Midterm Exam

Due Oct 14 at 11:59pm **Points** 100 **Questions** 33 **Available** Oct 14 at 7am - Oct 14 at 11:59pm about 17 hours
Time Limit 120 Minutes

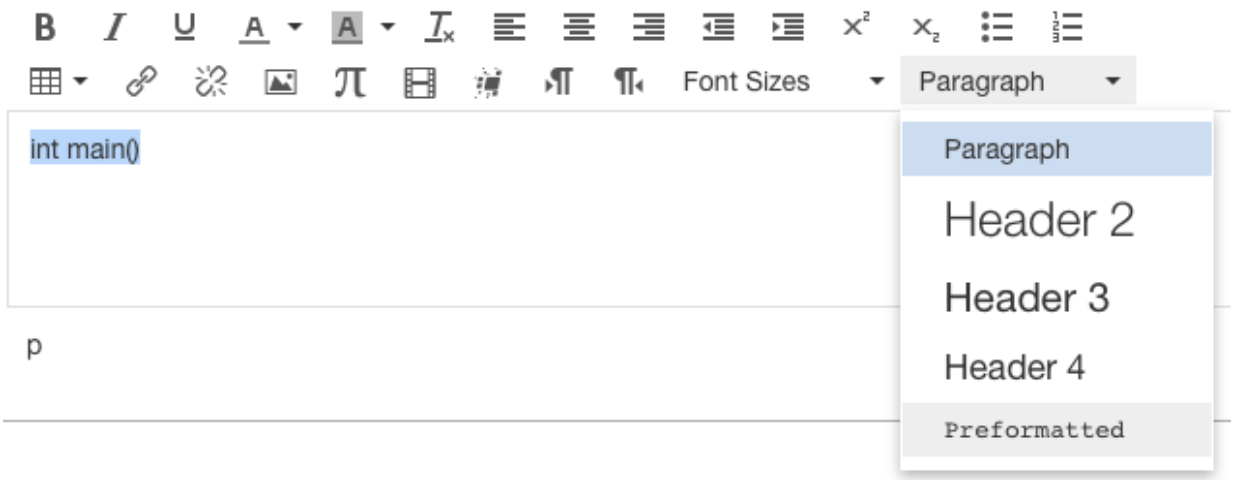
Instructions

This is a midterm test for understanding of the material covered in our labs, lectures, and readings through Module 6, Stacks And Queues. It is worth 100 "exam points".

There are true/false, multiple choice, short essay, code blocks, and fill-in-the-blank questions. The exam is *open book, open note, and open internet*.

You have 120 minutes from the time you start the exam! If you pause or leave to come back later, the timer will still run. It will NOT pause.

When you write code, write in "preformatted" instead of the default "paragraph". Switch like this:



Once you start the exam, you'll have 120 minutes of clock time to complete in. If you need to pause your exam, simply navigate to the Home page and resume later. But note that the time period allowed for the exam remains the same.

Once you choose to Submit the exam, you're done. The final results will be revealed once the exam is graded by the instructor.

This quiz was locked Oct 14 at 11:59pm.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	120 minutes	100 out of 100 *

* Some questions not yet graded

Score for this quiz: **100** out of 100 *
Submitted Oct 14 at 10:56pm
This attempt took 120 minutes.

Question 1 — 2 / 2 pts

Given a struct definition like this:

```
struct Student
{
  string name;
  int studentID;
  float gpa;
};
```

...identify the "***host object***" in the following code block in the main program:

```
  Student x;
  output(x); // where "output" is a function with a prototype and definition
```

- output
- Student

Correct!

- trick question -- there is no host object
- output's return value
- x

Question 2

2 / 2 pts

The C++ keyword "**this**" is:

Correct!

- a pointer to the host object
- the value of the first parameter in a member function
- a reference variable, referring to the host object in a member function

Question 3

2 / 2 pts

Write a C++ statement to return a reference to the host object from a member setter function.

