Installing Atom and the compiler

Go to this URL:

tiny.cc/cppclass

Remember your keybinds

- **Ctrl + N** or **Cmd + N:** New file (save as **<filename>.cpp**)
- **Ctrl + S** or **Cmd + S:** Save and check your code
- **F5:** Compile and **Run** your code

Day 1

Part 1

~~No Code; Just Data~~

Data and a tiny bit of code

1. Variables
2. Data Types
3. Data Structures
4. Simple Data Operations
5. Basic Input/Output

Variables

- Portions of memory in which we can store and access data
- **Identifier:** The *“name”* of a variable
  - It serves to distinguish it from the other variables
  - In C++: Case Sensitive, Must start with a letter

Data Types

- Specify *what* type of data a variable stores
- C++ is **statically typed**; It needs to know the type of a variable

Name	**Description**	**Size ***	**Range ***
int	An integer number.	4 bytes	-2147483648 to 2147483647 unsigned: 0 to 4294967295
float or double	A decimal number.	4/8 bytes	7 digits / 15 digits
bool	A boolean value.	1 byte	true or false
char	A character. (Stored as ASCII)	1 byte	0 to 255
string	A sequence of characters.	1 to it's complicated	From "hello world" to "Rocket landed successfully"

* Depends on the system and the compiler.

Declaring Variables

Always end with a **semicolon!**

- `type identifier = initial_value;`

Examples:

- `float a;`
- `char hey = ‘a’;`
- `unsigned int num = 4294967293;`
- `bool k;`

Note:
Declaring variables with no specified value gives them a “random” initial value.

Variable Operations

- Assignment (=)
  - `a=b=c=d=5;`
- Arithmetic Operators (+, -, *, /, %)
  - `a=2+b;`
- Compound Assignment (+=, -=, *=, /=, %=, *>>=, <<=, &=, ^=, |=*)
  - `a/=2; → a=a/2;`
  - `a%=b; → a=a%b;`
- Increase and Decrease
  - `a++; → a=a+1;` and `hey--; → hey=hey-1;`

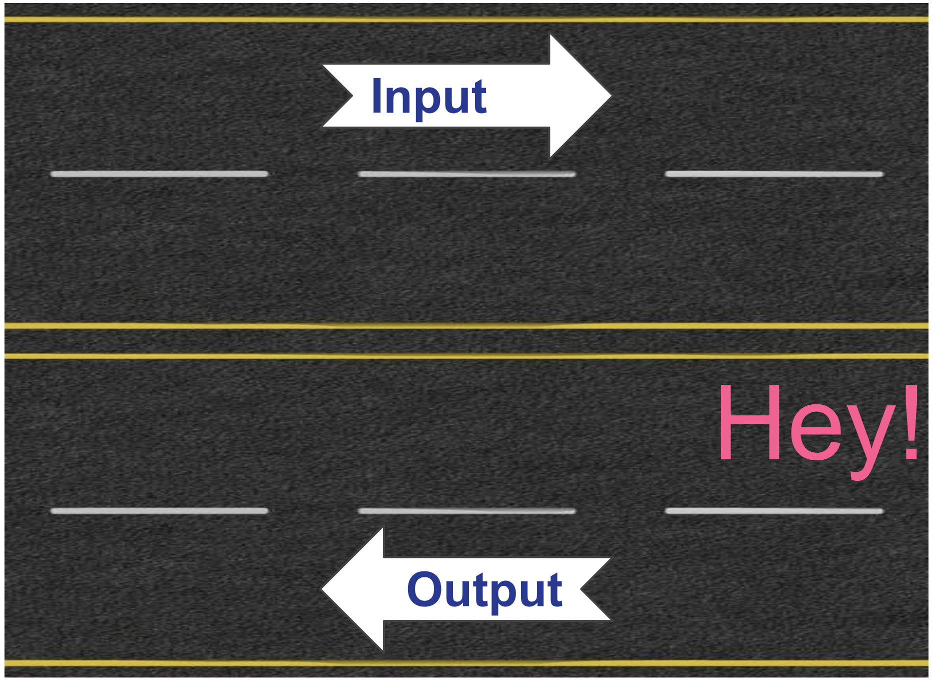
- Relational Operators (==, !=, >, <, >=, <=)
  - `a==b → True when a and b are the same`
  - `a!=b → False when a and b are the same`
  - `a>=b → True when a is greater than or equal to b`
- Logical Operators (||, &&, !)
  - `a||b → a OR b`
  - `a&&b → a AND b`
  - `!a → NOT a`

Basic Input/Output

A collection of useful code that has been already implemented.

- Standard I/O is included in the iostream **library**
  - `#include <iostream>`
- Prints and receives messages to and from the console
- Can store input in variables
- Can extract output from variables

What’s the *console* stream?



