Lecture 2 Notes

- Although we declared g in the following example, we did not initialize it
- The code will compile, but there will not be an output

```
double g;
double h = 2 * g; // we do not know what ‘g’ is equal to
cout << h;
```

- Let’s say we want a program that will show the following dialogue
    - Note that the text in **bold** are the outputs, and the normal text are your inputs

```
What is your name? Sir Robin
What is your quest? To seek the Holy Grail

Hello brave Sir Robin!
You want To seek the Holy Grail
```

- The program is as follows

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

int main()
{
cout << "What is your name? ";
string personsName;
getline(cin, personsName); // second argument must be a string variable

cout << "What is your quest? ";
string quest;
getline(cin, quest);

cout << "Hello, brave " << personsName << "!" << endl;
cout << "You want " << quest << endl;
}
```

- **Strings** are text, and `getline(cin, personsName)` is used to set up an input that takes up an entire line
    - If you type in `cin >> personsName`, it will only input the first word

      (in this case, `Sir Robin` is two words, so you need to use `getline`)
- Let's expand on the program so that there are integers so that the following dialogue applies:

```
What is your name? Sir Robin
How old are you? 32
What is your quest? To seek the Holy Grail

Hello brave Sir Robin!
You want To seek the Holy Grail
If you live, next year you will be 33
```

- The program is as follows:

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

int main()
{
cout << "What is your name? ";
string personsName;
getline(cin, personsName); // second argument must be a string variable

cout << "How old are you? ";
int age;
cin >> age;
cin.ignore(10000, '\n');

cout << "What is your quest? ";
string quest;
getline(cin, quest);

cout << "Hello, brave " << personsName << "!" << endl;
cout << "You want " << quest << endl;
cout << "If you live, next year you will be " << age+1 << endl;
}
```

- Note that we cannot use `getline` for a number… it only applies for strings
- If you read in a **number**, and the next thing you read is a **string** using `getline`, you need to type
  `cin.ignore(10000, '\n');`
  - Otherwise, the string input will be read as empty
- Let's revisit the code we wrote during Lecture 1:

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

int main()
{
cout << "How many hours did you work? ";
double hoursworked;
cin >> hoursworked;

cout << "What is your hourly rate of pay? ";
double payRate;
cin >> payRate;

double amtEarned = hoursworked * payRate;
cout.setf(ios::fixed);
cout.precision(2);
cout << "You earned $" << amtEarned << endl;
cout << "$" << (0.10 * amtEarned) << " will be withheld." << endl;
}
```

- **If-statements** have the following structure

```
if (condition)
      statement-if-true

else
      statement-if-false
```

- Adding onto the code written in Lecture 1,

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

int main()
{
cout << "How many hours did you work? ";
double hoursworked;
cin >> hoursworked;

cout << "What is your hourly rate of pay? ";
double payRate;
cin >> payRate;

double amtEarned = hoursworked * payRate;
cout.setf(ios::fixed);
cout.precision(2);
cout << "You earned $" << amtEarned << endl;

if (payRate >= 18.00)
cout << "$" << (0.10 * amtEarned) << " will be withheld." << endl;

else
cout << "$" << (0.05 * amtEarned) << " will be withheld." << endl;
}
```

- Signs for conditions are written in the following way:

```
oneThing < anotherThing
oneThing <= anotherThing
oneThing > anotherThing
oneThing >= anotherThing
oneThing != anotherThing // is not equal to
oneThing == anotherThing // is equal to
```

- There are rare instances where `>` and `>=` or `<` and `<=` are not distinguished

```
if (x >= 0)
cout << x << endl;

else
cout << -x << endl; // If x=0, x and -x would be the same thing
```

- Similar to joining sentences together using the word and in English, we can write something called a **compound statement** or **block**
    - This is in the form of
      `{statement; statement; statement;}`

```
if (x > 0)
{
cout << "Hello" << endl;
double z = 10;
...
cout << z << endl;
} // Compound statement

else
cout << "Wow!" << endl;
```

- If-statements also work if there are blank inputs:

```
cout << "What is your name? ";
string name;
getline(cin, name);
if (name == ""); // Blank input
cout << "You didn't type a name!" << endl;
else
cout << "Hello there." << endl;
```

- Sometimes, you can emit the "else" part of the if-statement if you do not want anything to happen under certain conditions

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

int main()
{
cout << "How many hours did you work? ";
double hoursworked;
cin >> hoursworked;

cout << "What is your hourly rate of pay? ";
double payRate;
cin >> payRate;
if (payRate < 15.00) // Single if-statement
cout << "Ask for a raise!" << endl;

double amtEarned = hoursworked * payRate;
cout.setf(ios::fixed);
cout.precision(2);
cout << "You earned $" << amtEarned << endl;

if (payRate >= 18.00)
cout << "$" << (0.10 * amtEarned) << " will be withheld." << endl;

else
cout << "$" << (0.05 * amtEarned) << " will be withheld." << endl;
}
```

- An **assignment statement** is used to set a variable that has already been declared to an expression

- The format of an assignment statement is as follows
  `variable = expression;`

- You cannot declare variables inside an if-statement

  - Thus, you must declare it before the if-statement

  - Use an assignment statement inside the if-statement

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

int main()
{
cout << "How many hours did you work? ";
double hoursworked;
cin >> hoursworked;

cout << "What is your hourly rate of pay? ";
double payRate;
cin >> payRate;
if (payRate < 15.00)
cout << "Ask for a raise!" << endl;

double amtEarned = hoursworked * payRate;
cout.setf(ios::fixed);
cout.precision(2);
cout << "You earned $" << amtEarned << endl;

double withholdingRate; // Variable is declared

if (payRate >= 18.00)
withholdingRate = 0.10; // Assignment statement inside if-statement

else
double withholdingRate = 0.05;

cout << "$" << (withholdingRate * amtEarned) << " will be withheld." << endl;
}
```