

1 Week 3/4

Week 3 material in this packet covers user input. Week 4 material covers control flow. Feel free to switch the order of user input and control flow if your instructor decides to cover control flow first.

1.1 User Input

Here is a list of the User Input rules in one place.

How `cin >> variable;` works:

1. If there is nothing in the input buffer, allow the user to enter input.
2. Move past all whitespace (spaces, tabs, `'\n'`).
3. Repeat steps 1 and 2 until something that is not whitespace is encountered.
4. When possible, interpret what follows as something of the same type as the type of `variable`, assign this to `variable`, and strip it from the input buffer. However, never move past whitespace in this step.

How `string s; getline(cin, s)` works:

1. If there is nothing in the input buffer, allow the user to enter input.
2. Remove everything in the input buffer before `'\n'` and assign it to `s`.
(This could result in `s == ""`.)
3. Remove `'\n'` from the input buffer.

How `cin.ignore()` works:

1. If there is nothing in the input buffer, allow the user to enter input.
2. Remove the first character from the input buffer.

How `char ch = cin.get()` works:

1. If there is nothing in the input buffer, allow the user to enter input.
2. Remove the first character from the input buffer, and assign it to `ch`.

How `char ch = cin.peek()` works:

1. If there is nothing in the input buffer, allow the user to enter input.
2. Assign the first character in the input buffer to `ch`; do NOT remove it from the input buffer.

How `cin.fail()` works:

`returns` a boolean indicating the fail state of `cin`. When this is `true`, no functionality of `cin` will work.

How `cin.clear()` works:

1. resets the fail state of `cin` to be `false`.
2. DOES NOT CLEAR THE INPUT BUFFER.

1.2 Exercises

1.2.1 Predict the output exercises

In the following section you will predict the output for snippets of code based on different responses from users. Explain the output based on the user input rules. Add some lines of code such as conditional print statements to debug.

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

int main() {
    cout << "Enter something:" << endl;

    int variable; cin >> variable;
    cout << variable << endl;
    string left_in_buffer; getline(cin, left_in_buffer);
    cout << left_in_buffer << endl;

    return 0;
}
```

A user runs the following code four different times and types the following. Note that they don't actually type '\n'; that's from the user pressing Enter.

- (a) 3 0009\n
- (b) 2.7\n
- (c) \n 28e3\n
- (d) 888cat\n

What is the output each time?

2. The code below is mostly the same except `int` is replaced with `double`.

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

int main() {
    cout << "Enter something:" << endl;

    double variable; cin >> variable;
    cout << variable << endl;
    string left_in_buffer; getline(cin, left_in_buffer);
    cout << left_in_buffer << endl;

    return 0;
}
```

A user runs the following code four different times and types:

- (a) 3 0009\n
- (b) 2.7\n
- (c) \n 28e3\n
- (d) 888cat\n

What is the output each time?

3. The code below is mostly the same except `double` is replaced with `char`.

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

int main() {
    cout << "Enter something:" << endl;

    char variable; cin >> variable;
    cout << variable << endl;
    string left_in_buffer; getline(cin, left_in_buffer);
    cout << left_in_buffer << endl;

    return 0;
}
```

A user runs the following code four different times and types:

- (a) 3 0009\n
- (b) 2.7\n
- (c) \n 28e3\n
- (d) 888cat\n

What is the output each time?

4. `#include <iostream>`
`using namespace std;`
- ```
int main() {
 cout << "Enter something:" << endl;

 int integer;
 double dub;

 cin >> integer >> dub;
 cout << integer << endl;
 cout << dub << endl;
}
```

```

 return 0;
}

```

Predict the output when the user types: 2.73\n

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

```

int main() {
 string text;
 int integer;
 cout << "Please enter something: ";
 cin >> integer;
 cin.ignore();
 getline(cin, text);
 cout << integer << endl;
 cout << text << endl;

 return 0;
}

```

Predict the output when the user types:

- (a) 3\n hello\n
- (b) 3.33\n
- (c) 3 4\n

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

```

int main() {
 cout << "Give me an integer and I'll add it to 1" << endl;
 int sum = 1;

 while (true) {
 int i; cin >> i;

 if (cin.fail()) {
 cout << "I only sum integers" << endl;
 break;
 }

 sum += i;
 cout << "The sum is " << sum << endl;
 cout << "Give me another integer to add to " << sum << endl;
 }
}

```

```

 }
 return 0;
}

```

Write down what the console would display when the user types:

- (a) 2\n
- (b) q\n
- (c) 2\n3\n4\nq\n
- (d) 2 3 4 q\n
- (e) 2.34\n

7. A user runs the script in an IDE, types the message below, and presses Enter.

It's Feb. 17 2023. 99 Ranch open?

