while loop intro (class slides)

CSc 110 - Intro to While Loops Adriana Picoral

While loops

- A **while** loop allows a programmer to repeat code
- You can think of it as an if-statement with the potential to repeat

```
statements . . .

while conditionA:
    statementA
    statementB
    . . .
    statementN

statements . . .
```

What will happen?

```
number = 15
while number < 50:
    print('number is less than 50')
```

While loops

- What if the condition never evaluates to `False`?
  - Infinite loop!
- There are two ways around this:

- Break (do not use in this class!)
- Designing the code such that the condition will eventually become `False`

What will happen?

To ensure our condition (`number < 50`) will eventually be evaluated as `False`, we need to updated `number` inside our loop:

```
number = 15
while number < 50:
    print('number is less than 50')
    number += 1
```

While loops – visualization

Go to Python Tutor to visualize how the `while` loop runs.

While loop – example

```
def multiply(n, m):
  result = 0
  count = 0
  while count < m:
    result += n
    count += 1
  return result

def main():
  print( multiply(2, 3) )
  print( multiply(5, 2) )

main()
```

```
6
10
```

Write a function

1. Write a function called `add_up_to` that takes an numeric argument `n`.
2. The function should add all numbers from 1 to `n` in a while loop
3. Return sum

```
print( add_up_to(5) ) # 15
print( add_up_to(10) ) # 55
```

Write a function – solution

```
def add_up_to(n):
  sum = 0
  current_number = 0
  while current_number <= n:
    sum += current_number
    current_number += 1
  return sum

def main():
  print( add_up_to(5) )
  print( add_up_to(10) )

main()
```

```
15
55
```

Age milestones

Modify the code below to use a `while` loop to request a valid input from the user.

```
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):
    age = int(age)
    print(age_milestones(age))
  else:
    print("Invalid age entered")

main()
```

Age milestones – solution

Modify the code below to use a `while` loop to request a valid input from the user.

```
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')
  while validate_age(age) == False:
    print("Invalid age entered. Please enter a valid age.")
    age = input('How old are you?\n')
  age = int(age)
  print(age_milestones(age))

main()
```