print("Hello, World!")

NCSS Challenge - Beginners
Week 2 Part 1

GROK LEARNING

What will we cover?

- Making decisions;
- Decisions with two options;
- Decisions about numbers;
- Complex decisions.

What does this cover?

- Define simple problems, and **describe and follow a sequence of steps and decisions** (algorithms) needed to solve them (ACTDIP010)
- Implement simple digital solutions as visual programs with algorithms involving **branching (decisions)** and **user input** (ACTDIP011)
- Design algorithms represented diagrammatically and in English, and trace algorithms to **predict output for a given input** and to identify errors (ACTDIP029)

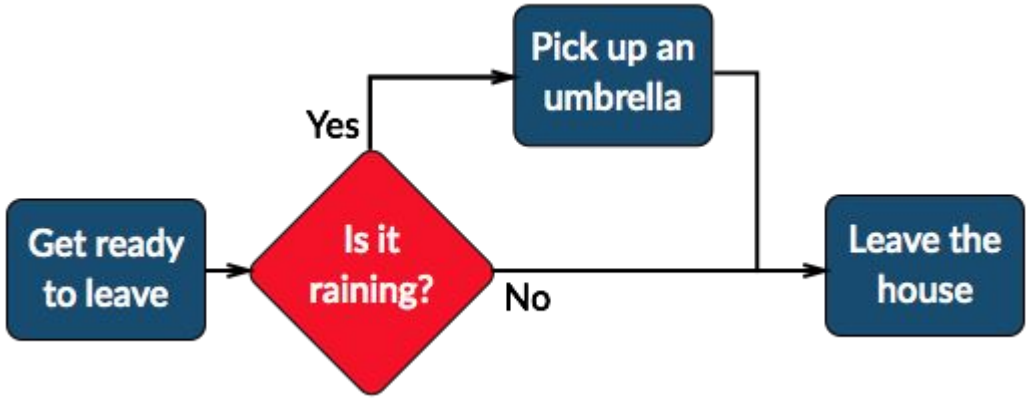
GROK LEARNING

1 Making Decisions with Code

Hello, World!

GROK LEARNING

Programs without decisions get a bit boring!



GROK LEARNING

Programs without decisions get a bit boring!

GROK LEARNING

`if` statements determine `True` and `False`

- `if` statements work out if a condition is `True` or `False`. If the condition is `True` then the block controlled by the `if` statement will be run.

```
raining = input('Is it raining? ')
if raining == 'yes':
  print('Pick up an umbrella.')
print('Leave the house.')
```

Indentation woes!

- Indentation makes a big difference in Python. It controls how a program flows.
- Encourage students to always use a consistent level of indentation - two spaces, for instance.

Testing equivalence with ==

- *Assign* a value to a variable using =

```
flavour = 'mango'
```

Think: Let `flavour` *equal* `'mango'`

- *Test* whether a value equals another value using ==

```
if flavour == 'mango':
```

Think: Is `flavour` *equivalent to* `'mango'`

Equals and Equivalent Errors!

- It’s very easy to get = and == mixed up!

Doing multiple things in an `if` statement block:

```
food = input('What food do you like? ')
if food == 'cake':
  print('Wow, I love cake too!')
  print('Did I tell you I like cake?')
```

Indentation matters!

- The lines of code that are indented under the `if` statement are run *only if* that condition is true
- If one line is indented with two spaces, and the next line with three, Python will raise an *`IndentationError: unexpected indent`*

Doing multiple things in an `if` statement block:

- You can use a `print` statement to test conditional expressions:

```
name = "Nicky"
print(name == "Nicky")
      → True
```

Pedagogical Philosophy - Interpreter to the rescue!

- You can use the interpreter to ‘inspect’ whether a conditional expression evaluates to `True` or `False`:

```
>>> name = "Nicky"
>>> name == "Nicky"
True
>>> name == "Sam"
False
>>>
```

GROK LEARNING

Test it out!

