] Write the complete C++ code for an employee copy constructor which will accept the following:
   EmpName
     Nancy("Doe", "Jane");
   EmpName
     Nancy2(Nancy);

19] Write the code of C++ code to create an integer stack, called Data, that contains 22 integers with subscripts 0 - 21.

20] Write the code to place 55 into the stack, called Data.

21] Write the line of code to extract the top element from stack Data; the value in that element shall be placed into an integer variable called TempInt.

22] Write the block of code display "Stack Empty" or "Stack Not Empty" as is appropriate for stack Data.

23] Write the code to create an Athlete Type stack, called Trinity, that contains 15 Athletes with subscripts 0 - 14.
Athlete
TempAthlete,
Sarah ("Sarah", 111, 1, 11.11);

24] Write the line of code to place Sarah into the stack, called Trinity.

25] Write the line of code to extract the top element from stack Trinity; a copy of that element shall be placed into integer TempAthlete as it is removed from the stack.

Coding Practice
You Should Be Able To Code Any Of The Following For A Quiz Or Exam
Find A Whiteboard & Practice With Your Study Group!

1] Write the complete code for a good dynamic template stack constructor. You may assume the presence of any other method.
   Stack (long int NewMax = 10);

2] Write the complete code a good dynamic template stack destructor. You may assume the presence of any other method.
   ~Stack (void);

3] Write the complete code a good dynamic template stack Empty method. You may assume the presence of any other method.
   bool Empty (void);

4] Write the complete code a good dynamic template stack Full method. You may assume the presence of any other method.
   bool Full (void);

5] Write the complete code a good dynamic template stack Push method. You may assume the presence of any other method.
   bool Push (InfoType NewInfo);

6] Write the complete code a good dynamic template stack Pop method. You may assume the presence of any other method.
   bool Pop (InfoType & OldInfo);

7] Write the complete code a good dynamic template stack StackTop method. You may assume the presence of any other method.
   bool StackTop (InfoType & OldInfo);

Coding Practice
I Almost Always Ask One Of These Three Problems On A Quiz Or Exam

1] Write the complete code a good dynamic template stack Resize method. You may assume the presence of any other method.
   bool Resize(long int SizeChange = 5);
2] Write the complete code a good dynamic template stack operator = overload. You may assume the presence of any other method.

```cpp
void operator = (const Stack & S);
```

3] Write the complete code a good dynamic template stack copy constructor method. You may assume the presence of any other method.

```cpp
Stack (const Stack & S);
```

---

Programming Component 15 Points

1] Download a copy of Stack-Student.zip. Extract the project → Call your folder TomH-Stack (Use your first name and last initial).

2] Review the slides. Complete all of the Stack methods; uncomment out all of the test code.

3] Examine the output carefully; compare it with my output below.

4] Always make sure that all functions are documented properly. Name, Date, etc. My name should only appear as the author on the Stack.Display method.

5] Set the STACK_DIAGNOSTIC_LEVEL = 2

6] Run the program one last time.

---

```text
Ouput
Memory Addresses Will Probably Not Match Those Below!
```

```
===================================================================
============= Program Stack ============
============= Written By Dr. Tom Hicks ============
===================================================================

*******************************************************************
*******************************************************************
*******************************************************************
*******************************************************************
************************* Start TestStack *************************
*******************************************************************
*******************************************************************
*******************************************************************

Contents of IntStack1
```
Stack Is Empty
Max = 10  Top = -1  &=8335592  &=7F30E8

Contents of IntStack2
2  -842150451
1  -842150451
0  -842150451
Stack Is Empty
Max = 2  Top = -1  &=8344640  &=7F5440

Contents of IntStack3
10  -842150451
9   -842150451
8   -842150451
7   -842150451
6   -842150451
5   -842150451
4   -842150451
3   -842150451
2   -842150451
1   -842150451
### Contents of FloatStack4

<table>
<thead>
<tr>
<th>Index</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-842150451</td>
</tr>
</tbody>
</table>

Stack Is Empty

Max = 10  Top = -1  &=8344696  &=7F5478

### Contents of CharStack5

<table>
<thead>
<tr>
<th>Index</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>i</td>
</tr>
<tr>
<td>4</td>
<td>i</td>
</tr>
<tr>
<td>3</td>
<td>i</td>
</tr>
<tr>
<td>2</td>
<td>i</td>
</tr>
<tr>
<td>1</td>
<td>i</td>
</tr>
<tr>
<td>0</td>
<td>i</td>
</tr>
</tbody>
</table>

