

# Code in Place 2024

Stanford CS106A

Section - Week 4

Python Control Flow

David Tsai, Code in Place

# Today's Agenda



## 1. Check-In

How are we all doing?



## 2. Concepts Review

Control Flow



## 3. Practice Problem

"High Low Game"



## Check-In

How is everyone doing?  
Hopefully your third week of CIP went well!

- Any questions?

# Concepts Review

# Control Flow



- Everything from Karel is still in play!
- for loops
- while loops
- booleans (True or False, used in conditional statements)
- if, else, elif

```
user_number = int(input("Enter a number: "))    # Variable assignment
if user_number == 0:                            # Equality comparison
    print("Your number is 0!")
elif user_number > 0:
    print("Your number is positive!")
else:
    print("Your number is negative!")
```

# Expressions & Arithmetic Operators



- Any **expression** on the right side of the equal sign is calculated before being assigned to a variable.

- Precedence of operations** (similar to “PEMDAS” in Algebra):
  - `()` “parentheses” **highest precedence**
  - `**` “exponentiation”
  - `-` “negation”
  - `*`, `/`, `//`, `%`
  - `+`, `-` **lowest precedence**



- Example:

```
x = (1 + 3 * 5 / 2) * (-3)    # Python will calculate the right-side expression.
x = (1 + 15 / 2) * (-3)
x = (1 + 7.5) * (-3)
x = (8.5) * (-3)
x = -25.5                    # float -25.5 assigned to “x”.
```

# Comparison Operators



Operator	Meaning	Example	Result
<code>==</code>	equals	<code>1 + 1 == 2</code>	True
<code>!=</code>	does not equal	<code>3.2 != 2.5</code>	True
<code>&lt;</code>	less than	<code>10 &lt; 5</code>	False
<code>&gt;</code>	greater than	<code>10 &gt; 5</code>	True
<code>&lt;=</code>	less than or equal to	<code>126 &lt;= 100</code>	False
<code>&gt;=</code>	greater than or equal to	<code>5.0 &gt;= 5.0</code>	True



WARNING: Notice the difference between variable **assignment** vs **equality** comparison.

- `x = 64`      # Assigns the int value 64 to the variable x.
- `x == 64`    # Checks if the value of x is equal to 64.

# Logical Operators

and

X	Y	X and Y
True	False	False
False	True	False
True	True	True
False	False	False

**If all variables are True, the outcome is True.**

or

X	Y	X or Y
True	False	True
False	True	True
True	True	True
False	False	False

**If one variable is True, the outcome is True.**

not

X	not X
True	False
False	True

**Reverse a variable's logical state.**



# Logical Operators: Examples



- **and**

```
if temperature > 0 and temperature < 30:  
    print("The temperature is good")  
else:  
    print("The temperature is bad")
```

- **or**

```
if temperature <= 0 or temperature >= 30:  
    print("The temperature is bad")  
else:  
    print("The temperature is good")
```

- **not**

```
is_sunny = True  
  
if is_sunny:  
    print("It is sunny outside!")  
if not is_sunny:  
    print("It is not sunny outside!")
```

# A Classic Joke: "The Mathematician's Answer"



**Q:** Would you like an apple or a banana?

**A:** Yes

# Boolean Expressions



- **Precedence of operations:**

- arithmetic  $5 * 7$
- comparison  $>=$
- not
- and/or

**highest precedence**



**lowest precedence**

- Example:

```
25 >= 3 + 2 * 10 and not False
25 >= 23 and not False
True and not False
True and True
True
```

```
# arithmetic
# comparison
# not
# and
```



- The following example can produce an infinite loop.

```
while True:
    body
```

# Boolean Variables



You can store expressions that evaluate to True/False into variables.

- The variables `x`, `y`, and `z` are assigned data of type `bool`:

- `x = 1 < 2` # True
- `y = "Michael" == "Michelle"` # False
- `z = True`
- You can use boolean variables and chain them together.

`if x and y:`  
*body*



`if True and False:`  
*body*



`if False:`  
*body*

# Bonus: f-strings



- **f-strings** are “formatted string literals” (introduced in Python 3.6)
- You create a f-string by prefixing the character “f” in front of a string.
- Inside an f-string, you can use variables inside curly braces { }

- Example:

```
name = "Jelani"  
age = 42  
print(f>Your name is {name} and your age is {age}")
```

- VS:

```
print("Your name is " + name + " and your age is " + str(age))
```



# Section Exercise: “High Low Game”

```
Welcome to the High-Low Game
-----
Round 1
Your number is 8
Do you think your number is higher or lower than the computer's?: lower
You were right! The computer's number was 35
Your score is now 1

Round 2
Your number is 88
Do you think your number is higher or lower than the computer's?: higher
Aww, that's incorrect. The computer's number was 100
Your score is now 1

Round 3
Your number is 63
Do you think your number is higher or lower than the computer's?: higher
You were right! The computer's number was 5
Your score is now 2

Thanks for playing!
```



## Game Steps:

- #1)** Randomly generate two numbers from 1 to 100 (inclusive):
  - One number for you, and one number for the computer
- #2)** Game prints your number but not the computer's. You make a guess:
  - Type “**lower**” if you think your number is lower than the computer's
  - Type “**higher**” if you think your number is higher
- #3)** If you guess correctly, you score 1 point!

# “High Low Game” Milestones



**Milestone #1:** Generate two random numbers.

**Milestone #2:** Get the player's choice of `lower` or `higher`.

**Milestone #3:** Write the game logic. Compare player's guess vs actual #.

**Milestone #4:** Play multiple rounds.

**Milestone #5:** Add a points system.

**Extension #1:** Safeguard user input.

**Extension #2:** Conditional ending messages.