Correct!

return *this;

Correct Answers

return *this;

return (*this);

return *this

return (*this)

Question 4

2 / 2 pts

"Data hiding" is when:

- data members in a class are used, but are not declared anywhere.

Correct!

- (•) a main program does not have direct access to an object's data members and has to use getters and setters instead.

- a main program has access to public data members of an object, but is not supposed to use them directly.

- a main program has no way at all to get or set hidden data in an object.

Question 5

2 / 2 pts

There are only two differences between a C struct and a C++ class. Explain both of them.

Your Answer:

1.spelling
2. Default of the members in struct is public, but default of the members in class is private

Question 6

2 / 2 pts

The differences between the C struct and the C++ class are very slight, and we really don't need both because we have the **public** and **private** keywords. But we still use both in programming. When should we use a C struct instead of a C++ class?

Your Answer:

If all values are public, we can use struct without type "public:".
It is just for convenience

Question 7

2 / 2 pts

What is the C++ keyword that designates a member function as a "getter"?

Correct!

const

Correct Answers

const

Question 8

2 / 2 pts

For the following struct:

```
struct Test
{
  void doSomething() const;
  void doSomething();
};
```

...which function gets called from this code block in the main program?

```
  Test test;
  test.doSomething();
```

- `void doSomething() const;`

Correct!

- (•) `void doSomething();`

- Trick question -- there cannot be two functions with the same name and parameter list

Question 9

2 / 2 pts

A member function does not mutate the host object.

- Write it either as a "getter" or as a "setter" -- either one is fine.
- Correct! Write it as a "getter".
- Write both "getter" and "setter" versions so that both const and non-const objects can call it.
- Write it as a "setter".

Question 10

2 / 2 pts

The compiler prevents the main program from calling "setter" functions on const objects, because const objects cannot be mutated. But there is a well-know loophole that allows this -- simply write a "getter" that calls a "setter".

- True
- Correct! False

Question 11

2 / 2 pts

A member function has a parameter named "size". But its class also was a data member called "size". Inside the function, I've lost access to the data member. I could just rename the parameter, but I don't want to do that. What should I write instead of the following to access the data member?

```
size
```

Correct! this->size

Correct Answers (*this).size

this->size

Question 12

2 / 2 pts

To write an inline member function in a templated class, you have to use the C++ **inline** keyword.

- True
- False

Correct!

Question 13

2 / 2 pts

Explain the concept of "range safe" in your own words.

Your Answer:

In c++, even if programmer did not set the value, program can access an invalid array. In this case, program show trash value or be crashed. So "range safe" make range limit to prevent program access to invalid indexes.

Question 14

2 / 2 pts

For an array of capacity 100, the lowest usable index is 0 and the highest is 99 .

Answer 1:

0

Correct!

Answer 2:

99

Correct!

Question 15

2 / 2 pts

For the following struct:

```
<template typename T>
struct Test
{
  T operator[](int) const;
  T& operator[](int);
};
```

...which function gets called from this statement in the main program?

```
  cout << a[i];
```

- the setter
- it depends on the declaration of "a"

Correct!

- the getter

Question 16

2 / 2 pts

What, if anything, is wrong with this setter, assuming that "values" is a valid data member array of ints of capacity 10?

```
int& Array::operator[ ](int index)
{
  if (index < 0 || index >= 10)
    return -999;
  return values[index];
}
```

Your Answer:

When we write a statement like a[11] = 3, we are setting a[11] to 3.
However, a[11] returns -999. -999 can not be set to 3. It does not make sense.
So The operator setter function should not return int value. but a variable like "dummy"

Question 17

2 / 2 pts

What's the purpose of "dummy" and *why* does it have to be a data member instead of a local variable inside a member function?

Your Answer:

Dummy is a value for range safety. If an invalid indexes are inputed, the function returns dummy variable not to crash the program. If dummy is a local variable, dummy would disappear after processing the function. If program call the dummy that already disappeared, the program would be crashed. So programmer should set dummy as a data member.

