Q-07 Results

Correct answers are hidden.

Submitted Jun 27 at 7:53am

Question 1

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
- bounds precondition
- advancing the loop
- [✓ loop bounds]
- goal operation
- goal precondition

Question 2

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
Can my loop reach its bounds?	necessary bounds
Has my loop reached its goal?	loop postcondition
What makes this loop quit?	loop bounds

Incorrect

Question 3

0 / 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 boundary condition]

○ None of these

○ a postcondition

○ a loop guard

○ an intentional condition

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

Below is the illustration from the loop building strategy. The ***highlighted lines*** represent.
Store the next character from str in current-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
```

[✓ ● advancing the loop]

○ loop postcondition

- goal operation
- goal precondition
- loop bounds
- bounds precondition

Incorrect

Question 6

0 / 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
- None of these
- [✓ a boundary condition]
- an intentional condition
- a loop guard

Question 7

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
- goal precondition
- [✓ bounds precondition]

advancing the loop

Incorrect

Question 8

0 / 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 loop guard
- None of these
- [✓ a boundary condition]
- an intentional condition
- a postcondition

Incorrect

Question 9

0 / 1 pts

Below is the illustration from the loop building strategy. The ***highlighted lines*** represent:

```
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 operation
- goal precondition
- bounds precondition
- [✓ advancing the loop]
- loop bounds

Question 10

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;
```

- 2
- [✓ 4]
- 5
- None of these

Question 11

1 / 1 pts

In `H05`, here is code for the loop that is used. What is the underlined portion?

```
for (size_t i{0}, len{str.size()}; i < len; ++i)
{
}
```

- the loop operation
- the goal precondition
- the loop bounds
- advancing the loop
- [✓ the bounds precondition]
- the loop postcondition

Partial

Question 12

0.25 / 1 pts

In `H05`, here is the pseudocode for the loop body. Which strings **will not be correctly processed** by this loop?

```
sum <- 0
number <- 0
for each character in str
    Set current character -> ch
    If ch is a digit then
        digit <- ascii-to-decimal(ch)
        number <- number * 10
        number <- number + digit
    Else
        sum <- sum + number
        number <- 0
```

- "a5r8rfl"
- "yl39x"
- "vl12013s"
- "dhvj7y365ut85019"
- [✓ "93";]
- [✓ "0uiw5x2v8lx";]

☐ "dir39"

☑ "9165847y44"

Question 13

1 / 1 pts

In H05, here is the pseudocode for the loop body. What code would turn an ASCII character into its digit value?

```
sum <- 0
number <- 0
for each character in str
   Set current character -> ch
   If ch is a digit then
      digit <- ascii-to-decimal(ch)
      number <- number * 10
      number <- number + digit
   Else
      sum <- sum + number
      number <- 0
```

○ `int digit = ascii_to_decimal(ch);`

○ `int digit = ch - "0";`

○ None of these answers is correct

[✓ `int digit = ch - '0';`]

○ `int digit = static_cast<int>(ch);`

Partial

Question 14

0.5 / 1 pts

In H05, here is the pseudocode for the loop body. Which strings **will be correctly processed** by this loop?

```
sum <- 0
number <- 0
for each character in str
   Set current character -> ch
   If ch is a digit then
      digit <- ascii-to-decimal(ch)
      number <- number * 10
      number <- number + digit
   Else
      sum <- sum + number
      number <- 0
```

☑ "yl39x"

☑ "a5r8rfl"

☑ "dir39"

☑ "0uiw5x2v8lx";

☐ "93";

☐ "9165847y44"

☐ "vl12013s"

☐ "dhvj7y365ut85019"

Question 15

1 / 1 pts

Here is an implementation of `everyNth()` from H06. What is its problem?

```
string everyNth(const string& str, int n) {
    string result;
    for (size_t i = 0, len = str.size(); i < len; ++i) {
        if (i % n == 0) {
            result += str.at(i);
        }
    }
    return result;
}
```

- It does not compile
- It should use `str.substr(i, 1)` instead of `str.at(i)` since it's more efficient
- [✓ It produces the correct output, but is less efficient than it could be]
- There is no problem. It works correctly and is efficient.
- It does not produce the correct output for every input, only some

Question 16

1 / 1 pts

Here is an implementation of `countCode()` from `H06`. What is its problem?

```
int countCode(const std::string& str) {
    int result = 0;
    for (size_t i = 0, len = str.size() - 3; i < len; ++i)
    {
        string subs = str.substr(i, 4);
        if (subs.substr(0, 2) == "co" && subs.back() == 'e')
        {
            result++;
        }
    }
    return result;
}
```

- It does not compile
- It produces the correct output for all input values
- It works correctly, but you should use `int` for your indexes, not `size_t`
- It compiles, but the loop should use `len = str.size() - 4`
- [✓ It produces incorrect output for strings with a length less than 3]

Question 17

1 / 1 pts

Here is an implementation of `prefixAgain()` from `H06`. What is its problem?

```
bool prefixAgain(const string& str, int n) {
    string prefix = str.substr(0, n);
    for (size_t i = 0, len = str.size(); i < len; ++i) {
        string word = str.substr(i, n);
        if (word == prefix) { return true; }
    }
    return false;
}
```

- It does not compile because you can't use == with strings
- [✓ It compiles and runs without crashing, but doesn't always produce the correct output]
- It doesn't compile because there is no `else` with the `if`

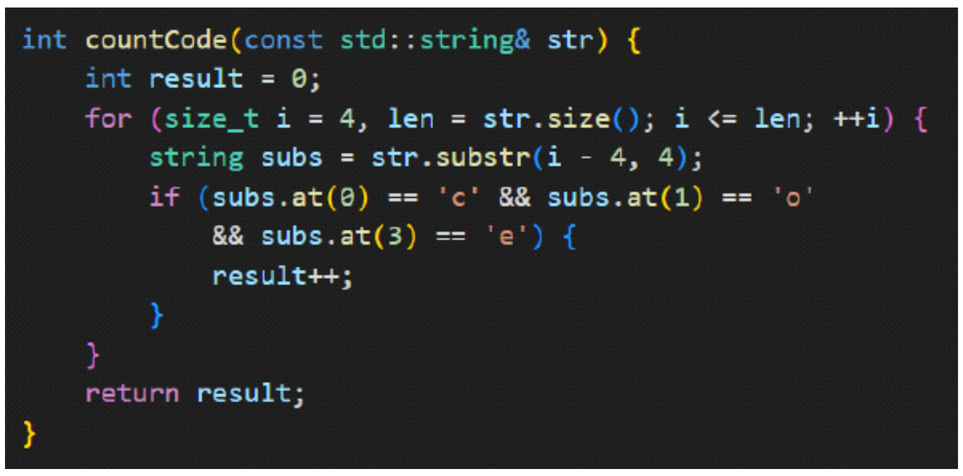
- It compiles and runs without crashing, but never produces the correct output
- There is no problem. It works correctly and is efficient.

Incorrect

Question 18

0 / 1 pts

Here is an implementation of `countCode()` from `H06`. What is its problem?



```
int countCode(const std::string& str) {
    int result = 0;
    for (size_t i = 4, len = str.size(); i <= len; ++i) {
        string subs = str.substr(i - 4, 4);
        if (subs.at(0) == 'c' && subs.at(1) == 'o'
            && subs.at(3) == 'e') {
            result++;
        }
    }
    return result;
}
```

- It does not produce the correct output
- It produces the correct output but performs more iterations than required
- It compiles, but the loop condition should be `i < len;`
- There is no problem. It works correctly and is efficient.
- [✓ It does not compile]