Stack Is Empty

Max = 5  Top = -1  &=8344848  &=7F5510

### Contents of InventoryStack6

<table>
<thead>
<tr>
<th>Index</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>0 0 0 0 0.00</td>
</tr>
<tr>
<td>5</td>
<td>0 0 0 0 0.00</td>
</tr>
<tr>
<td>4</td>
<td>0 0 0 0 0.00</td>
</tr>
<tr>
<td>3</td>
<td>0 0 0 0 0.00</td>
</tr>
<tr>
<td>2</td>
<td>0 0 0 0 0.00</td>
</tr>
<tr>
<td>1</td>
<td>0 0 0 0 0.00</td>
</tr>
<tr>
<td>0</td>
<td>0 0 0 0 0.00</td>
</tr>
</tbody>
</table>

Stack Is Empty
Max = 6 Top = -1 &=8344908 &=7F554C

Contents of ClassStack8

<table>
<thead>
<tr>
<th>8</th>
<th>0 Male</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>0 Male</td>
</tr>
<tr>
<td>6</td>
<td>0 Male</td>
</tr>
<tr>
<td>5</td>
<td>0 Male</td>
</tr>
<tr>
<td>4</td>
<td>0 Male</td>
</tr>
<tr>
<td>3</td>
<td>0 Male</td>
</tr>
<tr>
<td>2</td>
<td>0 Male</td>
</tr>
<tr>
<td>1</td>
<td>0 Male</td>
</tr>
<tr>
<td>0</td>
<td>0 Male</td>
</tr>
</tbody>
</table>

Stack Is Empty

Max = 8 Top = -1 &=8345292 &=7F56CC

Contents of GarageStack9

<table>
<thead>
<tr>
<th>9</th>
<th>0 Convertable</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>0 Convertable</td>
</tr>
<tr>
<td>7</td>
<td>0 Convertable</td>
</tr>
<tr>
<td>6</td>
<td>0 Convertable</td>
</tr>
<tr>
<td>5</td>
<td>0 Convertable</td>
</tr>
<tr>
<td>4</td>
<td>0 Convertable</td>
</tr>
<tr>
<td>3</td>
<td>0 Convertable</td>
</tr>
<tr>
<td>2</td>
<td>0 Convertable</td>
</tr>
<tr>
<td>1</td>
<td>0 Convertable</td>
</tr>
<tr>
<td>0</td>
<td>0 Convertable</td>
</tr>
</tbody>
</table>

Stack Is Empty

Max = 9 Top = -1 &=8345596 &=7F57FC

<Hit Any Key To Continue>
Valid ==> Stack1 is Empty!

Unable to test not empty because Push is not yet done.

*<Hit Any Key To Continue>*

---

<table>
<thead>
<tr>
<th>IntStack1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
</tr>
<tr>
<td>Max = 5 Top = 0 &amp;=8335592 &amp;7F30E8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IntStack1</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>Max = 5 Top = 1 &amp;=8335592 &amp;7F30E8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IntStack1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>Max = 5 Top = 2 &amp;=8335592 &amp;7F30E8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IntStack1</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>0</td>
</tr>
<tr>
<td>Max = 5 Top = 3 &amp;=8335592 &amp;7F30E8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IntStack1</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>2</td>
</tr>
</tbody>
</table>
Max = 5  Top = 4  &=8335592  &=7F30E8

IntStack1

Max = 5  Top = 5  &=8335592  &=7F30E8

Stack Overflow - Attempt to more in Stack than it can hold! - With resize, we should not see this until all of RAM is exhausted

<Hit Any Key To Continue>
IntStack1

4 | 50 |
3 | 40 |
2 | 30 |
1 | 20 |
0 | 10 |

Max = 4 Top = 4 &=8335592 &=7F30E8

OldInfo = 50

IntStack1

3 | 40 |
2 | 30 |
1 | 20 |
0 | 10 |

Max = 4 Top = 3 &=8335592 &=7F30E8

OldInfo = 40

IntStack1

2 | 30 |
1 | 20 |
0 | 10 |

Max = 4 Top = 2 &=8335592 &=7F30E8

OldInfo = 30

IntStack1

1 | 20 |
0 | 10 |

Max = 4 Top = 1 &=8335592 &=7F30E8

OldInfo = 20

IntStack1

0 | 10 |

Max = 4 Top = 0 &=8335592 &=7F30E8

OldInfo = 10

IntStack1
Stack Is Empty

Max = 4 Top = -1 &=8335592 &=7F30E8

Attempted Stack Underflow!

