# Midterm 2 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 20 | 26 minutes | 25 out of 25 |
| **LATEST** | Attempt 20 | 26 minutes | 25 out of 25 |
| | Attempt 19 | 22 minutes | 20 out of 25 |
| | Attempt 18 | 30 minutes | 23 out of 25 |
| | Attempt 17 | 19 minutes | 24 out of 25 |
| | Attempt 16 | 21 minutes | 24 out of 25 |
| | Attempt 15 | 30 minutes | 23 out of 25 |
| | Attempt 14 | 22 minutes | 24 out of 25 |
| | Attempt 13 | 25 minutes | 24 out of 25 |
| | Attempt 12 | 30 minutes | 22 out of 25 |
| | Attempt 11 | 29 minutes | 25 out of 25 |
| | Attempt 10 | 30 minutes | 20.17 out of 25 |
| | Attempt 9 | 29 minutes | 20 out of 25 |
| | Attempt 8 | 29 minutes | 20.5 out of 25 |
| | Attempt 7 | 28 minutes | 21 out of 25 |
| | Attempt 6 | 24 minutes | 21 out of 25 |
| | Attempt 5 | 20 minutes | 17.83 out of 25 |
| | Attempt 4 | 30 minutes | 17.67 out of 25 |
| | Attempt 3 | 22 minutes | 16 out of 25 |
| | Attempt 2 | 16 minutes | 16.17 out of 25 |
| | Attempt 1 | 30 minutes | 18.33 out of 25 |

Correct answers are hidden.

Submitted Jun 29 at 4:51pm

### Question 1

1 / 1 pts

In a ***guarded*** loop, the loop actions are always executed at least once.

- True
- False

### Question 2

1 / 1 pts

How many times is this loop ***entered***? (*That is, how many times is i printed?*)

```
for (int i = 1; i < 10; i++)
    cout << i;
cout << endl;
```

- Never
- 11
- 10
- 9

## Question 3

1 / 1 pts

Below is the illustration from the loop building strategy. The ***highlighted lines*** represent. Add one to (or increment) the counter variable:

```
Given: the variable str is a string (may be empty)
Create the counter variable, initialized to -1
If the variable str has any characters then
{
    Set counter to 0
    Create the variable current-character as a character
    Place the first character in str into current-character
    While more-characters and current-character not a period
    {
       Add one to (or increment) the counter variable
       Store the next character from str in current-character
    }
    If current-character is a period then
       Add one to the counter to account for the period.
    Else
      Set counter to -2
}
If counter is -1 the string was empty
Else if counter is -2 there was no period
```

- advancing the loop
- goal precondition
- loop bounds
- bounds precondition
- goal operation
- loop postcondition

## Question 4

1 / 1 pts

*Match each item with the correct question below.*

**What must I change in the test to go to the next iteration?** advance the loop

**What must I do to enter the loop?** bounds precondition

**Has my loop reached its goal?** loop postcondition

**Can my loop be entered at all?** loop guards

## Question 5

1 / 1 pts

A ***guarded*** loop is also known as a ***test-at-the-top*** loop.

- (•) True
- ( ) False

## Question 6

1 / 1 pts

Below is the illustration from the loop building strategy. The ***highlighted lines*** represent. While more-characters and current-character not a period:

```
Given: the variable str is a string (may be empty)
Create the counter variable, initialized to -1
If the variable str has any characters then
{
    Set counter to 0
    Create the variable current-character as a character
    Place the first character in str into current-character
    While more-characters and current-character not a period
    {
       Add one to (or increment) the counter variable
       Store the next character from str in current-character
    }
    If current-character is a period then
       Add one to the counter to account for the period.
    Else
      Set counter to -2
}
If counter is -1 the string was empty
Else if counter is -2 there was no period
```

- ( ) goal operation
- ( ) loop postcondition
- ( ) bounds precondition
- ( ) advancing the loop
- ( ) goal precondition
- (•) loop bounds

## Question 7

1 / 1 pts

The highlighted section below illustrates:

```
Given: the variable str is a string (may be empty)
Create the counter variable, initialized to -1
If the variable str has any characters then
{
    Set counter to 0
    Create the variable current-character as a character
    Place the first character in str into current-character
    While more-characters and current-character not a period
    {
       Add one to (or increment) the counter variable
       Store the next character from str in current-character
    }
    If current-character is a period then
       Add one to the counter to account for the period.
    Else
      Set counter to -2
}
If counter is -1 the string was empty
Else if counter is -2 there was no period
```

- a necessary condition
- a postcondition
- a boundary condition
- None of these
- an intentional condition
- a loop guard

## Question 8

1 / 1 pts

How many times is this loop ***entered***? (*That is, how many times is i printed?*)

```
for (int i = 0; i < 10; i++)
    cout << i;
cout << endl;
```

- 11
- 10
- 9
- Never

## Question 9

1 / 1 pts

Which line represents the ***intentional bounds*** in this loop?

```
1.      string s("Hello CS 150");
2.      while (s.size())
3.      {
4.         if (s.at(0) == 'C') break;
5.         s = s.substr(1);
6.      }
7.      cout << s << endl;
```

- (•) 4
- ( ) None of these
- ( ) 5
- ( ) 2

## Question 10

1 / 1 pts

Which of these may go into a header file?

