Martin Flores

HW 3

3.1 Python data types

a) numeric type (float)
b) numeric type (int)
c) Boolean value (bool)
d) text sequence type (str)
e) numeric and text sequence (list of str and int)
f) numeric and text sequence (str and int)
g) None-data type
h) Dictionary

3.2 Python Lists and Strings

a) `bag[1:3]`

b)

- **i)** reverses the list order
- **ii)** `bag[-2:-4:-1]`

c)

- **i)** `ga[0:4]`
- **ii)** `ga[15:20]`

d)

- **i)** replaces first element in list (‘guide’) with ‘book’
- **ii)**

```
In [27]: bag
Out[27]: ['book', 'towel', 'tea', 'mice']

In [28]: mybag
Out[28]: ['book', 'towel', 'tea', 'mice']

In [29]: yourbag
Out[29]: ['book', 'towel', 'tea', 42, 'money']
```

- **iii)** `x = a` ties two variables together by assigning them to the same data set.
  `y = a[:]` copies the content of a variable onto a new variable.

e)

- **i)** `TypeError: 'str' object does not support item assignment`
- **ii)**

```
In [38]: l = "three"

In [39]: g = ga[4:]

In [40]: l + g
Out[40]: 'three score and seven years ago'
```

f) The first command (`ga.split()`) takes the string and converts it into a list by making each word a separate element in the list. `a, b, c = ga.split()[:3]` takes the new list and assigns the first three elements to a, b and c respectively. `list([1,2,3])` creates a list with integers 1, 2 and 3 as the elements. `list(ga)` takes the string `ga` and turns it into a list by making each character, including spaces, a separate element.

g)

i) `bags[0]`
ii) `bags[0][1]`
iii) `bags[1][2]`

3.3 Loops (14 points)

a) code:
```
sentence = ["We", "must", "walk", "before", "we", "can", "run"]
for i in sentence:
        print(i)
```
bash output:
```
Martins-MacBook-Air:Work Martin$ python sentence.py
We
must
walk
before
we
can
run
```

b) code:
```
sentence = ["We", "must", "walk", "before", "we", "can", "run"]
for i in sentence[0::2]:
        print(i)
```
bash output:
```
Martins-MacBook-Air:Work Martin$ python sentence.py
We
walk
we
run
```

c) code:
```
num = range(1001)
total = 0

for i in num:
        total += i

print(total)
```
bash output:
```
Martins-MacBook-Air:Work Martin$ python integersum.py
500500
```

d) Both codes were created in the same file, terminal output is identical.

i) code:

```
num = range(11)

for i in num[::-1]:
print(i)
```

bash output:

```
Martins-MacBook-Air:Work Martin$ python countdown.py
10
9
8
7
6
5
4
3
2
1
0
```

ii) code:

```
ten = 10

while ten >= 0:
    print(ten)
    ten -= 1
```

bash output:

```
Martins-MacBook-Air:Work Martin$ python countdown.py
10
9
8
7
6
5
4
3
2
1
0
```

3.4 Simple coordinate manipulation in Python

a) code:

```python
positions = [[0.0, 0.0, 0.0],
             [1.34234, 1.34234, 0.0],
             [1.34234, 0.0, 1.34234],
             [0.0, 1.34234, 1.34234]]

particle2 = positions[1]

print(particle2)
```

b) code:

```python
positions = [[0.0, 0.0, 0.0],
             [1.34234, 1.34234, 0.0],
             [1.34234, 0.0, 1.34234],
             [0.0, 1.34234, 1.34234]]

y2 = positions[1][1]

print(y2)
```

c) code:

```python
from operator import add
positions = [[0.0, 0.0, 0.0],
             [1.34234, 1.34234, 0.0],
             [1.34234, 0.0, 1.34234],
             [0.0, 1.34234, 1.34234]]

t = [1.34234, -1.34234, -1.34234]

new_positions = list(map(add, positions[0], t))
new_positions += list(map(add, positions[1], t))
new_positions += list(map(add, positions[2], t))
new_positions += list(map(add, positions[3], t))

print(new_positions)
```