Lego Robotics Camp

Day 2: Conditionals and Loops
Review

• Yesterday we:
  • Engaged with the four pillars of computational thinking
  • Designed algorithms for simple tasks (PB&J, robot dancing)
  • Got acquainted with our robots and programming environment
  • Learned about variables (names for specific objects in our programs)
Today’s Plan

• Today we will:
  • Learn about **conditionals** and **loops**
  • Design simple **algorithms** for parking our robots
Simon Jeannie Says...
Conditional Statements

• **If** (you were born in the winter) then (touch your head)
  **Else** (clap your hands)

• **If** (you play the trumpet) then (jump up and down 3 times)
  **Else** (spin in a circle)

• **If** (your favorite ice cream flavor is chocolate) then (clap your hands)
  **Else if** (your favorite ice cream flavor is vanilla) then (touch your toes)
  **Else** (sit on the floor)

• **If** (some condition is true) then (do some action)
  **Else if** (some other condition is true) then (do some other action)
  **Else** (do some other different action)
Conditional Statements

- True and false are called **boolean** values
- If-else (or **conditional**) statements require checking if a **boolean condition** is true or false and responding appropriately
- In Python:

```python
if condition1 is true:
    do action 1
elif condition2 is true:
    do action 2
else:
    do action 3
```
Python example

```python
a = 200
b = 33
if b > a:
    print("b is greater than a")
elif a > b:
    print("a is greater than b")
else:
    print("a is equal to b")
```
Python example

```python
a = 200
b = 33
if b > a:
    print("b is greater than a")
elif a > b:
    print("a is greater than b")
else:
    print("a is equal to b")
```

```python
if condition1 is true:
    do action 1
elif condition2 is true:
    do action 2
else:
    do action 3
```
Python example

```python
a = 200
b = 33
if b > a:
    print("b is greater than a")
elif a > b:
    print("a is greater than b")
else:
    print("a is equal to b")
```

if condition1 is true:
    do action 1
elif condition2 is true:
    do action 2
else:
    do action 3

condition1: is b > a? False.
Python example

```python
a = 200
b = 33
if b > a:
    print("b is greater than a")
elif a > b:
    print("a is greater than b")
else:
    print("a is equal to b")
```

if condition1 is true:
do action 1
elif condition2 is true:
do action 2
else:
do action 3

condition2: is a > b?
Python example

```python
a = 200
b = 33
if b > a:
    print("b is greater than a")
elif a > b:
    print("a is greater than b")
else:
    print("a is equal to b")
```

if condition1 is true:
    do action 1
elif condition2 is true:
    do action 2
else:
    do action 3

condition2: is a > b? True. In this case, since it is true, we would print that a is greater than b.
Python example

```python
a = 33
b = 33
if b > a:
    print("b is greater than a")
elif a > b:
    print("a is greater than b")
else:
    print("a is equal to b")
```

Suppose condition1 and condition2 are both false. Then we end up here.

```python
if condition1 is true:
    do action 1
elif condition2 is true:
    do action 2
else:
    do action 3
```
Python example

```python
a = 33
b = 33
if b > a:
    print("b is greater than a")
elif a > b:
    print("a is greater than b")
else:
    print("a is equal to b")
```

Note that punctuation and indentation matter A LOT!
Square Dancing

• Suppose we want to design a simple algorithm for walking in a perfect square around the room
  • Start and end at same spot
  • Remember pillars of computational thinking
    (decomposition, pattern recognition, abstraction, algorithms)
Square Dancing

- Suppose we want to design a simple algorithm for walking in a perfect square around the room
  - Walk forward 10 steps
  - Turn left 90 degrees
  - Walk forward 10 steps
  - Turn left 90 degrees
  - Walk forward 10 steps
  - Turn left 90 degrees
  - Walk forward 10 steps
  - Turn left 90 degrees
Square Dancing

• Suppose we want to design a simple algorithm for walking in a perfect square around the room
  • Walk forward 10 steps
  • Turn left 90 degrees
  • Walk forward 10 steps
  • Turn left 90 degrees
  • Walk forward 10 steps
  • Turn left 90 degrees
  • Walk forward 10 steps
  • Turn left 90 degrees

How can this be simplified?
Square Dancing

• Suppose we want to design a simple algorithm for walking in a perfect square around the room
  • Walk forward 10 steps
  • Turn left 90 degrees
  • Walk forward 10 steps
  • Turn left 90 degrees
  • Walk forward 10 steps
  • Turn left 90 degrees
  • Walk forward 10 steps
  • Turn left 90 degrees

How can this be simplified?
Which lines are repeated?
Square Dancing

• Suppose we want to design a simple algorithm for walking in a perfect square around the room
  • Walk forward 10 steps
  • Turn left 90 degrees
  • Walk forward 10 steps
  • Turn left 90 degrees
  • Walk forward 10 steps
  • Turn left 90 degrees
  • Walk forward 10 steps
  • Turn left 90 degrees