Question 18

6 / 6 pts

Name the five tests that should be included in a test driver for a template with dynamic memory.

Your Answer:

1. Test all public functions
2. Const object test
3. Object copy test
4. Object assignment test
5. The "ifndef" test

Question 19

6 / 6 pts

Match the three tests with their code.

Correct! **ifndef test** — #include "Array.h"

Correct! **const object copy test** — const Array b = a;

Correct! **object assignment test** — Array b; b = a;

Other Incorrect Match Options:

- const Array b; b = a;
- #include <Array.h>
- Array b = a;

Question 20

2 / 2 pts

What coding error would be revealed by a failed ifndef test?

Your Answer:

The purpose of ifndef test is to confirm that the #ifndef, #define, and #endif statements are written correctly. If ifndef test were failed, program would make compiler error when cpp file has the same header file included more than once

Question 21

2 / 2 pts

What coding error would be revealed by the "*const*" part of the const object test?

Your Answer:

Const object test used "const" to confirm the public member function that programmer intended to be getters are in fact coded as getters. If the program tries to change the value during const object test, the coding error would appear.

Question 22

2 / 2 pts

What does the T stand for in the C++ STL?

Correct!

template

Correct Answers

template

TEMPLATE

Template

Question 23

2 / 2 pts

A header file contains a class template, and in that class there is a C++ string object.

There should be a #include for the string library AND a using namespace std; in the main program's CPP file, written before the H file's include.

There should be a #include for the string library.

Correct!

There should be a #include for the string library AND a using namespace std; in the header file.

Question 24

6 / 6 pts

Name the three dynamic memory management functions.

Your Answer:

1.assignment operator

2.copy constructor

3.destructor

Question 25

2 / 2 pts

When should the main program call a object's destructor function?

Before the object goes out of scope.

Before the program ends.

Correct!

Trick question -- destructors are called automatically.

Question 26

2 / 2 pts

What is the purpose of this statement in a capacity-adjusting member setter function?

```
int limit = min(cap, this->cap);
```

Your Answer:

"int limit" detect which number is smaller between int cap and current capacity. Program copies "int limit" number from original array and set rest value as default value. If capacity is increased, limit = cap. If capacity is decreased, limit = this->cap.

Question 27

6 / 6 pts

Explain in your own words "data abstraction".

Your Answer:

Data abstraction is that allowing object to get various data type(int, char, string, etc) Template is a kind of data abstraction. Data abstraction make program more flexible.

Question 28

4 / 4 pts

Why is it not a good idea to initialize values in a template class' data storage array to zero?

Your Answer:

Because we don't know what the data type of the value is. Default value of int and double are zero, but default value of string is not. So we can not initialize values as zero

Question 29

10 / 10 pts

Write a C++ ***declaration statement*** for a table of C++ strings with an unspecified number of columns, and an unspecified number of rows in each column. Do *NOT* track the number of columns, and do *NOT* track the number of rows in each column. Use either or both of your StaticArray and/or DynamicArray templates from the lab assignments, but no STL containers. Name the table as you wish.

Your Answer:

DynamicArray<DynamicArray<string>> table;

Question 30

10 / 10 pts

Write a new setter function template for your DynamicArray templated class per the following prototype:

```
void swap(int, int);
```

...that swaps the values stored at the indexes represented by the two parameters. *You* decide how to deal with indexes that are beyond the capacity of the DynamicArray. Write the *whole function template*.

Your Answer:

template <class V>

```
void DynamicArray<V>::swap(int i, int j)
{
  V temp = this->operator[](j);
  this->operator[](j) = this->operator[](i);
  this->operator[](i) = temp;
}
```

Question 31 **6 / 6 pts**

OPTIONAL. Any suggestions for making the course better? Anything about Canvas or the lectures or grading or quizzes -- anything.

Your Answer:

Quiz Score: **100** out of 100