CSC 223 - Advanced Scientific Programming

Python Control Flow

Control Flow

- Control flow is the order in which statements are evaluated.
- There are two main ways to alter sequential control flow:
  - Selection: conditional statements
  - Iteration: loop statements

Conditional Statements

- Conditional statements select blocks of code to execute based on some Boolean condition.

```
x = 42
if x == 0:
    print(x, "is zero")
elif x > 0:
  print(x, "is positive")
elif x < 0:
  print(x, "is negative")
else:
  print("this should not happen")
```

`for` loops

- Loops are a way to repeatedly execute a block of code
- The Python `for` loop is for iterating through a sequence:

```
for N in [2, 3, 5, 7]:
    print(N, end=' ')
```

- The `range` object generates a sequence of numbers

```
for i in range(10):
    print(i)
```

- The arguments to `range` are integers (*start*, *stop*, *step*) where stop is exclusive and the start and step are optional.

`while` loops

- A `while` loop iterates until some condition is met

```
i = 0
while i < 10:
    print(i, end=' ')
    i += 1
```

- A `while` loop is executed until the Boolean expression evaluates to False.

break and continue

- There are two statements that can alter how loops are executed:
  - The `break` statement breaks out of the loop entirely
  - The `continue` statement skips the remainder of the current iteration

```
for n in range(20):
    if n == 10:
        break # exit the loop if n equals 10
    if n % 2 == 0:
        continue # skip the rest of the loop
    print(n, end=' ')
```

Loops with an `else` Block

- Python allows a loop to have an `else` statement which is executed if the loop does not encounter a `break` statement.

```
L = []
nmax = 30

for n in range(2, nmax):
    for factor in L:
        if n % factor == 0:
            break
    else: # no break
        L.append(n)
```