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 16	30 minutes	24 out of 25
LATEST	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 18 at 10:23pm

Question 1

1 / 1 pts

A `catch(…)` will catch any kind of thrown exception.

- (•) True
- ( ) False

Question 2

1 / 1 pts

Match each item with the correct loop form below.

Indefinite limit loop that reduces its input	while (n != 0) { n /= 2; }
Indefinite limit loop that uses successive approximations	while(abs(g1 – g2) >= EPSILOI
Counter-controlled symmetric loop for producing a sequence of data	for (int i=12; i <= 19; i++) {. . .}
Indefinite data loop that uses raw input	while(cin.get(ch)) {. . .}
Counter-controlled asymmetric loop for processing characters	for (size_t i=0, len=s.size(); i < l
Iterator loop that may change its container	for (auto& e : col) {. . .}
Iterator loop that cannot change its container	for (auto e : col) {. . .}
Counter-controlled loop for processing substrings	for (size_t i=4, slen=4, len=s.siz
Indefinite data loop that uses formatted input	while(cin >> n) {. . .}

Question 3

1 / 1 pts

You can report a syntax error encountered in your code by using the `throw` keyword.

- ( ) True
- (•) False

Question 4

1 / 1 pts

The standard library version of `stoi("UB-40")` throws a runtime exception because there is no viable conversion.

- (●) True
- ( ) False

Question 5

1 / 1 pts

What term describes this block of code?

```
#if __APPLE__
istringstream in("  .75");
int n = 3;
in >> n;
#endif
```

- ( ) proprietary compilation
- ( ) selection statements
- ( ) compiler directives
- (●) conditional compilation
- ( ) alternative compilation
- ( ) None of these

Question 6

1 / 1 pts

In the ***primed loop pattern***, you use a `break` statement to exit the loop when the sentinel is found.

- ( ) True
- (●) False

Question 7

1 / 1 pts

Suppose you have written a non-interactive program that inputs data from a file. If the input file does not exist when the program executes, then you should choose which option?

- ( ) Ignore the error and continue
- ( ) Log the error and continue
- ( ) Throw an exception
- (●) Terminate the program with an error message

Incorrect

Question 8

0 / 1 pts

What happens when this code fragment compiles and runs?

```
#define N
#ifndef N
    cout << "Hello";
#else
    cout << "Goodbye";
#endif
```

- ( ) `prints "HelloGoodbye"`
- ( ) prints nothing
- ( ) It does not compile.
- ( ) `prints "Goodbye"`
- (●) `prints "Hello"`

Question 9

1 / 1 pts

A specialized error handling block of code, is called a `catch` block.

- (●) True
- ( ) False

Question 10

1 / 1 pts

`vector` subscripts begin at `0` and go up to the `vector size - 1`.

- (●) True
- ( ) False

Question 11

1 / 1 pts

What is stored in `data` after this runs?

```
vector<int> data{1, 2, 3};
data.pop_back();
```

- ( ) `[]`
- (●) `[1, 2]`
- ( ) `[1, 2, 3]`
- ( ) `[2, 3]`
- ( ) `[1, 2, 3, 0]`
- ( ) None of these

Incorrect

Question 12

0 / 1 pts

What prints?

```
vector<int> v{1, 2, 3, 4, 5};
v.pop_back();
cout << v.back() << endl;
```

- ( ) Nothing; run-time error.
- ( ) `1`
- ( ) `4`
- ( ) `5`
- (●) Nothing; compile-time error.

Question 13

1 / 1 pts

The following definition:

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

- ( ) None of these
- ( ) creates a `vector` of `[3.0, 5.0]`
- (●) creates a `vector` of `[5.0, 5.0, 5.0]`
- ( ) creates a `vector` of `[3.0, 3.0, 3.0, 3.0, 3.0]`
- ( ) is a syntax or compiler error

Question 14

1 / 1 pts

Assume `vector<double> speed(5);` Which line throws a ***runtime error***?

- (●) `cout << speed.at(speed.size());`
- ( ) `speed.front() = 12;`
- ( ) None of these
- ( ) `speed[0] = speed.back()`
- ( ) `speed.erase(speed.begin());`

Question 15

1 / 1 pts

User-defined types that combine multiple values into a single type are called ***scalar*** types.

- ○ True
- ◉ False

Incorrect

Question 16

0 / 1 pts

What prints when this code runs?

```
enum class Coin
{
   PENNY, NICKEL, DIME, QUARTER
};
cout << Coin::PENNY << endl;
```

- ◉ `0`
- ○ `PENNEY`
- ○ `Coin::PENNY`
- ○ `1`
- ○ Does not compile; Cannot output enumerated members without overloaded operator.

Incorrect

Question 17

0 / 1 pts

Assume `vector<int> v`; Writing `cout << v.front();` is undefined.

- ○ True
- ◉ False

Question 18

1 / 1 pts

Which array definition is illegal (even if it may compile on some compilers)?

```
int SIZE = 3;
int a1[SIZE];
int a2[3];
int a3[3]{};
int a4[] = {1, 2, 3};
int a5[3] = {1, 2};
```

- ○ `a5`
- ○ `a2`
- ◉ `a1`
- ○ None of these
- ○ `a3`

Question 19

1 / 1 pts

C++ arrays have no support for bound-checking.

- ◉ True
- ○ False

Question 20

1 / 1 pts

What is the term used to describe a variable which stores a memory address?

- ○ `lvalue`
- ◉ `pointer`
- ○ None of these
- ○ `reference`

- rvalue

Question 21

1 / 1 pts

What is printed?

```
int a[] = {1, 2, 3};
int b[] = {1, 2, 3};

if (a == b) cout << "a == b" << endl;
else cout << "a != b" << endl;
```

- Undefined behavior
- a != b
- a == b
- Syntax error; does not compile.

Question 22

1 / 1 pts

Assume that ***ppi*** correctly points to ***pi***. Which line prints the value stored inside ***pi***?

```
int main()
{
    double pi = 3.14159;
    double *ppi;
    // code goes here
    // code goes here
}
```

- cout << &ppi;
- cout << ppi;
- None of these
- cout << *pi;
- cout << π

Question 23

1 / 1 pts

What is true about this code?

```
int n{500};
int *p = &n;
```

- p stores the same value as n
- &p represents the indirect value of n
- *p is the value of n
- &p is the direct or explicit value of n
- &n is the indirect value of p

Question 24

1 / 1 pts

You can calculate the number of elements in an array, provided the array definition is in scope.

- True
- False

Question 25

1 / 1 pts

You must use an integral constant or literal to specify the size of a built-in C++ array.

- True
- False