プログラミング入門
#2 条件分岐

高校 3 年 赤澤侑

2021/05/04

1　if - 条件分岐その 1

条件分岐のやり方を学びましょう。C++ で条件分岐 (もし〜ならば...) をプログラムに利用するときは次のように記述します。

```
if（条件式）{
    処理
}
```

`if` のあとの丸カッコ内には条件式を記述します。例えば `a > 0` や `a == 0`, `a <= 0` などです。そして中カッコの中に、条件を満たす場合に実行したい処理を書きます。

また条件を満たさない場合の処理を書くこともできます。次のコードを見てください。

```
if（条件式）{
    条件を満たす場合の処理
} else {
    条件を満たさない場合の処理
}
```

このプログラムに追加された `else` という語が条件を満たさない場合の処理を記述するためのキーワードです。`else` の後に続く中カッコの中に条件を満たさない場合の処理を記述します。

では次の問題を問いてみましょう。

Problem 1 : Greeting Law 1

マイコン王国で新しい法律が施行された。マイコン王国の人々はハローと話しかけられたらハローとかえし、それ以外の言葉で話しかけられた場合には無視しなければならない（悲しいね）。あなたにかけられた言葉が与えられる。その言葉が ‘Hello’ であった場合には ‘Hello’ と、それ以外の言葉であった場合には ‘...’ と出力せよ。

制約

文字列の長さは 10^4 を超えない。文字列は大文字、小文字の英字のみからなる。

入出力例 1

Input	Output
Hello	Hello

ちゃんと返事を返してくれました。

入出力例 2

Input	Output
GoodMorning	...

悲しいね。

さて、ここで文字列を扱う変数について考えましょう。文字列というのは文字の集まりですから文字すなわち char[1]の配列と考えることができます。ですが、ここは char 型の配列よりも便利に文字列を扱うことのできる string というクラスを使ってみましょう。string を使うためにはプログラムの最初の部分を次のように変更する必要があります。

```
#include <iostream>
#include <string>
using namespace std;

int main(){

}
```

クラスはあとの章で詳しく扱いますが、ここでは変数のようなものだと考えてください。string 型の変数は int や char などと同じように代入したり cout に渡して中身を出力することができます。実際の使用例は次のプログラムを参考にしてください。

Source Code (1): string の使い方

```
#include <iostream>
#include <string>
using namespace std;

int main(){
    string s = "hogehoge";
    cout << s << endl;

    string t;
    cin >> t;
    cout << t << endl;

    return 0;
}
```

このコードを実行してみると次のような出力が得られます。左が入力、右が出力です。

[1] char は “一文字” を扱うことのできる変数です。

Input/OutPut (1): string の使い方 - 実行結果

piyopiyo	hogehoge piyopiyo

では string 型の使い方がわかったところで例題を解いてみましょう。回答例は次のとおりです。自力で回答を作成したい方は一度ここで考えてみましょう。

Source Code (2): Greeting Law1 の回答例

```
1  #include <iostream>
2  #include <string>
3  using namespace std;
4
5  int main(){
6      string s;
7      cin >> s;
8      if(s == "Hello"){
9          cout << "Hello" << endl;
10     } else {
11         cout << "..." << endl;
12     }
13
14     return 0;
15 }
```

プログラムの雛形部分と入力の受け取りに関する説明は省きます。さて、8 行目で `s == "Hello"` という条件式が用いられています。左辺と右辺の内容が一致しているかを判別するときには==というように、イコールを 2 つ並べる必要があります。では `s = "Hello"`という記述は誤りなのでしょうか？はい、誤りです。実際このようにコードに書き換えてみるとエラーが出るはずです[2]。

この問題を解くことができたら次の問題へ移りましょう。

[2] なお、これは string 型の場合の挙動であって、int や double では異なった挙動をします。具体的には代入する値が 0 のときには条件式は false と判断され、それ以外の値のときは true と判断されます。

Problem 2 : Greeting Law 2

大変だ！ Greeting Law に問題が見つかった！
‘Hello’ の代わりに ‘hello’ と挨拶した人が無視され始めたのだ！彼らにも温かい言葉をかけてやらねばならない。
あなたにかけられた言葉が与えられる。その言葉が ‘Hello’ あるいは ‘hello’ であった場合には ‘Hello’ と、それ以外の言葉であった場合には ‘...’ と出力せよ。

制約

文字列の長さは 10^4 を超えない。文字列は大文字、小文字の英字のみからなる。

入出力例 1

Input	Output
hello	Hello

小文字挨拶さんもにっこりです。

入出力例 2

Input	Output
GoodMorning	...

悲しいね。

```cpp
string res;
if (s == "Hello") {
    res = "Hello";
}
if (s == "hello") {
    res = "Hello";
}
if (s == ""){
    res = "...";
}
cout << res << endl;
```

```cpp
if (s == "Hello") {
    cout << "Hello" << endl;
} else if (s == "hello") {
    cout << "Hello" << endl;
} else {
    cout << "..." << endl;
}
```

```cpp
if (s == "Hello" || s == "hello") {
    cout << "Hello" << endl;
} else {
    cout << "..." << endl;
}
```

Problem 3 : Greeting Law 3

@∑ : 「hElLo」
王様 : 「……まじかよ」

法律の施行から 3 日後、様々なハローを使う人々が現れた。そこで政府は “ハロー文字列”を以下のように定義し、ハロー文字列に対してのみ挨拶を返して良い、ということにした。

ハロー文字列 : 大文字、小文字を問わず、英字のエイチ、イー、エル、エル、オーをこの順番に連結した文字列

あなたにかけられた言葉が与えられる。その言葉がハロー文字列であった場合には ‘Hello’ と、それ以外の言葉であった場合には ‘...’ と出力せよ。

制約

文字列の長さは 10^4 を超えない。文字列は大文字、小文字の英字のみからなる。

入出力例 1

Input	Output
hElLo	Hello

入出力例 2

Input	Output
GutenMorgen	...

悲しいね。

```cpp
#include <iostream>
#include <string>
#include <regex>
using namespace std;

int main(){
    string s;
    cin >> s;
    if (regex_match(s, regex("HELLO", regex_constants::icase))) {
        cout << "Hello" << endl;
    } else {
        cout << "..." << endl;
    }
    return 0;
}
```

```cpp
for (int i = 0; i < s.size(); ++i) {
    if ('a' <= s[i] && s[i] <= 'z') {
        s[i] = s[i] - 'a' + 'A';
    }
}
if (s == "HELLO") {
    cout << "Hello" << endl;
} else {
    cout << "..." << endl;
}
```

```cpp
#include <iostream>
#include <string>
#include <algorithm>
using namespace std;

int main(){
    string s;
    cin >> s;
    transform(s.begin(), s.end(), s.begin(), ::toupper);
    if (s == "HELLO") {
        cout << "Hello" << endl;
    } else {
        cout << "..." << endl;
    }
    return 0;
}
```

transform について、引数はイテレータの開始位置、終了位置、コピー先のイテレータの開始位置、適用する処理。グローバル名前空間との衝突に注意。