# **Introduction to programming - ASSIGNMENTS 7**

## **General instructions:**

- Type in and test your programs using Python Idle.
- Bring in your answers to the next assignment (i.e. demonstration) session in memory stick, or save them to a web folder accessible in class. Alternatively, you can bring written answers with you, though this is not recommended.
- Remember to comment your code this does NOT mean, that every single line should be commented. Comment the important parts in your program.
- Prepare to present your solution to the class.

1.

Write a function parseIntegers (mixedList), which gets as an argument a list with mixed type items. The function creates a new list and stores all the integers from the mixedList into it. Finally the new list is returned. Do not change the list given as an argument in any way!

Example run (in Python Shell):

```
>>> testList = [1, 3, "abc ", 5, True, 3.2, False, "Hello ", 7]
>>>print parseIntegers(testList)
[1, 3, 5, 7]
```

2.

Write a function matrixSum (m), which calculates and returns the sum of all items in a 4 x 4 matrix m. You can assume that the items are all numbers.

Example run (in Python Shell):

```
>>> myMatrix = [ [1,1,1,1], [2,2,2,2], [3,3,3,3], [2,2,2,2] ] 
>>> print matrixSum(myMatrix) 
32
```

3.

Write a function capitalizeName (nameAgeTuple), which gets as an argument a tuple containing person's first name and age. The function returns a new tuple with same data, except that the name is capitalized (i.e. the first letter is in upper case).

Example run (in Python Shell):

```
>>> test = ("john ", 32)
>>> print capitalizeName(test)
('John ', 32)
```

4.

A Tic-tac-toe game (see <a href="http://en.wikipedia.org/wiki/Tic-tac-toe">http://en.wikipedia.org/wiki/Tic-tac-toe</a>) can be simulated by a computer using a 3 x 3 matrix of strings. Each string has one of three possible values, "X", "O" or "" (an empty block). Moreover, individual move of a player can be simulated with a tuple, containing three values: the x-coordinate (0..2), the y-coordinate (0..2) and a string containing the mark of the player ("X" or "O").

Hence, the matrix modeling a tic-tac-toe board would look something like Figure 1, after the following tuples are inserted into it:

```
(0, 0, "X")
(1, 1, "O")
(0, 1, "X")
(0, 2, "O")
```

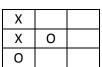


Figure 1 a tic-tac-toe matrix after 4 moves inserted

Write a procedure insertMoves (board, listOfMoves), which has two parameters: a 3 x 3 matrix board and a list of players moves as tuples. The procedure then inserts all moves in a list to the game board.

Example run (in Python Shell):

```
>>> row = [""] * 3
>>> board = [ row, row[:], row[:] ]
>>> moves = [ (0,0,"X"), (1,1,"0"), (0,1,"X"), (0,2,"0")]
>>> insertMoves(board, moves)
>>> print board
[["X", "", ""], ["X", "O", ""], ["O", "", ""]]]
```

Consider a person tuple which consists of three values: person's name (string), height (integer) and weight (integer). Write a function <code>getTaller(listOfPeople, height)</code> which gets as an argument a list of person tuples and returns a new list with all people taller than given height.

#### Example run (in Python Shell):

```
>>> person1 = ("John", 170, 69)
>>> person2 = ("James", 180, 79)
>>> person3 = ("Lisa", 163, 57)
>>> person4 = ("Anne", 174, 55)
>>> person5 = ("Peter", 195, 101)
>>> personList = [ person1, person2, person3, person4, person5 ]
>>> tallerList = getTaller(personList, 175)
>>> print tallerList
[('James', 180, 79), ('Peter', 195, 101)]
```

## 6. \*\* Expert assignment (double points)

Consider a number tuple, which consists of one integer and one floating point number. A number tuple A is considered to be larger than number tuple B, if the product of items in A is larger than product of items in B. For example, tuple (3, 2.0) is larger than tuple (2, 2.5) since  $3 \cdot 2.0 > 2 \cdot 2.5$ .

Write a procedure sortList(tupleList), which gets as an argument a list of number tuples, and sorts the list into increasing order. In other words, for all n

```
1 \le n < len(tupleList): tupleList[n] \ge tupleList[n-1].
```

### Example run (in Python Shell):

```
>>> myList = [(2, 3.0), (3, 1.0), (4, 2.5), (1, 1.0)]
>>> sortList(myList)
>>> print myList
[(1, 1.0), (3, 1.0), (2, 3.0), (4, 2.5)]
```