Attempted Stack Underflow!

8 Students Pushed In Alphabetical Order!

<table>
<thead>
<tr>
<th></th>
<th>Last Name</th>
<th>First Name</th>
<th>Year</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>Guest</td>
<td>Ben</td>
<td>109</td>
<td>Male</td>
</tr>
<tr>
<td>6</td>
<td>Merka</td>
<td>Lauren</td>
<td>108</td>
<td>Female</td>
</tr>
<tr>
<td>5</td>
<td>Silliman</td>
<td>Mark</td>
<td>107</td>
<td>Male</td>
</tr>
<tr>
<td>4</td>
<td>Wilson</td>
<td>Jennifer</td>
<td>106</td>
<td>Female</td>
</tr>
<tr>
<td>3</td>
<td>Kelly</td>
<td>Nick</td>
<td>105</td>
<td>Male</td>
</tr>
<tr>
<td>2</td>
<td>Elsaifi</td>
<td>Leslie</td>
<td>104</td>
<td>Female</td>
</tr>
<tr>
<td>1</td>
<td>McBryde</td>
<td>Michael</td>
<td>103</td>
<td>Male</td>
</tr>
<tr>
<td>0</td>
<td>Weston</td>
<td>Clay</td>
<td>102</td>
<td>Male</td>
</tr>
</tbody>
</table>

Max = 10 Top = 7 &=8346636 &=7F5C0C

Guest, Ben 109 Male
Merka, Lauren 108 Female
Silliman, Mark 107 Male
Wilson, Jennifer 106 Female
Kelly, Nick 105 Male
Elsaifi, Leslie 104 Female
McBryde, Michael 103 Male
Weston, Clay 102 Male

Stack Underflow - Attempt To Get More Out Of Stack Than Available!

Stack Underflow - Attempt To Get More Out Of Stack Than Available! <Hit Any Key To Continue>

===================================================================
Test StackTop
STACK_DIAGNOSTIC_LEVEL = 8
===================================================================

4 Athletes Pushed

<table>
<thead>
<tr>
<th></th>
<th>Last Name</th>
<th>Year</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Thomas</td>
<td>1004</td>
<td>12 44.44</td>
</tr>
<tr>
<td>2</td>
<td>Rebecca</td>
<td>1002</td>
<td>2 33.33</td>
</tr>
<tr>
<td>1</td>
<td>Michael</td>
<td>1002</td>
<td>11 22.22</td>
</tr>
<tr>
<td>0</td>
<td>Sarah</td>
<td>1001</td>
<td>1 11.11</td>
</tr>
</tbody>
</table>
Max = 4  Top = 3  &=8346636  &=7F5C0C

---

**Peek At Stack Top**
Name ...........: Thomas  
No ............: 1004  
Sport No ......: 12  
Equipment Value : 44.44

---

**Peek At Stack Top**
Name ...........: Rebecca  
No ............: 1002  
Sport No ......: 2  
Equipment Value : 33.33

---

**Peek At Stack Top**
Name ...........: Michael  
No ............: 1002  
Sport No ......: 11  
Equipment Value : 22.22

---

**Peek At Stack Top**
Name ...........: Sarah  
No ............: 1001  
Sport No ......: 1  
Equipment Value : 11.11

Nothing in Stack to peek at!  
Nothing in Stack to peek at!  

<Hit Any Key To Continue>

---

**Test Resize**

**STACK_DIAGNOSTIC_LEVEL = 9**

---

30 Athletes Pushed To A Stack Of 5!

<table>
<thead>
<tr>
<th>No</th>
<th>Name</th>
<th>Sport No</th>
<th>Equipment Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>David</td>
<td>1006</td>
<td>66.66</td>
</tr>
<tr>
<td>4</td>
<td>Nancy</td>
<td>1005</td>
<td>44.44</td>
</tr>
<tr>
<td>3</td>
<td>Thomas</td>
<td>1004</td>
<td>44.44</td>
</tr>
<tr>
<td>2</td>
<td>Rebecca</td>
<td>1002</td>
<td>33.33</td>
</tr>
<tr>
<td>1</td>
<td>Michael</td>
<td>1002</td>
<td>22.22</td>
</tr>
<tr>
<td>0</td>
<td>Sarah</td>
<td>1001</td>
<td>11.11</td>
</tr>
</tbody>
</table>

