If statements (class slides)

CSc 110 If statements

The if statement

- If statements can be used to **run code conditionally**
  - **Before if-statements:** Code has pretty much just run in a straight line
  - **With ifs:** Can run code optionally, depending on the value of a condition

This means our code can **branch** in different directions

The if statement

The condition is an expression that is evaluated to a bool.

Example:

```
def greeting(name):
  if name == "Bond":
    return "Welcome on board 007."
  else:
    return "Hello, " + name

def main():
  user_name = input("Enter your name:\n")
  print( greeting(user_name) )

main()
```

Improve the function

Instead of checking if the name entered is “Bond”, also check whether the name is “James Bond”.

Conditional execution

The computer program branches out, or makes decisions

```python
def greater_than_zero(n):
  if n > 0:
    return "Greater than zero"
  else:
    return "Not greater than zero"

def main():
  result = greater_than_zero(4)
  print(result)

  result = greater_than_zero(0)
  print(result)

  result = greater_than_zero(-3)
  print(result)

main()
```

```
Greater than zero
Not greater than zero
Not greater than zero
```

Write a function

Write a Python function that does the following:

1. Its name is `absolute`
2. It takes one numeric argument (integer or float) `n`
3. It returns the absolute value of `n`: if `n` is positive, it results `n`, if `n` is negative, it returns `n * -1`

Test cases:

```
print( absolute(4) ) # 4
print( absolute(-4) ) # 4
print( absolute(0) ) # 0
```

Write a function

```
def absolute(n):
  if n > 0:
    return n
  else:
    return -n

def main():
  print( absolute(4) )
  print( absolute(-4) )
  print( absolute(0) )

main()
```

```
4
4
0
```

Write a function

1. Function name is `age_milestones` and it takes one integer argument: `age`
2. It returns:
    - 'You may apply to join the military' if `age` is greater or equal to `18`
    - 'You may drink' if `age` is greater or equal to `21`
    - 'You may run for president' if `age` is greater or equal to `35`

```
print( age_milestones(18) ) # You may apply to join the military.
print( age_milestones(30) ) # You may apply to join the military. You may drink.
print( age_milestones(0) ) #
```

Age milestones

```
def age_milestones(age):
  '''
  This function prints an informative message based on,
  a person's age.
  Args:
    age: integer representing a person's age
  Returns:
    A string with a message to the user
  '''
  message = ""
  if age >= 18:
      message += 'You may apply to join the military.'

  if age >= 21:
      message += ' You may drink.'

  if age > 35:
      message += ' You may run for president.'

  return message

def main():
  print( age_milestones(18) ) # You may apply to join the military.
  print( age_milestones(30) ) # You may apply to join the military. You may drink.
  print( age_milestones(0) ) #

main()
```

```
You may apply to join the military.
You may apply to join the military. You may drink.
```

Input validation

String methods

In addition to having built-in **functions** (`len()`, `print()`, `int()`, `float()`, etc.), Python also has a number of **methods** we will be using in this class.

Check the documentation for string methods and read what `.isnumeric()` does.

Validating numbers

- The `input()` function always returns a `string`
- We can use the string built-in method `.isnumeric()` to determine if a string represents a number
- The idea is to ensure the input string only contains digits

Try these out:

```
name = "Jimmy"
name.isnumeric()
```

```
False
```

```
name = "42"
name.isnumeric()
```

```
True
```

```
age = 37
age.isnumeric() # this throws an error
```

Write a validation function

Write a Python function that does the following:

1. Its name is `validate_age`
2. It takes a string argument: `age`
3. It returns `True` if `age` contains only 0-9 digit characters, and `False` otherwise

Call this validation in your previous code for age milestones.

Age milestones

```
def age_milestones(age):
  '''
  This function prints an informative message based on,
  a person's age.
  Args:
    age: integer representing a person's age
  Returns:
    A string with a message to the user
  '''
  message = ""
  if age >= 18:
      message += 'You may apply to join the military'

  if age >= 21:
      message += 'You may drink'

  if age > 35:
      message += 'You may run for president'

  return message

def validate_age(age):
  return age.isnumeric()

def main():
  '''
  This functions takes input from the user and calls the
  check_age() functiont to print a message
  '''
  age = input('How old are you?\n')
  if validate_age(age):
    print(age_milestones(age))
  else:
    print("Invalid age entered")

main()
```

Quiz 04

You have 10 minutes to complete the quiz

- No need for comments
- No need for a `main()`
- No need to print test cases
- Just write your function and what's inside the function

Built-in functions you can use: `round()`, `input()`, `float()`, `str()`, `int()` — you don't have to use these at all