Duration: **105 minutes**
Aids Allowed: **Reference sheet handed out with the test**

Student Number: |_|_|_|_|_|_|_|_|_|_|_|_|_|

Family Name(s): _____

Given Name(s): _____

Lecture Section: ☐ LEC0101 (M4, W4, R11)

☐ LEC0102 (M5, W2, R3)

---

*Do **not** turn this page until you have received the signal to start.*
In the meantime, please read the instructions below *carefully*.

This term test consists of 6 questions on 22 pages (including this one), printed on both sides of the paper. *When you receive the signal to start, please make sure that your copy of the test is complete, fill in the identification section above, and write your name on the back of the last page.*

Answer each question directly on the test paper, in the space provided, and use the reverse side of the pages for rough work. If you need more space for one of your solutions, use the reverse side of a page and *indicate clearly the part of your work that should be marked.*

When you are asked to write code, *no* error checking is required: you may assume that all user input and argument values are valid, except where explicitly indicated otherwise.

Write up your solutions carefully! Comments and docstrings are *not* required to receive full marks, except where explicitly indicated otherwise. However, they may help us mark your answers, and part marks *might* be given for partial solutions with comments clearly indicating what the missing parts should accomplish.

You may use functions written to answer one question as part of the solution to another question.

You may **not** `import` any modules unless otherwise specified.

MARKING GUIDE

1: _____/ 8

2: _____/12

3: _____/ 8

4: _____/ 4

5: _____/ 4

6: _____/ 1

TOTAL: _____/37

Good Luck!

OVER...

Use this page for rough work—clearly indicate any section(s) to be marked.

Question 1. [8 MARKS]

Treat each subquestion independently (*i.e.*, code in one question is not related to code in another), and answer each question in the space provided.

Part (a) [2 MARKS]

Complete the following function. The function takes in a string `holiday_name` with the name of a holiday, and *returns* the string `"Happy <holiday_name>!"`. For example, the call `wish_happy_holiday("Halloween")` should return the string `"Happy Halloween!"`.

```
def wish_happy_holiday(holiday_name):
```

Part (b) [2 MARKS]

Complete the following function. The function takes in a list `L` of even length, and prints the first half of the elements of the list. One element should be printed per line. For example,

```
print_first_half(["pumpkins", "candy", "costumes", "autumn"])
```

should print

```
pumpkins
candy
```

```
def print_first_half(L):
```

Use this page for rough work—clearly indicate any section(s) to be marked.

Part (c) [2 MARKS]

Complete the function `h()` so that the effect of running the code below is to print "trick or treat" (and nothing else). Quotes should not be printed.

```
def h():




trick = "midterm"
treat = "exam"
treat = h()
print(trick + " or " + treat)  #should print "trick or treat"
```

Part (d) [2 MARKS]

What is the output of the following piece of code?

```
x = 3
y = x - 2
y += (x + y)
x = 7
print(x, y)
```

Use this page for rough work—clearly indicate any section(s) to be marked.

Question 2. [12 MARKS]

Part (a) [4 MARKS]

Complete the following function. The function takes in a list of strings `costumes`, and returns the number of times that `"engineer"` appears in the list. For example,

`count_engineers(["engineer", "doctor", "firefighter", "engineer", "pirate", "artsie"])`

returns 2, since `"engineer"` appear twice in the list given to the function. You must *not* use `list`'s `.count()` method.

```
def count_engineers(costumes):
```

Part (b) [4 MARKS]

Write code that prints the smallest factor (larger than 1) of 272483. (For example, the smallest factor of 6 is 2 since $6 = 2 \times 3$, and the smallest factor of 21 is 3 since $21 = 3 \times 7$.)

Use this page for rough work—clearly indicate any section(s) to be marked.

Part (c) [4 MARKS]

In Python, you can use a *list of lists* to store a matrix, with each inner list representing a row. For example, you can store the matrix

$$\begin{pmatrix} 5 & 6 & 7 \\ 0 & -3 & 5 \end{pmatrix}$$

by storing each row as a list: `M = [[5,  6, 7], [0, -3, 5]]`.

Complete the following function. The function takes in a matrix `M` in a list-of-lists format, and two column indices `i` and `j`. The matrix `M` can be of arbitrary size. The function modifies the matrix M so that column `i` and column `j` switch places.

For example, for

```
M = [[5,  6, 7],
     [0, -3, 5]]
```

`switch_columns(M, 0, 1)` modifies `M` to become

```
M = [[6,  5, 7],
     [-3, 0, 5]]
```

since column 0 and column 1 got switched. You may assume that `i` and `j` are valid column indices. **Please make sure that the columns and not the rows are switching places!**

```
def switch_columns(M, i, j):
```

Use this page for rough work—clearly indicate any section(s) to be marked.

Question 3. [8 MARKS]

Each of these subquestions contains a piece of code. Treat each piece of code independently (*i.e.*, code in one question is not related to code in another), and **write the expected output for each piece of code**. If the code produces an error, write down the output that the code prints before the error is encountered, and then write "ERROR." You do not have to specify what kind of error it is.

Part (a) [2 MARKS]

```
def f(n):
  print(n)

n = 1
m = 2
f(m)
```

Output:

Part (b) [2 MARKS]

```
def f(L):
  L = [3]

L = [5]
f(L)
print(L[0])
```

Output:

Part (c) [2 MARKS]

```
def f():
  L[0] = 0
  L = [4, 5, 6]

L = [1, 2, 3]
f()
print(L)
```

Output:

Part (d) [2 MARKS]

```
L = [[1, 2], 3]
M = L[:]
M[0][1] = 5
M[1] = 5
print(L)
```

Output:

Use this page for rough work—clearly indicate any section(s) to be marked.

Question 4. [4 MARKS]

A list is symmetric if it is the same read from the left and from the right. For example, `[1, 2, 3, 2, 1]` is symmetric, `[1, 2, 2, 1]` is symmetric, but `[1, 2, 3]` is not symmetric. Write the function `is_symmetric(L)` that returns `True` iff the list `L` is symmetric.

Use this page for rough work—clearly indicate any section(s) to be marked.

Question 5. [4 MARKS]

We call a list *almost-symmetric* if it could be made symmetric by swapping two of its elements. For example, `L = [1, 2, 1, 2]` is almost-symmetric since it could be made symmetric by swapping `L[2]` and `L[3]`. Write the function `is_almost_symmetric(L)` that returns `True` iff the list of integers `L` is almost-symmetric. You may use the function `is_symmetric()` even if you have not implemented it.

Use this page for rough work—clearly indicate any section(s) to be marked.

Question 6. [1 MARK]

Write a function that takes in a list of 8 integers, and returns `True` iff some non-empty subset of the integers sums up to 0. For example, the function should return `True` for `[-3, 10, 20, 5, 10, 11, 2, 1]` since $-3 + 2 + 1 = 0$, and the function should return `False` for `[1, 2, 3, 4, 5, 6, 7, 8]`, since no subset of those numbers sums up to 0.

Use this page for rough work—clearly indicate any section(s) to be marked.

Use this page for rough work—clearly indicate any section(s) to be marked.

On this page, please write nothing except your name.

Family Name(s): _____

Given Name(s): _____