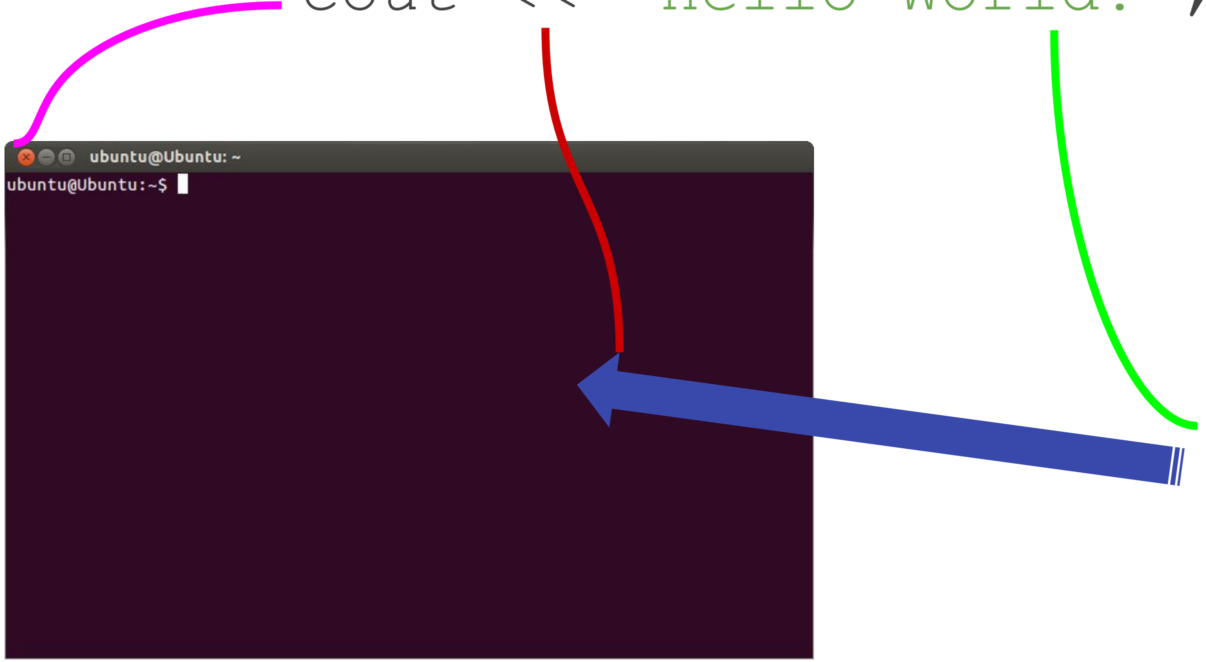
```
#include <iostream>
using namespace std;

int main() {
  cout<<"Hey!";
  return 0;
}
```

Output

Notice how the ***insertion operator*** points to the console **out** stream.

```
cout << "Hello World!";
```



```
ubuntu@Ubuntu:~$
```

```
Message:
```

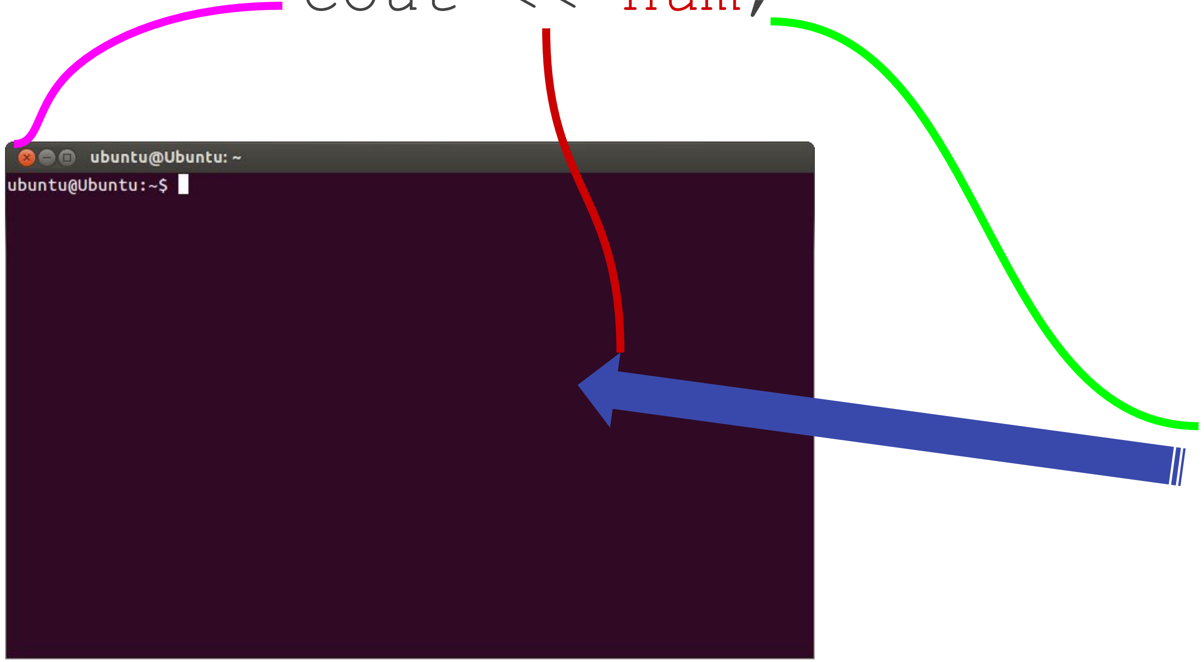
Hello World!