Max = 5  Top = 5  &=8346636  &=7F5C0C
CSCI2320 is full with 4

<table>
<thead>
<tr>
<th></th>
<th>Name</th>
<th>ID</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Thomas</td>
<td>1004</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>Rebecca</td>
<td>1002</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>Michael</td>
<td>1002</td>
<td>11</td>
</tr>
<tr>
<td>0</td>
<td>Sarah</td>
<td>1001</td>
<td>1</td>
</tr>
</tbody>
</table>

Max = 3  Top = 3  &=8346636  &=7F5C0C

CSCI3343 is Empty With 10

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Stack Is Empty

Max = 10  Top = -1  &=8346876  &=7F5CF0C

====> CSCI2320 is full with 4

<table>
<thead>
<tr>
<th></th>
<th>Name</th>
<th>ID</th>
<th>Grade</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>Thomas</td>
<td>1004</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>Rebecca</td>
<td>1002</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>Michael</td>
<td>1002</td>
<td>11</td>
</tr>
<tr>
<td>0</td>
<td>Sarah</td>
<td>1001</td>
<td>1</td>
</tr>
</tbody>
</table>

Max = 3  Top = 3  &=8346636  &=7F5C0C
CSCI3343 is CSCI3343

3  Thomas  1004 12  44.44
2  Rebecca  1002  2  33.33
1  Michael  1002 11  22.22
0  Sarah    1001  1  11.11

Max = 3  Top = 3  &=8347452  &=7F5F3C

<Hit Any Key To Continue>

===================================================================
===================================================================
===============
STACK_DIAGNOSTIC_LEVEL = 11
===============
===================================================================
===================================================================
CSCI2320

3  Thomas  1004 12  44.44
2  Rebecca  1002  2  33.33
1  Michael  1002 11  22.22
0  Sarah    1001  1  11.11

Max = 3  Top = 3  &=8346636  &=7F5C0C

CSCI3343

3  Thomas  1004 12  44.44
2  Rebecca  1002  2  33.33
1  Michael  1002 11  22.22
0  Sarah    1001  1  11.11

Max = 3  Top = 3  &=8346876  &=7F5CFC

<Hit Any Key To Continue>

*******************************************************************
*******************************************************************
***************************
End TestStack
***************************
*******************************************************************
*******************************************************************
*******************************************************************
*******************************************************************
What To Turn In

- - - - - - - - - - - - - No Lab Is Complete Until Both Are Complete - - - - - - - - - - - - -

1] You sign & submit the Pledge form.
   a) Make sure that all program files have a header box with a purpose that clearly defines what you are accomplishing in this lab.
   b) Make sure that each and every program function has a well formed documentation box that clearly describes the purpose.
   c) Make sure that each and every program function header box has the appropriate Written By and Date.
   d) Review the Pledge statement
   e) Sign & Pledge
   f) Record the amount of time you think you spent on this lab
   g) Staple all pages of this lab. Fold in half length-wise (like a hot-dog). Put your name on the outside. Place it on the professor desk before the beginning of lecture on the day it is due. The penalty for late homework will not exceed 25% off per day.

2] Place all programming code associated with this program, if any, in the Professor’s Code Drop Box
   a) I do not accept programs by mail; do not submit labs via email!

- - - - - - - - - - - - - - - - - Comments - - - - - - - - - - - - - - - - -

A] Programs that do not compile are worth little, if anything.
B] If a print statement format is off, the penalties will often be less than the 25% per day late penalty; turn in the lab. You would not be happy if you went to Best Buy and purchased a large screen TV that did everything except show the picture; you would consider it pretty worthless. Most users consider software that does not work properly pretty useless as well. If the lab is not working correctly, credit will be small (if any); you might be better to accept a 25% (1 day) late penalty and turn in the lab working correctly!
C] Start all programs early so that you can get in contact with the professor if you have problems.
D] If you are turning in this lab late, you may
   – hand it to me if I am in the office
   – put it in the mail box outside my office door
   – slide it under the outer door to our suite (if locked)
   – slide it under my office door. The sooner I get late labs, the sooner the late penalty meter quits clicking.
E] Backup your programs in at least three places. Put a copy on your Y drive. Put a copy on your flash drive. Put a copy on your personal computer. Send yourself a copy in your e-mail.