How can this be simplified?
Which lines are repeated?
How many times are they repeated?
Suppose we want to design a simple algorithm for walking in a perfect square around the room

- Repeat 4 times:
  - Walk forward 10 steps
  - Turn left 90 degrees

This is called a loop

- Simplifies and shortens repeated code
robot.straight(500)
robot.turn(90)
robot.straight(500)
robot.turn(90)
robot.straight(500)
robot.turn(90)
robot.straight(500)
robot.turn(90)
Robot Example

```python
num_turns = 0;
while num_turns < 4:
    robot.straight(500)
    robot.turn(90)
    num_turns = num_turns + 1
```

robot.straight(500)
robot.turn(90)
robot.straight(500)
robot.turn(90)
robot.straight(500)
robot.turn(90)
robot.straight(500)
robot.turn(90)
Loops

- While condition is true:
  - Do some action repeatedly
- If condition never becomes false, the loop will go on forever!
- This is called an **infinite loop**
- In Python, loops look like:

```python
while condition1 is true:
    do action1
```

Will repeat until the condition becomes false!
Python example

```python
a = 1
while a < 6:
    print(a)
    a = a + 1
```

while `condition1` is true:
do action1

Print the value of `a` while `a` is less than 6. What will this print?
a = 1
while a < 6:
    print(a)
a = a + 1

Print the value of \texttt{a} while \texttt{a} is less than 6. What will this print?
1
2
3
4
5
Python example

```python
a = 1
while True:
    print(a)
    a = a + 1
```

What will this print?

while *condition1* is true:
    do *action1*
Python example

```
a = 1
while True:
    print(a)
a = a + 1
```

What will this print?

1 2 3 4 5 6

... it will go on forever!

while condition1 is true:
do action1
a = 1
b = 5
while a < b:
    print(a)
    a = a + 1

What will this print?
Python Quiz

a = 1
b = 5
while a < b:
    print(a)
    a = a + 1

What will this print?
1
2
3
4
a = 1
b = 5
c = 3

while a < b:
    print(a)
    if a == c:
        print(“we are here!”)
        print(“Is a equal to c?”)
    a = a + 1

What will this print?
Hard Python Quiz

a = 1
b = 5
c = 3
while a < b:
    print(a)
    if a == c:
        print("we are here!")
    a = a + 1

What will this print?
1
2
3
we are here
4
Summary

• Conditionals and loops allow us to solve much more interesting problems with our robots
• Tomorrow we’ll look at some examples
BREAK
Parking Algorithms

- Yesterday we made our robots move
- Today we’ll examine algorithms for parking your robots
- We won’t need conditionals or loops (yet)
- The goal of today’s lab is to gain more experience with moving and turning our robots

- What are common parking scenarios?
Parking Algorithms

• Yesterday we made our robots move
• Today we’ll examine algorithms for parking your robots
• We won’t need conditionals or loops (yet)
• The goal of today’s lab is to gain more experience with moving and turning our robots

• What are common parking scenarios?
  • Perpendicular parking
  • Parallel parking
Perpendicular Parking

- Bad examples:
Perpendicular Parking

- Park in spaces that are perpendicular (90 degrees) away from your car’s straight line motion
Think Pair Share

• Work with a partner to develop your own algorithm for perpendicular parking your robots!
• You aren’t writing actual code (yet!)
• You are thinking about the logical steps
• Example:
  • Move forward 500 cm
  • Turn 90 degrees clockwise
  • Move backward 500 cm
  • Turn 60 degrees counter-clockwise
Challenges

• Why was this hard?
• What information did you need to write this algorithm?
• Suppose we want to make our robots autonomous (self-driving). How would this work for your parking algorithm?
Parallel Parking

• First, let’s look at an example of a bad algorithm
Parallel Parking

- Here’s a good algorithm
Parallel Parking
Parallel Parking

- One more using EV3 robots
Think Pair Share

• Work with a partner to develop your own algorithm for parallel parking your robots
• You still aren’t writing actual code (yet!)
• You are thinking about the logical steps
• Example:
  • Move forward 500 cm
  • Turn 90 degrees clockwise
  • Move backward 500 cm
  • Turn 60 degrees counter-clockwise
Self-Driving Cars

• Let’s extend this idea a bit and think about self-driving cars
• What decisions do cars have to make when parking?
• What other decisions do cars make when driving?
• How do self-driving cars work?
Lab

- For lab today, you will write code for parking your robots
- Start with perpendicular, then try parallel
- You can create your own practice course
- But you have to pass my test to get your license!
- Think about what it would take to make your robot autonomous with respect to parking.
LUNCH BREAK