Loops

Repeated steps
Increment and decrement

\[
\begin{align*}
x &= x + 1 & \quad & x &= x - 1 \\
x &= x + 1 & \quad & x &= x - 1 \\
a &= a \times 2 & \quad & a &= a \times 2 \\
a &= a / 10 & \quad & a &= a / 10 \\
a &= a \mod 3 & \quad & a &= a \mod 3 \\
a &= a ^ 2 & \quad & a &= a ^ 2 
\end{align*}
\]
Indefinite loops (while)

```python
a = 0  # iteration variable
while a < 5:
    print('Hello')
    a += 1
```
Infinite loops

```python
a = 0
while True:
    print('Hello')
    a += 1
    if a == 5:
        break
```
Boolean expressions

• Constants
  • True
  • False

• Expressions
  • 1 < 2
  • a > 0

• Class 'bool'
  >>> type(True)
  <class 'bool'>
  >>> type(1+2==3)
  <class 'bool'>
Statement **continue**

```python
a = 0
while a < 10:
    a += 1
    if a % 2 == 0:
        continue
print(a)
```
Changing iteration variable by more than by 1

```python
a = 1
while a < 10:
    print(a)
    a += 2
```
Definite loops (for)

for variable in list:
    statement
    statement

for a in [1, 2, 3, 4, 5]:  # a is iteration variable
    print('Hello')

students = ['Joseph', 'Glenn', 'Sally']
for student in students:
    print('Good morning,', student)
print('Done!')
Constant value None

>>> type(None)
<class 'NoneType'>

• Variable doesn’t have a value
largest_so_far = None

• Function doesn’t return a value
def printA():
    print('A')

print(printA())
Operator `is`

- can be used in logical expressions
- "is the same as"
- similar to, but stronger than `==`
- it is better always to use `==`
  ```python
  >>> 4 == 4.0
  True
  >>> 4 is 4.0
  False
  ```
- `is not` is similar to `!=`
Reserved words

while  True
break  False
continue  None
for  is
in
Homework (part of the multiplication table, number of days in months)

Questions?
Test 4

Questions?
Exercises

https://courses.cs.ut.ee/2018/nkp/fall/Main/During4