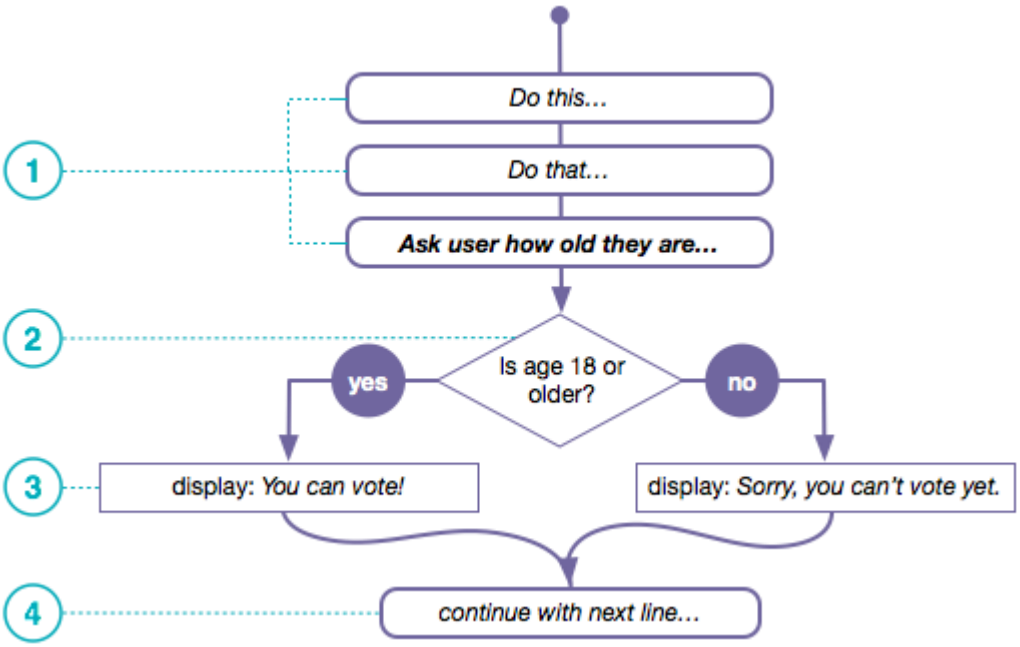
Set 27: If Statements

Skill 27.01: Interpret a program flow charts

Skill 27.01 Concepts

Programs are said to have a "flow of execution". You start by executing some line of code and then the next and so on. A flow chart is a common visual that's used to represent the various paths of execution that your program might take. Many people use them to help plan. Below is an example.



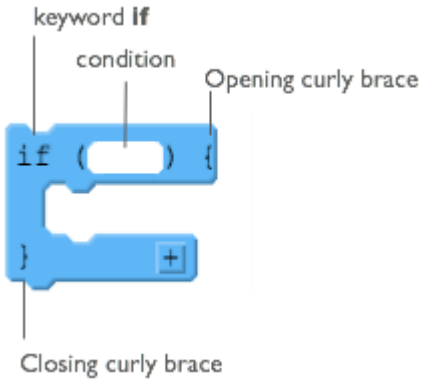
1. The flow chart above depicts a program executing one line after another until it gets to a point where it needs to make a decision.
2. In order to determine which path to take you state some **condition**. It should be a **Boolean expression** - something that evaluates to ***true*** or ***false***. Here we have a simple **comparsion** of two values: the person's age and the number 18.
3. The program does one thing if the condition is true, and something else if the condition is false.
4. The program can continue a single thread of execution after the condition as well.

Skill 27.01 Exercises 1 thru 3

Skill 27.02: Interpret If-Statement pseudocode

Skill 27.02 Concepts

if statements are the lines of code you need to change the flow while you're program is running. You can write code that *makes a decision that determines which lines of code should be run next.*



At the right is a diagram showing the elements of a basic *if* statement in JavaScript.

There are two basic parts to an if-statement.

1. A condition to be evaluated (A Boolean expression that evaluates to **true** or **false**)
2. Code that should run *if* the expression was true - enclosed in curly braces

Each row in the table below presents a small program that uses if-statements and robot commands. Trace the code and plot the movements of the robot for the 3 scenarios shown to the right of the code. If the robot is directed to move onto a black square, it "crashes" and the program ends. If the robot doesn't crash, then draw a triangle showing its ending location and direction.

There are a few patterns to the ways if-statements are typically used:

- Basic If-statements
- Sequential If-statements
- Nested If statements
- Combinations of all of the above

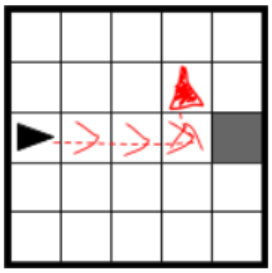
Each section below presents an example of one of these common patterns. For each type **study, and make sure you understand, the example** and why each of the 3 scenarios ends up in the state shown.

EXAMPLE: Basic If-statement

Code is executed sequentially from top to bottom. The code inside the if-block executes ONLY if the condition is true, otherwise the block is skipped and execution picks up on the first line after the if-block.

```
MOVE_FORWARD ()
IF (CAN_MOVE (forward))
{
   MOVE_FORWARD ()
   MOVE_FORWARD ()
}
ROTATE_LEFT ()
MOVE_FORWARD ()
```

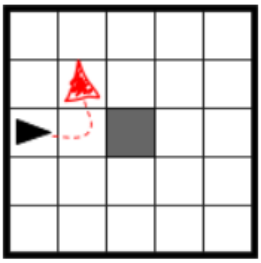
Scenario 1:



Use the diagram to trace each robot move.

