

- If a multiple choice question has several correct answers, you must give all correct answers to receive credit.
 - **Write clearly.** Answers that are difficult to read may receive 0 points.
 - You must pass part A (4 correct out of 8 questions) to have your part B graded.
 - Write only on one side of each paper!
 - No **import** are allowed in your answers in part B (neither Python's standard modules nor external libraries) unless it is explicitly mentioned in the assignment. Built-in functions like **len**, **range**, **print**, and **sum** are allowed.
 - **Aids:** An A4 with as much information as you want. You can write on both sides.
 - **Grade thresholds:** E: 10, D: 12, C: 14, B: 16, A: 18, of maximum 20.
-

Part A: Multiple choice

Please collect your answers to part A on a single piece of paper.

1. What is printed by the code to the right?

- A. 3
- B. 4
- C. 7
- D. 10

```
y = 10
for x in [2, 1, 3]:
    y -= x
print(y)
```

E. An exception is raised.

2. What is the printout from the code to the right?

- A. -5
- B. 0
- C. 6
- D. 10
- E. 14

```
def f(x=2, y=8):
    y = 4
    return x + y

x = 10
y = -15

print(f())
```

3. What is the purpose of the Python instruction `open`?

- A. To handle exceptions.
- B. To get access to code from modules.
- C. To make it possible to read from and write to files.
- D. To ask the user for a string.
- E. None of the above.

4. What will be the value of the variable `text` after running the code to the right?

- A. `top`
- B. `tttoop`
- C. `oop`
- D. `tooppp`
- E. `opp`

```
text = ""
msg= "top"
for j in range(len(msg)):
    i = j
    while i > 0:
        text += msg[j]
        i -= 1
```

5. Which of the following statements are true for Unix (assuming the bash interpreter as used in the course compendium)?

- A. `ls` lists the contents of a folder.
- B. `rm file.txt` creates a new file `file.txt`
- C. The `|` command is used to transfer data between commands (pipelines)
- D. `mv old.txt new.txt` will keep the file `old.txt`.
- E. `cp old.txt new.txt` will keep the file `old.txt`.

6. Which of the following calls will return 5 given `m = [[1, 2, 3], [4, 5, 6], [7, 8, 9]]`?

- A. `m[1][1]`
- B. `m[2][2]`
- C. `m[1, 1]`
- D. `m[2, 2]`
- E. It is not possible to access 5 from within `m` without iterating.

7. Which of the following statements are true in Python 3?

- A. Dictionaries can contain lists as values.
- B. Dictionaries can contain dictionaries as values.
- C. Dictionaries can contain lists as keys.
- D. Lists can contain dictionaries.
- E. `argparse` is a module for parsing arguments from the command line.
- F. Every Python file (`.py`) is a module.

8. What is the result of the code to the right?

- A. `{'D': 1, 'A': 1, '7': 1, '0': 1, '6': 4 }`
- B. `{'D': 6, 'A': 6, '7': 6, '0': 6, '6': 6 }`
- C. `{'D': None, 'A': None, '7': None, '0': None, '6': None }`
- D. `{ 'D': 1, 'A': 1, '7': 1, '0': 1, '6': 2 }`
- E. An error is raised.

```
s = 'DA7066'
d = {}
for x in s:
    d[x] = s.count(x)
```

Part B: Coding

Please use a separate piece of paper (or several) for each question in part B. Multipart questions such as 9A and 9B can be written on the same piece of paper.

9.

- A. Write a function with the name `contains_substring(lis, ss)` that takes a list of strings `lis` as an argument and returns a list of all strings in `lis` containing the string `ss`. (2p)

Example usage:

```
[In: ] print(contains_substring(['ab', 'cde'], 'de'))
[Out:] ['cde']
[In: ] print(contains_substring(['apple', 'strawberry', 'pear'], 'r'))
[Out:] ['strawberry', 'pear']
```

- B. Add error handling to `contains_substring(lis, ss)` using `try-except`. The function should be able to handle lists that contain elements which are not strings (for example `int`, `float`, `bool`) by printing `print('invalid datatype at index:', i)` for the items where this happens, where `i` is the index variable. (1p)

Example usage:

```
[In: ] print(contains_substring(['ab', 'cde', 2, True], 'de'))
[Out:] ['cde']
[In: ] print(contains_substring([False, 'apple', 'strawberry', 'pear', 3.14],
                                'r'))
[Out:] ['strawberry', 'pear']
```

10. The hacker @Kim_L337_Hax0r has decided that comments are unnecessary and has written code to remove all comments from python files (see code below). The code contains precisely two errors (can be either syntactical or semantical). What should be corrected for the code to work as intended? One point per found error. (2p)

```
def line_cleaner(line):
    for i in len(line):
        if line[i] == '#':
            return line[i:]
    return line

def comment_remove(in_filename, out_filename):
    with open(in_filename) as in_handle, open(out_filename, 'w') as out_handle:
        for line in in_handle:
            print(line_cleaner(line), file=out_handle, end='')
```

11. Write a function `substring_abundance` that takes a string `s` and an integer `k` and returns the substrings of length `k` (`k`-mers) and their counts as a dictionary. (2p)

Example usage:

```
[In: ] print(substring_abundance('AAA', 2))
[Out:] {'AA' : 2}
[In: ] print(substring_abundance('ACGACGA', 3))
[Out:] {'ACG' : 2, 'CGA' : 2, 'GAC' : 1}
```

12. Write a function `most_abundant` that takes a string `s` and an integer `k` and returns the most frequent substring of length `k`. Your function should make use of (i.e., call) the function `substring_abundance` in question 10. If there are ties, the function should return all of the most abundant substrings. **Note:** you don't need to solve question 10 to get points on this questions as you may simply use the output from the function call to `substring_abundance`. (3p)

Example usage:

```
[In: ] print(most_abundant('AAA', 2))
[Out:] {'AA' : 2}
[In: ] print(most_abundant('ACGACGA', 3))
[Out:] {'ACG' : 2, 'CGA' : 2}
```

13. Write a function that reads in numbers from a user. If the user enters a number that has already been entered before, the function should print that the number has been entered before. If the user enters the string 'quit', the function should return the numbers in a list; the order of the numbers in the list does not matter. You can assume that the user enters appropriate input, i.e., you don't need to do any error handling. (2p)

Example usage:

```
[In: ] lst = read_numbers()
[Out:] Write a number: 1
       Write a number: 1
       The number exists.
       Write a number: 99
       Write a number: 2
       Write a number: 99
       The number exists.
       Write a number: quit
[In: ] print(lst)
       [1, 99, 2]
```