Midterm 3 Study Guide

Due No due date **Points** 25 **Questions** 25 **Time Limit** 30 Minutes **Allowed Attempts** Unlimited

Take the Quiz Again

Attempt History

	Attempt	Time	Score
KEPT	Attempt 30	23 minutes	25 out of 25
LATEST	Attempt 33	23 minutes	24 out of 25
	Attempt 32	24 minutes	23 out of 25
	Attempt 31	29 minutes	24 out of 25
	Attempt 30	23 minutes	25 out of 25
	Attempt 29	23 minutes	24 out of 25
	Attempt 28	22 minutes	25 out of 25
	Attempt 27	20 minutes	20 out of 25
	Attempt 26	16 minutes	25 out of 25
	Attempt 25	18 minutes	24 out of 25
	Attempt 24	15 minutes	25 out of 25
	Attempt 23	21 minutes	24 out of 25
	Attempt 22	24 minutes	24 out of 25
	Attempt 21	18 minutes	24 out of 25
	Attempt 20	30 minutes	19 out of 25
	Attempt 19	20 minutes	23 out of 25
	Attempt 18	30 minutes	21 out of 25
	Attempt 17	30 minutes	23.5 out of 25
	Attempt 16	30 minutes	24 out of 25
	Attempt 15	30 minutes	21 out of 25
	Attempt 14	30 minutes	23 out of 25
	Attempt 13	24 minutes	22 out of 25
	Attempt 12	16 minutes	24 out of 25
	Attempt 11	17 minutes	19 out of 25
	Attempt 10	17 minutes	22 out of 25
	Attempt 9	20 minutes	20 out of 25
	Attempt 8	21 minutes	20 out of 25
	Attempt 7	25 minutes	21.5 out of 25
	Attempt 6	25 minutes	21 out of 25
	Attempt 5	30 minutes	17 out of 25
	Attempt 4	21 minutes	23 out of 25
	Attempt 3	26 minutes	19.89 out of 25
	Attempt 2	30 minutes	22.5 out of 25
	Attempt 1	27 minutes	21 out of 25

Correct answers are hidden.

Submitted Jul 20 at 6:21pm

Question 1 1 / 1 pts

A loop that reads data until some special value is found is called a sentinel loop.

- [✓ True]
- False

Question 2 1 / 1 pts

Assume that you have the following code:

```
istreamstring in("one");
int n;
```

Which of these (erroneous) statements cause the program to terminate?

- [ ] `in >> n;`
- [✓] `assert(2 + 2 == 5);`
- [ ] `cout << sqrt(-1);`
- [✓] `cout << stoi("one");`

Question 3 1 / 1 pts

What happens with the following section of code?

```
if (__APPLE__)
   cout << "Running on a Mac" << endl;
else if (__WIN32)
   cout << "Running on Windows" << endl;
else if (__linux)
   cout << "Running on Linux" << endl;
else
   cout << "Running on an unknown platform" << endl;
```

- ○ The program will crash if compiled on one platform, but run on another.
- ○ Only the lines that identify your platform will be included in the executable
- ○ All lines will be included in the program. It will print the platform you are running on.
- [✓ The program will not compile]

Question 4

1 / 1 pts

Variables tested with the `#if` preprocessor directive are created using `#define`.

- [✓ True]
- ○ False

Question 5

1 / 1 pts

Programmers use _______________ to reason about logical correctness in their code.

- ○ completion codes
- [✓ assertions]
- ○ exceptions
- ○ assumptions

Question 6

1 / 1 pts

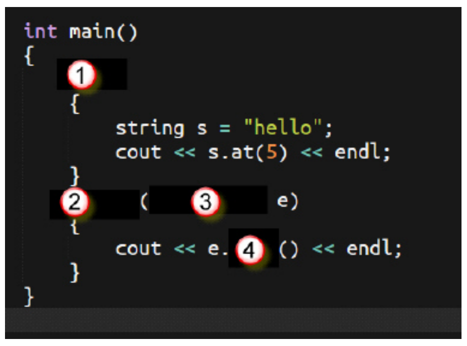
A loop that reads data until some special value is found is called a data loop.