- [x] function prototypes
- [x] constant definitions
- [ ] function definitions
- [ ] global variable definitions

## Question 11

1 / 1 pts

To allow *f()* to change the argument passed here, the parameter *str* should be declared as:

```
void f( . . . str);
int main()
{
    string s = "hello";
    f(s);
}
```

- ( ) const string&
- ( ) string
- (•) string&
- ( ) const string
- ( ) It is not possible for *f()* to change the argument passed here.

## Question 12

1 / 1 pts

In a library, the ***interface*** file:

- consists of function calls
- consists of instructions that produce the executable
- None of these
- (selected) consists of declarations or prototypes
- consists of function definitions

## Question 13

1 / 1 pts

What kind of error is this?

```
ex1.cpp:6:5: error: use of undeclared identifier 'a'
    a = 4;
    ^
```

- Linker error (something is missing when linking)
- Operating system signal or trap
- (selected) Compiler error (something is missing when compiling)
- Syntax error (mistake in grammar)
- Runtime error (throws exception when running)
- Type error (wrong initialization or assignment)
- None of these

## Question 14

1 / 1 pts

An ***undeclared*** error message is a compiler error.

- (selected) True
- False

## Question 15

1 / 1 pts

What kind of error is this?

**Segmentation fault**

- Runtime error (throws exception when running)
- None of these
- Compiler error (something is missing when compiling)
- Syntax error (mistake in grammar)
- Linker error (something is missing when linking)
- Type error (wrong initialization or assignment)
- (selected) Operating system signal or trap

## Question 16

1 / 1 pts

What kind of error is this?

```
ex1.cpp:6:12: error: no viable conversion from 'int' to 'string'
    string a = 15;
           ^   ~~
```

- Operating system signal or trap
- Runtime error (throws exception when running)
- None of these
- Linker error (something is missing when linking)
- Syntax error (mistake in grammar)
- (selected) Type error (wrong initialization or assignment)
- Compiler error (something is missing when compiling)

## Question 17

1 / 1 pts

A while loop is a hasty or unguarded loop.

- True
- (selected) False

## Question 18

1 / 1 pts

Which of these ***are not*** state filters?

- [x] copy a file
- [ ] print one sentence per line
- [ ] counting word transitions
- [ ] compress input by turning off echo when reading blank spaces
- [x] search for a particular value in a stream
- [x] translating data from one form to another

## Question 19

1 / 1 pts

What is the value of ***r(8818)***?

```
int r(int n)
{
    if (!n) return 0;
    return (n % 10 == 8) + (n % 100 == 88) + r(n / 10);
}
```

- ( ) Does not compile
- ( ) 3
- ( ) 1
- ( ) Stack overflow
- (•) 4

## Question 20

1 / 1 pts

Two quantities *a* and *b* are said to be in the *golden ratio* if $\frac{(a+b)}{a}$ is equal to $\frac{a}{b}$. Assuming *a* and *b* are line segments, the *golden section* is a line segment divided according to the golden ratio: The total length *(a + b)* is to the longer segment *a* as *a* is to the shorter segment *b*. One way to calculate the golden ratio is through the continued square root (also called an *infinite surd*): golden ratio = $\sqrt{1+\sqrt{1+\sqrt{1+\sqrt{1+\ldots}}}}$. In a recursive implementation of this function, what should be the *base case* for the recursion?

- ( ) `if (number <= 1) { return sqrt(number);}`
- (•) `if (number <= 1) { return 1.0;}`
- ( ) `if (number <= 1) { return 0.0;}`
- ( ) `if (number <= 1) { return pow(number, 2.0);}`

## Question 21

1 / 1 pts

When using `cat` with redirection, the program only stops running when you press Control+D.

- ○ True
- ◉ False

## Question 22

1 / 1 pts

How can you ensure that a recursive function terminates?

- ○ Provide a special case for the most complex inputs.
- ○ Call the recursive function with more complex inputs.
- ◉ Provide a special case for the simplest inputs.
- ○ Use more than one return statement.

## Question 23

1 / 1 pts

Which command displays a the names of the files in a folder in reverse order?

- ☐ `None of these`
- ☑ `ls -r`
- ☐ `ls | wc -l`
- ☐ `ls | sort`
- ☑ `ls | sort -r`
- ☐ `ls -r | sort`

## Question 24

1 / 1 pts

The `cout` object is an instance of the `ostream` class.

- ◉ True
- ○ False

## Question 25

1 / 1 pts

`cat < a.txt > b.txt` makes a copy of `a.txt` in the file `b.txt`

- (●) True
- ( ) False