Output

Notice how the ***insertion operator*** points to the console **out** stream.

```
unsigned int num = 4294967293;
```

```
cout << num;
```



```
ubuntu@Ubuntu:~$
```

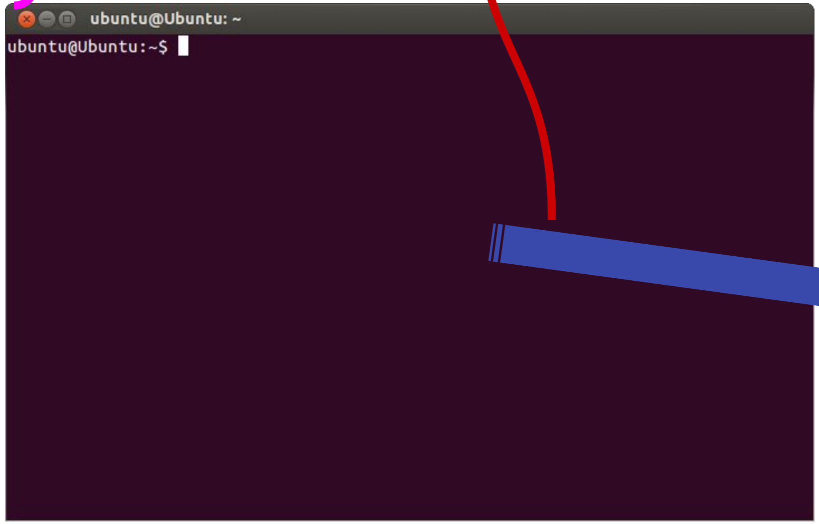
Message:

4294967293

Input

Notice how the ***insertion operator*** points from the console **in** stream to the **variable**.

```
cin >> num;
```



