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 11	29 minutes	25 out of 25
LATEST	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 28 at 1:46pm

Question 1

1 / 1 pts

An ***unguarded*** loop is also known as a ***test-at-the-bottom*** loop.

- (•) True
- ( ) False

Question 2

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
```

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

- (●) loop bounds
- ( ) goal operation

Question 3

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 4

1 / 1 pts

Below is the illustration from the loop building strategy. The ***highlighted lines*** represent.
Create the variable current-character as a character:

```
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
```

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

Question 5

1 / 1 pts

Match each item with the correct statement below.

Keeps processing input until a particular value is found in input.	sentinel loop
Keeps processing until the output gets no closer to the answer.	limit loop

Repeats its actions a fixed number of times	definite loop
Keeps processing until the input device signals that it is finished.	data loop

Question 6

1 / 1 pts

Loops are used to implement selection in C++.

- ○ True
- ◉ False

Question 7

1 / 1 pts

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

- ○ True
- ◉ False

Question 8

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;
```

- ○ 10
- ○ Never
- ○ 11
- ◉ 9

Question 9

1 / 1 pts

In the classic *for* loop, loop control variables going from 0 to less-than n are said to employ:

- ○ None of these
- ◉ asymmetric bounds
- ○ symmetric bound
- ○ necessary bounds
- ○ intentional bounds

Question 10

1 / 1 pts

What prints here?

```
auto a = 3, b = 3;
cout << (a != b ? "panda": "tiger") << endl;
```

- ○ Crashes when run
- ○ Undefined behavior
- ◉ tiger
- ○ panda
- ○ Does not compile

Question 11

1 / 1 pts

A while loop is a hasty or unguarded loop.

- ○ True
- ◉ False

Question 12

1 / 1 pts

Given the ***overloaded*** functions prototypes and the variable definition below, which of the function calls will fail to compile?

```
int f(int&);
int f(int);
int f(int, int);
int a = 7;
```

- ☑ f(a);
- ☐ f(2.0);
- ☐ f(3)
- ☐ f('a', 'b')
- ☐ None of these fail to compile

Question 13

1 / 1 pts

What kind of error is this?

Segmentation fault

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

Question 14

1 / 1 pts

Arguments passed to a function that has a constant reference parameter must be:

- constants
- either lvalues or rvalues are fine
- reference arguments
- rvalues
- lvalues

Incorrect

Question 15

0 / 1 pts

Assume that the input is `4 4 3 2 5`. What will print?

```
int i = 1;
do
{
    int n;
    cin >> n;
    i++;
}
while (n % 2);
cout << i << endl;
```

- 4
- infinite loop
- Does not compile
- 3
- 2

Question 16

1 / 1 pts

Default arguments allow you to write several different functions that have the same name.

- True
- False

Question 17

1 / 1 pts

Which of these are ***dependencies***?

```
EXE=digit-tester
OBJS=client.o digits.o
$(EXE): $(OBJS)
    $(CXX) $(CXXFLAGS) $(OBJS) -o $(EXE)
```

- [x] client.o
- [x] digits.o
- [ ] $(EXE)
- [ ] digit-tester

- [ ] None of these

Question 18

1 / 1 pts

Examine the code below.

```
int mystery3(int n) {
    if (n < 2) return 1;
    return n * mystery3(n - 1);
}
```

- [ ] mystery3 is a recursive wrapper
- [ ] mystery3 has a stack overflow for some numbers.
- [x] mystery3 correctly implements the Factorial algorithm
- [x] mystery3 completes for all inputs

mystery3 is an implementation of the Factorial algorithm. It completes for all inputs, but negative inputs produce the wrong output. It is efficient and it is not a wrapper. if (n < 2) is a base case.

Question 19

1 / 1 pts

What is the value of ***r("xhixhix")***?

```
string r(const string& s)
{
    if (s.size()) {
        auto c = s.at(0);
        auto t = c == 'x' ? 'y' : c;
        return t + r(s.substr(1));
    }
    return 0;
}
```

- [x] yhiyhiy
- [ ] xyyxyyx
- [ ] yyyyyyy
- [ ] Stack overflow
- [ ] xyhixyhixy

Incorrect

Question 20

0 / 1 pts

What does this function do?

```
int mystery(int n)
{
  if (n == 1) return 1;
  return n + mystery(n-1);
}
```

- [ ] Computes the Gauss series (sum) of 1..n

- Produces a stack overflow
- Computes the reverse of the input n
- Computes the Fibonacci number n
- (selected) Computes the Factorial number n

Question 21

1 / 1 pts

One remarkably simple formula for calculating the value of π is the so-called Madhava–Leibniz series: $\frac{\pi}{4} = 1-\frac{1}{3}+\frac{1}{5}-\frac{1}{7}+\frac{1}{9}-\ldots$ Consider the recursive function below to calculate this formula:

```
double computePI(int number)
{
    if (number <= 1) { return 1.0;}
    int oddnum = 2 * number - 1;
    return  computesign(number) * 1.0 / oddnum
        + computePI(number - 1);
}
```

In this recursive function, what is the recursive base case?

- When the parameter variable is greater than one
- When the value that is returned from the function is zero
- When the parameter variable is zero
- (selected) When the parameter variable is less than or equal to one

Incorrect

Question 22

0 / 1 pts

When using the `get()` member function to read a character, leading whitespace is not skipped.

- True
- (selected) False

Question 23

1 / 1 pts

Which line opens the file `in.txt` for reading?

- None of these
- `ofstream in; in.open("in.txt");`
- (selected) `ifstream in("in.txt");`
- `istream in("in.txt");`
- `ifstream open("in.txt");`

Question 24

1 / 1 pts

What is the value of ***r("axxbxx")***?

```
string r(const string& s)
{
```

```
    auto front = s.substr(0, 1);
    if (front.empty()) return "";
    return (front == "x" ? "" : front) + r(s.substr(1));
}
```

- ◉ "ab"
- ○ "ax bx "
- ○ "a b "
- ○ Stack overflow
- ○ "xxxx"

Question 25

1 / 1 pts

When using the `get()` member function, a stream will fail only if there are no characters left in the input stream.

- ◉ True
- ○ False