- ○ True
- [✓ False]

Question 7

1 / 1 pts

What is correct for # 1?



```
int main()
{
    1
    {
        string s = "hello";
        cout << s.at(5) << endl;
    }
   2      (   3      e)
    {
        cout << e. 4 () << endl;
    }
}
```

- [✓ `try`]
- ○ `while`
- ○ `catch`
- ○ `exception&`
- ○ None of these
- ○ `if`
- ○ `what`

Question 8

1 / 1 pts

In the ***loop-and-a-half***, you use a `break` statement to exit the loop when the sentinel is found.

[✓ True]

False

Question 9

1 / 1 pts

The order of the `catch` blocks does not affect the program.

True

[✓ False]

Question 10

1 / 1 pts

The declaration: `vector<int> v(10);` creates a `vector` object containing uninitialized elements.

True

[✓ False]

Question 11

1 / 1 pts

The following definition:

```
vector<double> v(3, 5);
```

creates a `vector` of `[3.0, 5.0]`

creates a `vector` of `[3.0, 3.0, 3.0, 3.0, 3.0]`

[✓ creates a `vector` of `[5.0, 5.0, 5.0]`]

None of these

is a syntax or compiler error

Question 12

1 / 1 pts

The declaration: `vector<int> v(10, 5);` creates a `vector` object containing five integers.

True

[✓ False]

Question 13

1 / 1 pts

Assuming that `Star` is a structure, the declaration: `vector<Star> stars(3);` creates three uninitialized `Star` objects.

True

[✓ False]

Question 14

1 / 1 pts

The declaration: `vector<string> v(5);` creates a `vector` containing five empty `string` objects.

[✓ True]

False

Incorrect

Question 15

0 / 1 pts

Which statement is false? The elements in a vector:

- [✓ None of these]
- are accessed by name
- are stored next to each other in memory
- are homogeneous
- are all of the same type

Question 16

1 / 1 pts

To count the number of elements in a `vector` that match a particular value, use the STL function:

- `count_if`
- `find`
- `search`
- [✓ `count`]
- `minmax_element`

Question 17

1 / 1 pts

Examine the following code (which is legal). What changes are necessary to allow the statement `if (m1 != m2)` … to compile?

```
struct Money { int dollars{0}, cents{0}; } m1, m2;

bool equals(const Money& lhs, const Money& rhs)
{
    return lhs.cents == rhs.cents &&
        lhs.dollars == rhs.dollars;
}
```

- This is not possible in C++.
- The name of `equals()` must be changed to `operator==`
- [✓ You must write a function named `operator!=`]
- The function `equals()` must be named `notEquals()`.
- The type `Money` needs to be a `class`

Question 18

1 / 1 pts

What is a common pointer error?

- Dereferencing a pointer
- Assigning a new value to a pointer
- Using indirection on a pointer
- Setting a pointer value to `nullptr`
- [✓ Using a pointer without first initializing it]

Question 19

1 / 1 pts

The elements of a C++ ***`string`*** array with no explicit initialization, created in a function will be set to ***`null`***.

- True
- [✓ False]

Question 20

1 / 1 pts

These pointers should point to "nothing". Which is not correctly initialized?

- `vector<int> *vp(0);`
- [✓ All are correctly initialized to point to nothing.]
- `int *pi = nullptr;`

- ○ `Star *ps = NULL;`
- ○ `double *pd{};`

Question 21

1 / 1 pts

The subscripts of a C++ array range from ***0*** to the allocated array size –***1***.

- [✓ True]
- ○ False

Question 22

1 / 1 pts

A forward reference can be used when you want to use a pointer to a structure as a data member without first defining the entire structure.

- [✓ True]
- ○ False

Question 23

1 / 1 pts

What prints?

```
int a[] = {1, 3, 5, 7, 9};
int *p = a;
cout << *p++;
cout << *p << endl;
```

- ○ 22
- ○ 12
- ○ 33
- ○ None of these
- [✓ 13]

Question 24

1 / 1 pts

Examine the following code. What is stored in ***a*** after it runs.

```
int f(int * p, int x)
{
    *p = x * 2;
    return x / 2;
}
. . .
int a = 3, b, c;
c = f(&b, a);
```

- ○ 2
- ○ 1
- ○ 6
- ○ Does not compile
- [✓ 3]

Question 25

1 / 1 pts

If ***img*** is a pointer to the first byte in an image loaded into memory, ***Pixel*** is a structure, you can create a ***Pixel*** pointer pointing to the image by writing:
`Pixel *p = static_cast<Pixel *>(img);`

- ○ True
- [✓ False]