Scenario 2:

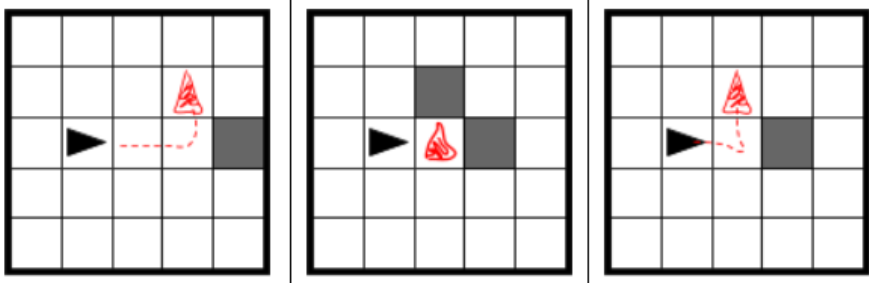
Scenario 3:



EXAMPLE: Sequential If-statements

Lines of code, including if statements, are evaluated separately, one at a time, in order from top to bottom. An if-block executes ONLY if the expression is true. Note that an earlier if-statement might change the state of the of world for an if-statement that comes later. This makes it hard to predict what will happen unless you trace the robot moves and take each line one at a time.

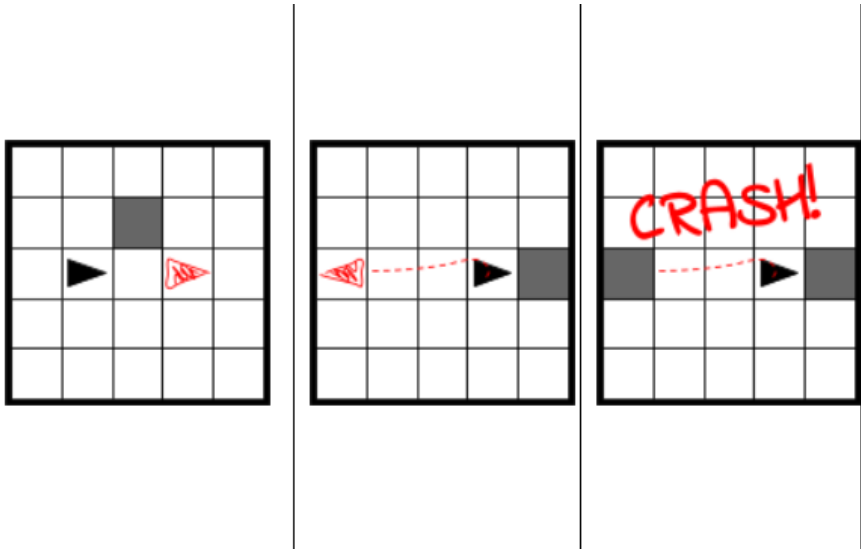
```
IF (CAN_MOVE (forward))
{
   MOVE_FORWARD ()
}
IF (CAN_MOVE (forward))
{
   MOVE_FORWARD ()
}
ROTATE_LEFT ()
IF (CAN_MOVE (forward))
{
   MOVE_FORWARD ()
}
```



EXAMPLE: Nested If-Statements

You can put if- and if-else statements inside other if-statements. All previous rules apply, but tracing the code can be tricky.

```
IF (CAN_MOVE (forward))
{
   MOVE_FORWARD ()
}
ELSE
{
    IF(CAN_MOVE( backward ))
    {
       ROTATE_LEFT ()
       ROTATE_LEFT ()
       MOVE_FORWARD ()
    }
    MOVE_FORWARD ()
}
MOVE_FORWARD ()
```



[Skill 27.02 Exercises 1 thru 3]

Skill 27.03: Write and If-Statement in JavaScript

Skill 27.03 Concepts

In javascript, if statements take the following form.

```
if(true){
    console.log(“This message will print!”);
}
//Prints “This message will print!”;
```

Notice in the example above, we have an if statement. The if statement is composed of:

- The if keyword followed by a set of parentheses () which is followed by a code block, or block statement, indicated by a set of curly braces {}.
- Inside the parentheses (), a condition is provided that evaluates to true or false.
- If the condition evaluates to true, the code inside the curly braces {} runs, or executes.
- If the condition evaluates to false, the block won't execute.

Skill 27.03 Exercise 1

Skill 27.04: Write a complex If-Statement

Skill 27.04 Concepts

If-Statements can include more than one boolean statement in the parentheses. For example, what if we wanted to evaluate the username and password of a potential user? We could write the following,

```
if(username == un && password == pw){
    console.log(“You’re in!”);
}
```

In fact any of the boolean expressions you experimented with in the previous lesson can be placed in between the parentheses following the if statement. For example,

```
var x = 79;
var y = 46;
var z = -3;
var w = 13.89;
var y = 40.0;

if(y/2 > w && w != x){
     console.log(“True”);
}
```

Skill 27.04 Exercises 1