```

#include <iostream>
using namespace std;

int main() {
 double d;
 int i;
 char ch;
 string s;

 cin >> s;

 cin >> d;

 cin >> i;

 cin.ignore();

 ch = cin.get();

 getline(cin,s);

 cout << d << endl;
 cout << i << endl;
 cout << ch << endl;
 cout << s << endl;
}

```

```

 return 0;
}

```

### 1.2.2 Practice coding exercises

8. Create a simple calculator that accepts user input. A user should be able to type 2+3 and press 'Enter' and see `The result is 5` printed to the console. A user should also be able to type 2.1-3 (no spaces) and press 'Enter' and see `The result is -0.9` printed to the console. The user should be able to add, subtract, multiple, and divide pairs of numbers.

- Decide what types you want to store the input. You want to store two numbers somehow as well as the symbol for the mathematical operation.
- Write the control flow using `if-else` statements. Here is a short example:

```

#include <iostream>
using namespace std;

int main() {
 double total = 0.25;
 double cost = 10.00;

 if (total < cost) {
 cout << "You do not have enough cash."
 << " Please return with an extra $"
 << cost - total << ".\n";
 }
 else if (total > cost) {
 cout << "Thank you. Your change is $"
 << total - cost << ".\n";
 }
 else if (total == cost) {
 cout << "Thank you." << endl;
 }

 return 0;
}

```

### 1.2.3 Error spotting exercises

Error spotting will be on the exam. Look at the follow code below and state:

- What errors occurred (build, run time, or logic) or if there are none.
- In the case of a logic error, what does the code currently accomplish?
- How the code can be fixed (use the line numbers to help with the description).

A description of how the code should perform will be put in a block comment. After you make your initial guess, type the code in your IDE and run it. Copying very carefully may help you find some of the errors. Play around with the code to fix it.

```
9. 1 #include <iostream>
 2 using namespace std;
 3
 4 /* This code is meant to prompt for your favorite letter
 5 and print out the letter you like most. */
 6 int main() {
 7 cout << "What's your favorite letter?" << endl;
 8
 9 letter; cin >> letter;
 10
 11 cout << "You like the letter " << letter << " most." << endl;
 12
 13 return 0;
 14 }
```

```
10. 1 #include <iostream>
 2 using namespace std;
 3
 4 /* This code is meant to prompt for your favorite letter
 5 and print out the letter you like most. */
 6 int main() {
 7 cout << "What's your favorite letter?" << endl;
 8
 9 int letter; cin >> letter;
 10
 11 cout << "You like the letter " << letter << " most." << endl;
 12
 13 return 0;
 14 }
```

```
11. 1 #include <iostream>
 2 using namespace std;
 3
 4 /* This code is meant to prompt the user for integers,
 5 then sum those integers to the correct sum. */
 6 int main() {
 7 int sum = 0;
 8 int integer;
 9
 10 cout << "Please enter space separated integers then press Enter. ";
```

```

11 cout << "I'll add them together." << endl;
12
13 while (cin.get() != '\n'){
14 cin >> integer;
15 sum += integer;
16 }
17
18 cout << sum << endl;
19 return 0;
20 }

```

```

12. 1 #include <iostream>
 2 #include <string>
 3 using namespace std;
 4
 5 /* A user is meant to be able to type their favorite number,
 6 press Enter, then type their name, press Enter, and have
 7 them stored as 'num' and 'name', respectively. */
 8 int main() {
 9 cout << "What's your favorite number and name?" << endl;
 10
 11 int num; cin >> num;
 12
 13 string name; getline(cin,name);
 14
 15 return 0;
 16 }

```