CMPT 120

Topic: Iterative Statements – Part 2
-> while loop
Last Lecture

- Iterative Statements
  - for loop
  - range() built-in function
  - in operator
Learning outcomes

At the end of this course, a student is expected to:

• Create (design), analyze, and explain the behaviour of simple algorithms:
  • ...  
  • Create **while** loops with compound conditions
  • Design algorithms that **avoid** the use of **break** statements to interrupt **loops** or to **avoid** the use of **multiple return** statements to exit a function

• Create (design) small to medium size programs using Python:
  • ...  
  • Refactor repeated statements into for and **while** loops
Today’s Menu

• Iterative statements
  • while loop
Back to our Guessing Game

- What if we wanted to allow the user (i.e., player) to guess (i.e., play our game) as long as s/he wished?
Syntax of a while loop

<stmt: initialize condition variable>

while <Boolean condition> :
    <first statement to be repeated>
    <second statement to be repeated>
    ...
    <stmt: modify condition variable>
<statement outside (after) the loop>
Syntax of a while loop

```plaintext
<stmt: initialize condition variable>
while <Boolean condition> :
    <first statement to be repeated>
    <second statement to be repeated>
    ...
    <stmt: modify condition variable>
<statement outside (after) the loop>
```

- **Important** – About Indentation
  - *Statements inside the loop* (i.e., statements executed at each iteration of the loop) are the statements indented with respect to the `while` keyword
  - *Statements outside the loop* (before and after the loop) are the statements that are **not** indented with respect to the `while` keyword – these statements are considered to be at the same level of indentation as the `while` loop
Back to our Guessing Game

• What if we wanted to allow the user (i.e., player) to guess (i.e., play our game) as long as s/he wished?

• Solution: GuessingGame_6.py

• Step 5 – Testing
  • How would we test this program?
How does a **while** loop work?

| Iteration # | playing (loop condition) |
Difference between **while** and **for** loops

```python
<stmt: initialize condition variable>
while <Boolean condition> :
    <first statement to be repeated>
    <second statement to be repeated>
    ...
    <stmt: modify condition variable>
<statement outside (after) the loop>
```

```python
<statement outside (before) the loop>
for <iterating variable> in <sequence> :
    <first statement to be repeated>
    <second statement to be repeated>
    ...
    <last statement to be repeated>
<statement outside (after) the loop>
```
When best to use a \texttt{while} loop

- If we know there is a condition that will occur in our program and its occurrence will dictate when we stop repeating a set of statements, then we use the iterative statement called \texttt{while} loop.
- That condition is often called a \texttt{sentinel} or \texttt{flag}.
  - Examples:
    - User termination
      - User enters yes/no (\texttt{flag}) or some special value (\texttt{sentinel})
      - User selects ‘X’ to eXit from a menu (menu-driven program)
    - Occurrence of an error
    - Reading data from a file -> EOF
Here is a Python code fragment with a loop!

```
fruit = ["banana", "apple", "plum"]
index = 0
while index < len(fruit):
    print(fruit[index])
    index = index + 1
```

Any observations?
Let’s challenge ourselves!

Let’s convert the previous \texttt{while} loop using the most appropriate iterative statement?
GPS: Not to be used in this course!

why?

```python
while True :
    while 1 :
        break
        continue
        pass
```

Why?
Summary

- Iterative statements – while loop
Next Lecture

- Commonly used algorithms using loops