

## VE477

### Introduction to Algorithms

#### Lab 1

Manuel — UM-JI (Fall 2017)

#### Goals of the lab

- Review on dictionaries
- Review on hash tables
- Starting with Python

## 1 C programming

Using the C standard library, write an efficient implementation of the following data structures:

- Dictionary with the seven basic operation. Use sorted double link lists.
- Implement hash tables.

## 2 Starting with Python

Python is an interpreted programming language first developed in the early 1990s. Being high-level and general purpose it is widely used in computer science and various fields of engineering.

Install Python 3 and run the `python3` command in a terminal. On Linux systems it is recommended to install software through the package manager.

### Basic prompt

At the python prompt run the following basic commands:

```
1 print("Good morning!")
2 (1+2+3*2)/3\%5
3 a=0; b=a+3; b=3*b; print(b)
4 # print("Good morning!")
5 print(I said "Good morning!")
6 print("Good\
7 Morning!")
8 a="good"; b="morning"; print(a + " " + b + "!")
9 nb = input("Input a number: ")
10 help()
```

### Arrays

Python has no native implementation for arrays but instead features *tuples*, *lists*, and *dictionaries*.

**List:** simple list of values; index starts at 0;

**Tuples:** similar to lists but read-only;

**Dictionaries:** values are not numbered, neither are they ordered; values can be accessed using keys;

## Example

```
1 # tuple
2 days = ('Monday','Tuesday','Wednesday','Thursday','Friday','Saturday', 'Sunday')
3 # list
4 students = ['Sam', 'Tom', 'Laura', 'Dave', 'Sarah']
5 print(students[2])
6 students.append('Catherine')
7 del students[3]
8 print(students[2:4])
9 # dictionary
10 gradebook = {'Andrew Parson': [ 'a-', 'b', 'c+'], \
11             'Paul Black': [ 'a-', 'b+', 'a+' ]}
12 del gradebook['Andrew Parson']
13 gradebook['Angela Tree'] = ['d', 'c-', 'f']
14 gradebook
```

## Loops and conditional statements

In the rest of the lab it is recommended to write the code in a text editor and then run it from the terminal using the command `python filename.py`.

In Python any code **must be indented**. A program that is not properly indented **will not run**.

### If statement

```
1 #!/usr/bin/python
2 a=0; b=1; c=4
3 if a < 3:
4     print(a)
5 if a < 3 and b > 0:
6     print (c)
7 elif b == -1:
8     print(b)
9 else:
10    print(a)
```

### Ternary operator

```
1 a=0; b=1
2 a if a < b else b
```

## While loops

```
1 #!/usr/bin/python
2 a=0
3 while a != 0:
4     a = input("Input a number: ")
5     a+=1
6     print(a)
7 print('Bye')
```

## For loops

```
1 #!/usr/bin/python
2 for i in list(range(3,12,2)):
3     print(i)
4 students = ['Sam', 'Tom', 'Laura', 'Dave', 'Sarah']
5 for i in students:
6     print("{} is a student".format(i))
```

## Defining and calling functions

Defining functions requires to use the keyword `def`. Python programs do not need to define a main function.

## Simple functions

```
1 def gm(course):
2     return "Good morning " + course + "!"
3 course='ve477'
4 message=gm(course)
5 print(message)
6 print(gm(input("Input course code: ")))
7 def list(students,courses):
8     for i in students:
9         print(i)
10        del students[1]
11        courses.append('ve281')
12    return students,courses
13 c = ['ve477', 've370']
14 s = ['Sam', 'Tom', 'Laura', 'Dave', 'Sarah']
15 a,b=list(s, c)
16 print(students)
17 print(a, b, d, e)
```

## Function arguments

```
1 def list(students,courses):
2     for i in students:
3         print(i)
4         del students[1]
5     courses.append('ve281')
6 return students,courses
7 c = ['ve477', 've370']
8 s = ['Sam', 'Tom', 'Laura', 'Dave', 'Sarah']
9 # the input order can be changed
10 d,e=list(courses=c,students=s)
11 # define a function with a variable number of input
12 def varin(*args):
13     return args
14 varin(1,2,3); varin(1); varin(1,2,3,4,5,6,7,8)
15 # define a function with a variable number of keywords
16 def varkw(**kwargs):
17     return kwargs
18 varkw(students=['Sam', 'Tom'], courses=['ve477', 've370'],\
19       sports=['tennis', 'soccer'])
```

## First class functions

In Python a function can be passed as an argument or returned by a function.

```
1 def adding(x):
2     def add(y):
3         return x+y
4     return add
5 add_2=adding(2)
6 print(add_2(123))
```

## List, set, and dictionary comprehensions

A list, set, or dictionary comprehension is a way to construct new lists, sets, or dictionaries from existing ones. It follows a syntax similar to set definition in mathematics.

```
1 [i+3 for i in [1, 2, 3]]
2 [5*i for i in [3, 4, 5, 6, 7] if i < 5]
3 {x for x in 'abcdef' if x not in 'abc'}
4 {i: i**3 for i in range(2,10,3)}
```

## Lambda functions

A Lambda function is a Python facility allowing to embed a function within the code. It can be viewed as an “embedded function” which does not need to be declared using `def`.

### Basic use

```
1 l = [lambda x: x ** 2, lambda x: x ** 3, lambda x: x ** 4]
2 for i in l:
3     print(i(2))
4 min = (lambda x, y: x if x < y else y)
5 min(12,16)
```

### Lambda and map functions

```
1 # distances in Miles
2 mi=[ 123, 2, 45, 87.2, 192 ]
3 # distances in km
4 km=map(lambda x: x*1.609344,mi)
5 for i in km:
6     print(i)
```

### Lambda and filter functions

```
1 gradebook = {'Andrew Parson': [ 55, 40, 60], 'Paul Black': [ 85, 80, 95 ]}
2 list(filter(lambda x: x < 60, gradebook['Andrew Parson']))
```

## Exercise

Python 3 short practice:

- Construct more complex examples showing how to use:
  - Lambda functions with `map()`, and `filter()`
  - List, set, and dictionary comprehensions
- Write a phone book application, where the user can add, remove, and edit entries.