IF-ELSE Statement

Problem F

Chin Ka Wang {rina__owo}

The 1st Buddhist Sin Tak College Computer Club Programming Contest

October 18, 2024

stpc();

Background

Problem Idea by `rina__owo`
Preparation by `rina__owo`, `pepper1208`

Problem Restatement

- In this question, the code for an ‘if-else‘ statement must follow this format:

```
if(//condition){
    cout<<//s1;
}else{
    cout<<//s2;
}
```

If the code has no `Compile Errors` and the `//condition` is true, the program outputs `//s1`, otherwise, it outputs `//s2`.

Problem Restatement

- Given a piece of ‘if-else‘ statement code, output the program’s output.

  If the input code is valid, it must adhere to the following format:

  1. Each line of code contains no spaces between characters.

  2. The first line has the format ‘if(//condition){‘, where ‘//condition‘ is an inequality between two integers. The comparison operator can only be one of ‘==‘, ‘!=‘, ‘<=‘, ‘>=‘, ‘<‘, or ‘>‘.

Problem Restatement

3. The second line starts with any number of spaces and has the format ‘cout«//s1;‘, where ‘//s1‘ is any non-blank string or character (‘string‘ or ‘char‘). Strings must be enclosed in double quotes “""‘, while characters can be enclosed in single quotes “”‘ or double quotes “""‘. The content of the string or character is guaranteed not to contain “"‘, “’‘, ‘; or spaces.

4. The third line must be ‘}else{‘.

5. The fourth line has the same format and definition for ‘//s2‘ as the second line.

6. The fifth line must be ‘}‘.

Problem Restatement

If the input code does not conform to the above format, it is considered a compilation error, and the output should be ‘Compile Error‘.

Statistics

Points are given per checkpoint in this problem. There are 30 checkpoints in this question.

Attempts: 34

0 points	10	+	1	=	11
67 points	0	+	2	=	2
100 points	0	+	0	=	0

First solved by **No one**!

Solution

This question is a large-scale **simulation** and **string processing** question.

The STL function `substr()` is extremely important in this question.

Solution

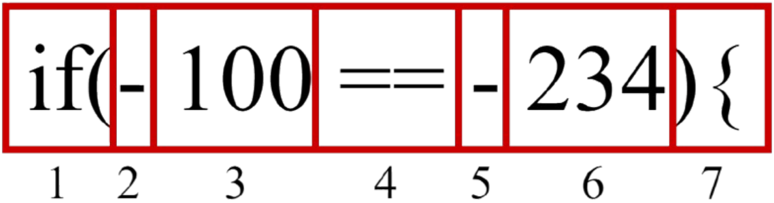
We can treat the code as five inputs. Each input is a line of the code.

Set up 3 variables `num1, op, num2` to store the first number, the operator, and the second number respectively. You are recommended to declare the variables in `string`, as we are storing the information character by character, instead of directly storing the whole integer inside. The `string` data type can implement this operation well.

Set up some `while loop`s to iterate different parts of the characters in the condition. We can access the i^{th} character (i starts with 0) of the input `inp` simply through `inp[i]`.

Solution

In line 1, we can actually break down the code into 7 parts:



```
if(-100==-234){
```

Solution

1. use the `substr()` function to check whether the first 3 characters are `if(`.

2. Note that the integers in the condition can be negative. We should start by checking whether the first character is a `-` sign, using a `if` statement. Store the negative symbol into the end of `num1` if there is any.

3. Set up a `while loop` and use the STL function `isdigit()` to keep iterating the digits in the first number. When the `while loop` ends (`isdigit()` returns `false`), we know that the first integer is over. Check if the number is valid.

Solution

4. Set up a `while loop` and use `if` statement to store the operator. The `if` statement should check whether the characters are one of the following: `<`, `>`, `=`, `!`. After the `while loop` ends, we know that the operator is over. Check if the operator is valid.

5. Repeat step 2 but store the symbol in `num2` if any.

6. Repeat step 3 but store the number in `num2`.

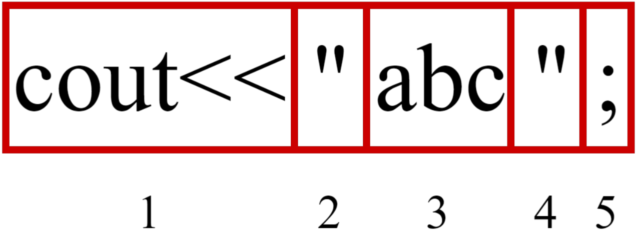
7. Using the STL function `substr()`, check whether the remaining characters are exactly `){`.

Solution

By using `if` statements to detect the operators, calculate the condition by casting `num1` and `num2` back to integers. Store the result into a boolean `condition`.

Solution

In line 2, we can break down the code into 5 parts:



cout<< " abc " ;

1 2 3 4 5

As cin» of c++ automatically ignores all leading spaces, we don’t need to deal with it in this question. However, we need to use the trim() function to remove them if you are using python.

Solution

1. Using `substr()`, check whether the first 6 characters are `cout«`.

2. Detect the whether the quotation mark is single or double. Store it into a variable `type`.

3. Set up a `while loop` to iterate all the following characters until we detect a `" or ’`. Store the output into a variable `ans` if the `condition` is `true`.

Solution

4. Detect the quotation mark after the output string is the same as the one stored in `type`. Besides, if `type` is `'`, check whether the output string is one character only. Note that when we are detecting quotation marks, the program might misunderstood your code as quotation marks are used to declare a `string` or `char`. Therefore we have to add a \ in front of the quotation marks.

5. Check whether the remaining character is `;` by using `substr()`.

Solution

In line 3, we can simply check whether the input is `else{`.

In line 4, repeat the operation we did in line 2, but store the output string into `ans` if the `condition` is `false`.

In line 5, check whether the input is `}`.

Solution

- If the input violates any of the operation above, we can directly output `Compile Error` and stop the whole program by `return 0;`.

  However, if the condition is `true` in line 1, we should not output the string in line 2 immediately and terminates the program as there might be compile errors in the `else` part or even till the end of the code.

  Instead, we should iterate the whole code first. If the whole code does not have compile errors, output `ans`.

Solution

The ability of thinking and dealing with **corner cases** is extremely important in this questions. If you can't come out with some corner cases, you might lose lots of marks.

Solution

Examples of corner cases:

1. `num1` or `num2` is `-` or blank.

2. The operator is something like `===`.

3. There are multiple `}` s at the end of line 1, but your program just checked if the 2 character after the condition is `){`. In this case, `){{{{` will be claimed as no compile error.

4. Your program claimed `-0 == 0` to be `false`.

5. Your program misidentifies having only a character in `"  "` as compile error.

Takeaways

1. Make good use of STL functions.
2. Be familiar of c++ syntax. E.g. `\"`.
3. Think carefully about the corner cases.