Repeated decisions using `while`

CS111 Lab
Queens College, CUNY
Instructor: Kent Chin
Repeated Decisions

We make repeated decisions in our everyday lives.

While you can't find what you're looking for, continue searching for it.

Computers can make repeated decisions too!
A **boolean statement** is a statement that evaluates to *true* (1) or *false* (0). We use *comparison operators* to compare variables/ constants against other variables/ constants. 

NOTE: “iff” means *if and only if*

**Comparison (Relational) Operators**

- `(x == y) → true` iff `x “is equal to” y`, *false* otherwise
- `(x != y) → true` iff `x “is not equal to” y`, *false* otherwise
- `(x > y) → true` iff `x “is greater than” y`, *false* otherwise
- `(x >= y) → true` iff `x “is greater than or equal to” y`, *false* otherwise
- `(x < y) → true` iff `x “is less than” y`, *false* otherwise
- `(x <= y) → true` iff `x “is less than or equal to” y`, *false* otherwise

**Logical Operators (Joins and yields boolean statements)**

- `(stmt1 || stmt2) → true` if `stmt1 “or” stmt2` is true, *false* otherwise
- `(stmt1 && stmt2) → true` iff `stmt1 “and” stmt2` is true, *false* otherwise
- `(!stmt) → true` iff `stmt` is false ("not" true), *false* otherwise
Examples of Boolean Statements

int x = 4, y = 6;
string str1 = “hello”, str2 = “Hello”;

(x == y) → false  (x > y) → false  (x < y) → true
(x != y) → true  (x >= y) → false  (x <= y) → true
(x == 4) → true  (y > 10) → false  (y < x) → false
(x != 4) → false  (y >= 6) → true  (y <= x) → false
(str1 == str2) → false  (str1 == “hello”) → true
(str1 != str2) → true  (str2 != “Hello”) → false

(y > 10 || y < 100) → true
(y > 10 && y < 100) → false
(! (x == y)) → true

You can group statements e.g. (((y > 10) || (y < 100)))
The While-Loop

```
while (boolean statement) {  //← enter if boolean statement is true
    Perform Action(s)
    Update variables used in boolean statement
    //Go back to “top” of loop (evaluate boolean statement again)
}
//Rest of Program
```

![Flowchart of the While-Loop](image-url)
Example

//force user to enter positive number
int n;
cout << "Enter a positive number: ";
cin >> n;

while (n <= 0) {
    //test variable
    cout << "Positive number please: ";
cin >> n;  //update variable
}

cout << "Positive number please: ";
cin >> n;

Rest of Program
Updating Variables

```c
int i = 0;
i = i + 3;  //i = 3
i += 1;    //i = 4, same as i = i + 1;
i++;      //i = 5 (POST-INCREMENT)
++i;       //i = 6 (PRE-INCREMENT)
i = i - 2; //i = 4
i -= 4;    //i = 0, same as i = i - 4;
i--;       //i = -1 (POST-DECREMENT)
--i;       //i = -2 (PRE-DECREMENT)
i *= 2;    //i = -4, same as i = i * 2;
i /= 1;    //i = -4, same as i = i / 1;
```
Update in Loop Example

```cpp
int count_down = 10;
while (count_down >= 0) {
    // test variable
    cout << count_down << endl; // use variable
    --count_down; // update variable
}
cout << “We have liftoff!” << endl;
```

Output
10
9
8
7
6
5
4
3
2
1
0
We have liftoff!