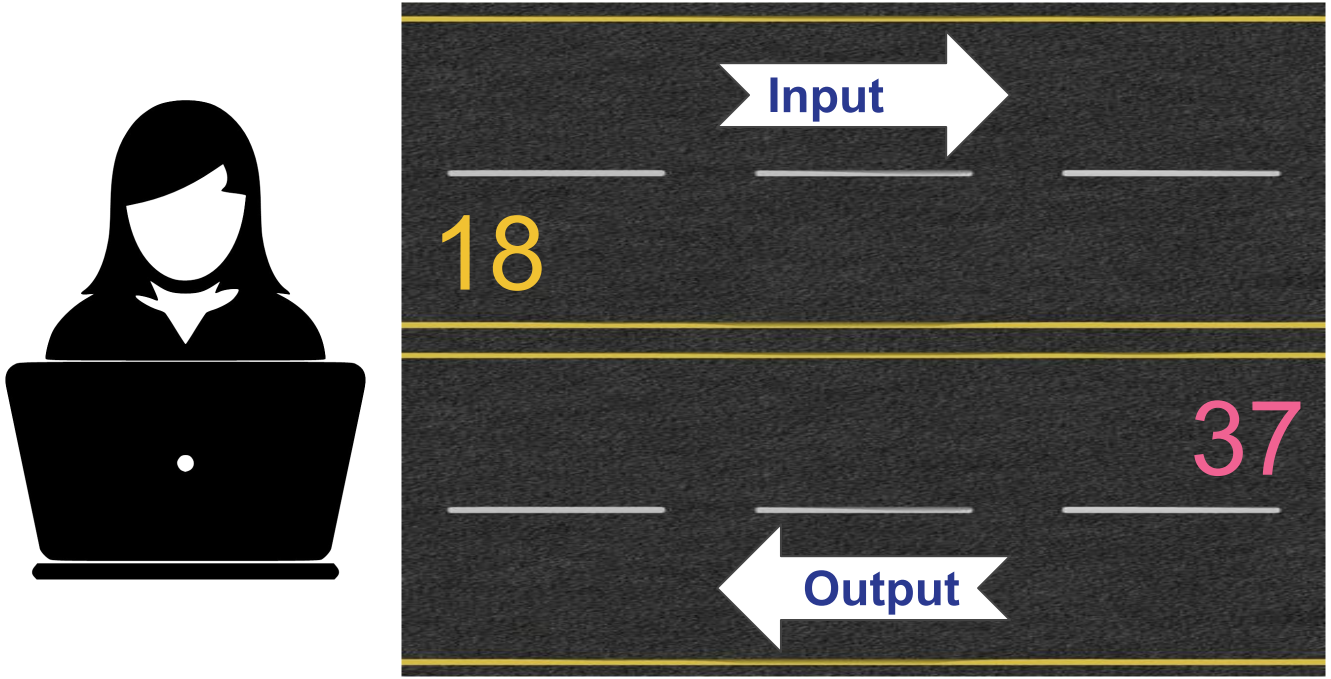
```
ubuntu@Ubuntu:~$
```

At this point, the code waits for the user to type something **until they press enter**.

`Set num to:`

Whatever the user types

What’s the *console* stream?



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

int main() {
  int num;
  cin>>num;
  cout<<num*2+1;

  return 0;
}
```

Chaining insertion operators

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

int main() {
  int age;
  cout<<"How old are you? ";
  cin>>age;
  cout<<"You will be "<<age+1<<" next year.\n";
  cout<<"Neat!";

  return 0;
}
```

\n means new line
\t means tab

Remember the syntax

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

int main() {
}
```

Test your compiler

Write a program to:

1. Input a number from the user
2. Find the square of that number ($x^2 = x*x$)
3. Output: “The square of your number is: ” and the number

Summary of console streams

- The **cin** stream
  - Takes data the user types in the console
  - Assigns it to a variable with the **>>** operator
- The **cout** stream
  - Shows data in the console
  - Assigns it to a variable with the **<<** operator
- Insertion operators can be chained

Day 1

Part 2

Making Decisions

1. Compound Statement
2. Conditional Structure

Compound Statements or Blocks

- Single statements are commands such as `cout<<"wat";`
- Enclosing many of them in brackets forms a block

-
  ```
  {
    cout<<1821/0;
    cout<<"wat";
  }
  ```

Conditional Structure

```
if(condA){
  <code to be run if condA is true>
}
else if(condB){
  <code to be run if condA is false and condB is true>
}
else{
   <code to be run if none of the above were true>
}
```

If can take compound and single statements.

Conditional Structure

```
if(2 == 1){
  cout<<"2 is equal to 1!";
  cout<<"wat";
}
else if(3 == 2){
  cout<<"3 is equal to 2!";
  cout<<3/0;
}
else{
  cout<<"Why even bother evaluating these?";
}
```

Conditional Structure

```
if (x > 0)
   cout<<"x is positive";
else if (x < 0)
   cout<<"x is negative";
else
   cout<<"x is 0";
```

Note:
If you only want to execute a single statement, no brackets are required.

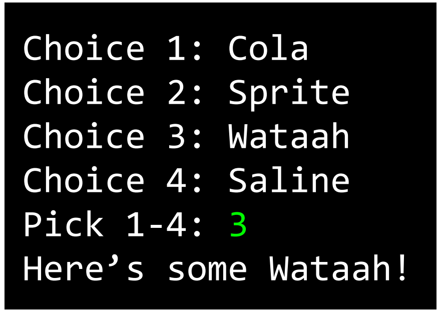
Grading Program

Write a program that allows the user to enter a grade **(0-100)**

1. If the user scored a 100 then notify the user that they got a perfect score
2. Modify the program so that if the user scored a 90-100 it informs the user that they scored an A
3. Modify the program so that it will notify the user of their letter grade
    - 0-59 F 60-69 D 70-79 C 80-89 B 90-100 A

Cola Machine

- Write a program that presents the user with a choice of your 4 favorite beverages
- Then allow the user to choose a beverage by entering a number 1-4
- Output which beverage they chose
    - If they type a wrong number, output an error message



```
Choice 1: Cola
Choice 2: Sprite
Choice 3: Wataah
Choice 4: Saline
Pick 1-4: 3
Here’s some Wataah!
```

Ordering Tickets

- A ticket costs 10€
- Order at least A tickets → 10% discount
  - B → 20%   C → 30%   D → 40%
- You want to order N tickets
- **Inputs *(with cin)*:** N, A, B, C, D
- **Output:** The minimum sum of money you can pay
  - **Note:** It is possible that you can order more tickets to get a lower price!

Double Time

Given the output of a stopwatch (HH:MM:SS) find what the stopwatch will output at twice that time.

Switch Case

```
switch (variable){
  case (possible value):
    Commands;
    break;

  default:
    Commands;
    break;
}
```