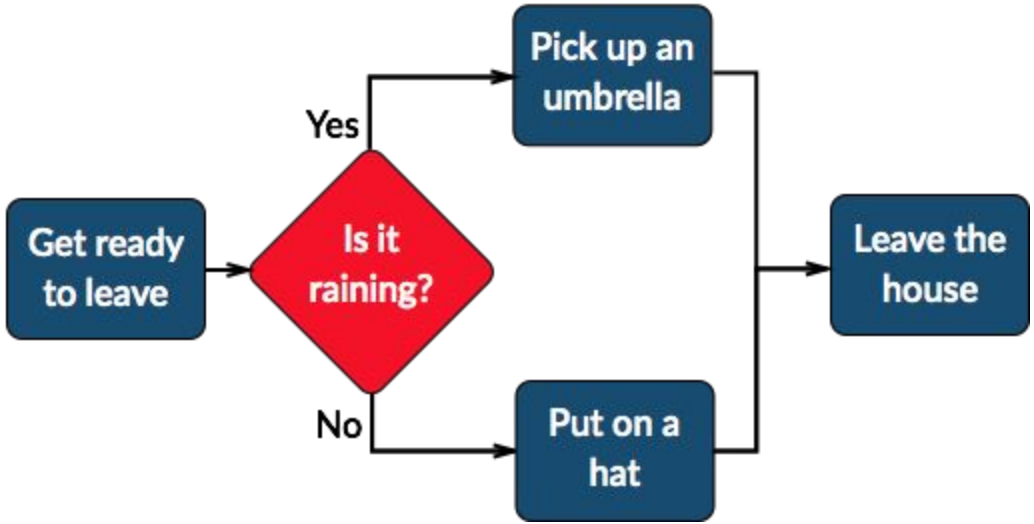
Try the first question now!

GROK LEARNING

2 Decisions with two options

GROK LEARNING

Decisions with two options



GROK LEARNING

Decisions with two options: the `else` keyword

```
raining = input('Is it raining? ')
if raining == 'yes':
  print('Pick up an umbrella.')
else:
  print('Put on a hat.')
```

Teacher Aside: The case of the forgotten colon.

- Watch out for those colons! Students often forget the colon at the end of a condition. One easy way for students to notice this is if automatic indentation doesn't seem to be working correctly!

Test it out!

Try the second question now!

GROK LEARNING

3 Comparing Things

GROK LEARNING

Comparison operators for use in `if` statements

Operation	Operator
equal to	==
not equal to	!=
less than	<
less than or equal to	<=
greater than	>
greater than or equal to	>=

Comparison Operators, bigger, smaller, equal or not!

- Now that we have more comparison operators, we can make decisions on lots more things!

```
x = 5
print(x <= 10)
    → True
```

Teacher Aside! Gotchas with comparisons:

- Comparing things that aren't the same type!

```
>>> '10' < 10
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
TypeError: unorderable types: str() < int()
>>> '10' != 10
True
>>> '10' == 10
False
>>>
```

Making decisions with numbers

- We can use the conditional operators to make decisions based on numerical input:

```
height = int(input('How tall in cm? '))
if height > 120:
  print("Tall enough for this ride.")
else:
  print("Not tall enough for this ride.")
```

Test it out!

Try the third question now!

GROK LEARNING

3 Nested decisions

GROK LEARNING

Making decisions in decisions!

- The body of an `if` statement may contain another `if` statement. This is called *nesting*.

```
x = int(input('Enter a number: '))
if x <= 3:
  print('It is less than or equal to three')
  if x >= 3:
    print('It is greater than or equal to
three')
```

Making decisions between multiple options

- You could nest decisions inside one another...

```
x = int(input('Enter a number: '))
if x < 3:
  print('x is less than three')
else:
  if x == 3:
    print('x is equal to three')
  else:
    print('x is greater than three')
```

Elif often makes for a much nicer option

```
x = int(input('Enter a number: '))
if x < 3:
  print('x is less than three')
elif x == 3:
  print('x is equal to three')
else:
  print('x is greater than three')
```

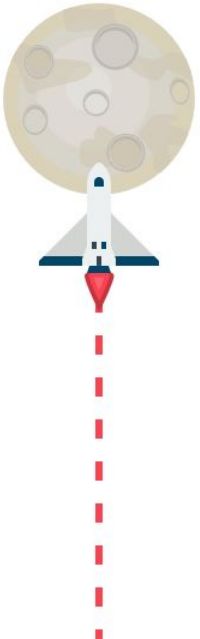
Test it out!

Try the third question now!

GROK LEARNING

Teacher Aside!

- Nested if statements are a tricky one to grasp. We don’t test them here - only introduce them as a concept. We’ll keep working up to them!



Any Questions?

Find me at:
@groklearning
nicky@groklearning.com

GROK LEARNING