CS 161 – Day 7 Notes

John Waczak

4/16/2018

1 Named constants

Most of the time people just call them constants. They are different from variables (named chunks of memory we can control) and literals (literal values). A constant is a reserved, named piece of memeroy but the value never changes. When you declare you have to say what the value is.

Constants are easy to change (think for example sales tax that might change year to year) and much better for catching typographical errors than if you manually type out the same thing over and over again.

```
#include <iostream>
#include <string>

using std::cin;
using std::cout;
using std::endl;
using std::string;

int main()
{
  const double SALES_TAX_RATE = 7.25;
  cout << SALES_TAX_RATE << endl;
  return 0;
}
```

```
7.25
```

Be careful about declaring global variables (variables outside of functions and the main loop). This has to do with scope. Global variables are BAD

however global constants are acceptible and make sense.

2 Decision makeing (If-statements)

An If-statement allows us to evaluate boolean conditions...

```
#include <iostream>
#include <string>

using std::cin;
using std::cout;
using std::endl;
using std::string;

int main()
{
  const double SALES_TAX_RATE = 7.25;
  if (SALES_TAX_RATE > 5.0)
  {
    cout << "true\n";
  }
  else
  {
    cout << "false\n";
  }
  return 0;
}
```

```
true
```

You can do without the curly braces if you only have one line in your statement. You can also nest your if statements to make a decision tree.

```
#include <iostream>
#include <string>

using std::cin;
using std::cout;
using std::endl;
```

```
using std::string;

int main()
{
  int num = 12;

  if (num > 10)
    cout << " num > 10\n";
  else if (num == 9)
    cout << " num is exactly 9\n";
  else
    cout << " num < 9";



  return 0;
}

num > 10
```

3 Loops

The simplest type of loop is probably the while loop. It uses a conditional expression which, when true, continues to loop throug the curly braces.

```
#include <iostream>
#include <string>

using std::cin;
using std::cout;
using std::endl;
using std::string;

int main()
{
  int num = 0;

  while (num <= 10)
    {
```

```
      cout << num << endl;
      num ++ ;
    }
  return 0;
}
```

```
0
1
2
3
4
5
6
7
8
9
10
```

So we see that the everything starts with the condition. If that condition argument evaluates to true, we then we enter the loop. Once finished with the while-loop's code block, the condition is reevaluated. This continues until the argument evaluates to false.

If you accidentally run an infinite loop (i.e. it gets stuck on always true), type Ctrl-c or Ctrl-d. One of those should force the program to quit.