

# Test Introduction to Programming

August 2018

Provide your answers by editing the files in the *solutions* (**sol-xxx**) directory. Any free text, such as program comments or explanations, can be written in Danish or English, but English is preferred.

Before handing in, replace the **xxx** the directory name **sol-xxx**, with your student ID.

## 1.txt (5 points)

For each of the following expressions, give their result.

```
3 + 2
3 * 2
3 ** 2
3 / 2
not (False and False)
int("0")
str(3) * 2
```

## 2.txt (5 points)

What is printed?

```
d = {'alice': 'cooper', 'bob': 'marley'}
a = ['peter', 'paul', 'mary', 'bob']
print(d['bob'])
print(a[0])
print(d[a[3]])
```

## 3.txt (5 points)

Assume that

```
s = 'I want to '
t = 'ride my bicycle'
```

What is printed?

```
s[2]
'y' in t
(s + t)[7:10]
t.count('y')
t.find('y')
```

#### 4 (10 + 10 points = 20 points)

Consider the following statement:

```
fullname = firstname + " " + middle_initial + "." + " " + lastname
```

For instance, if the three string variables are “Peter”, “M”, and “Swanson” then the resulting value of fullname would be “Peter M. Swanson”.

##### 4-1.py (10 points)

Rewrite the statement into a function taking three parameters. Choose informative names for the function and the parameters. Include an explanatory docstring.

##### 4-2.py (10 points)

Modify your answer to the previous question so that it returns a meaningful name even if the second argument (for the middle initial) is the empty string. (“Alice”, “”, and “Cooper” should produce “Alice Cooper”, not “Alice . Cooper”).

#### 5.txt (5 points)

What is the purpose of the following function:

```
result = False
for s in colour_list:
    if s is 'blue':
        result = True
```

Your answer is a single of the letters a, b, c, d, or e (and only that), meaning:

- a. to find the first occurrence of the word ‘blue’ in the list
- b. to count the number of colours
- c. to count the number of blue colours
- d. to change all colours in the list to ‘blue’
- e. to determine if the list contains ‘blue’

#### 6.py (8 points)

The intended functionality of the following piece of code is to make all numbers in the input list positive, by changing their sign if necessary.

```
def make_positive(a):
    for i in range(len(a)-1):
        if a[i] < 0:
            a[i] = -a[i]
    return
```

For instance, if `a = [-3, 1, -2, 5, -3]` then the result should be `[3, 1, 2, 5, 3]`. It is not. Correct the function.

### 7.py (12 points)

Write a function

```
def count_small(l):
```

that takes a list `l` of integers and returns the number of integers in `l` that are between 1 and 100 (inclusive). For instance, `count_small([-1, 10, 1, 100, 200, 20, 10])` returns 5.

### 8 (7 + 8 points = 15 points)

Consider the class `ColourChanger` defined in `8/ColourChanger.py`. The purpose of this class is to provide a method that removes the colours *red* and *green* from a list of colours, and replaces them by *black* and *white*. (This could be motivated by wanting to help people with *deuteranopia* (red/green colour blindness).)

#### 8-1.py (7 points)

Currently, the class expects all words to be written in lower case. Make it work so that it even works on upper case. For instance, `Red` should be changed to `Black`.

#### 8-2.py (8 points)

Change the class so that the method does *not* change any colours unless the list contains *both* red and green. For instance, `['red', 'Green']` should be changed to `['black', 'White']` as before, but `['red', 'red', 'yellow', 'Red']` should not be changed.

### 9 (7 + 8 + 10 points = 25 points)

Consider the class `9/Mario.py`. It contains a very simple model of a video game character called Mario and his position as a two-dimensional integer coordinate. There is a single “star” on the playing field; Mario gains a point when he reaches it.

#### 9-1.py (7 points)

Add methods `left`, `right`, and `down` with the corresponding functionality.

### 9-2.py (8 points)

The files `moves-i.txt` for various `i` contain a sequence of characters `LRUD` where `L` means “left” etc. Add a method `simulate(1)` that takes a sequence of characters and returns the number of points collected by Mario if he moves as indicated, always starting from position `(0, 0)` and with the star at `(0, 2)`.

### 9-3.py (10 points)

The file `9/field-1.txt` contains a simple textual representation of a playing field, where `*` denotes the position of the star, and `#` denotes the position of an obstacle that kills Mario. The bottom left corner is `(0, 0)`. When Mario is killed, he can no longer move. Rewrite `Mario.py` so that it reads the file `field-1.txt`. Calling `simulate(1)` should now move Mario through this playing field, correctly collecting points or killing him, and return the number of points collected.