Duration: **110 minutes**
Aids Allowed: **Reference sheet handed out with the test**

Student Number:	
UTORid:	
Family Name(s):	
Given Name(s):	
UofT email:	@mail.utoronto.ca

*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 7 questions on 12 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.*

You **must write your student number at the bottom of every odd-numbered page**. **Failure to do so will result in a deduction of 5 points.**

Answer each question directly on the test paper, in the space provided. *indicate clearly the part of your work that should be marked.*

If you need extra space, you can use the last page of the midterm. **Indicate clearly that you have done so if you have**.

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:	______/ 15
# 2:	______/ 30
# 3:	______/ 10
# 4:	______/ 10
# 5:	______/ 10
# 6:	______/ 10
# 7:	______/ 15
TOTAL:	______/100

Good Luck!

Question 1. [15 marks]

Write a function that takes the number `n` and returns a list of all the perfect squares between 0 and n. A perfect square is a number s such that $k^2 = s$ for some integer k. For example, `get_perfect_squares(36)` should return the list `[0, 1, 4, 9, 16, 25, 36]`

```
def get_perfect_squares(n):
```

Question 2. [30 MARKS]

Part (a) [15 MARKS]

Write a function that computes the product of the elements of a list of integers. For example, `prod([2, 3, 4])` should return 24, since $2 \times 3 \times 4 = 24$.

```
def prod(L):
```

Part (b) [15 MARKS]

Write a function with the signature `duplicates(list0)`, which returns `True` iff list0 contains at least two adjacent elements with the same value.

Question 3. [10 MARKS]

Write code that repeatedly prompts the user for input, and then outputs the number of items that the user entered the input before the user entered `"pumpkin spice latte"` (with any capitalization)

For example, an interaction with the user might look like

```
What is your order? (User input:) Pumpkin pie
What is your order? (User input:) Candy
What is your order? (User input:) pumPkin Spice latte
2
```

The number `2` is printed because the user ordered two items before ordering a pumpkin spice latte.

Question 4. [10 MARKS]

Recall that a matrix can be stored as a list of lists. For example, the matrix

$$\begin{pmatrix} 5 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

can be stored as `[[5, 0, 0], [0, 0, 1]]`

Write a function that takes in two matrices stored as lists of lists of numbers, and returns the sum of the two matrices, stored as a list of lists. If the two matrices cannot be added because of a dimension mismatch, return the string `"ERROR"`.

```
def matrix_sum(A, B):
```

Question 5. [10 MARKS]

Consider the following dictionary that contains the information on which kids got which candy in which house:

```
    halloween_haul = {"house1": {"Annie": ["snickers", "mars"],
                                 "Johnny": ["snickers"] },
                      "house2": {"Annie": ["coffee break", "mars"],
                                 "Jackie":["coffee break"]}
}
```

Write a function that takes in a dictionary in a format like the above and returns the name of the kid who collected the most candy items. For example `luckiest_kid(halloween_haul)` should return `"Annie"`, since Annie collected 4 candy items in total. The input dictionary can have any number of houses, any number of kids, and any number of different candy items. Assume that there is just one kid who collected the most candy items.

```
def luckiest_kid(haul_dataset):
```

Question 6. [10 MARKS]

Write a function that, when called, returns the next digit of π (approx 3.14159...). You may assume that the function will not be called more than 10 times.

The function would be used like this:

```
print(next_digit_pi()) # 3
print(next_digit_pi()) # 1
print(next_digit_pi()) # 4
print(next_digit_pi()) # 1
```

You may import `math` and use `math.pi`

```
def next_digit_pi():
```

Question 7. [15 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. For each subquestion, you will earn either 3 or 0 marks.

Part (a) [3 MARKS]

```
def f(L):
  global L
  L = [1, 2, 3]
  L[0] = 5
  print(L[0])

L1 = [4, 5, 6]
f(L1)
print(L)
print(L1)
```

Output:

Part (b) [3 MARKS]

```
def g(L):
  L = [1, 2, 3]
  L[0] = 3

L = [4, 5, 6]
g(L)
print(L)
```

Output:

Part (c) [3 MARKS]

```
s = "abcd"
print(s)
sorted(s)
print(s)
```

Output:

Part (d) [3 MARKS]

```
L = [1, [2, 3], 4 ]
L1 = L[:]
L[0] = 7
L[1][0] = 8
L1[1][1] = 9
print(L)
print(L1)
```

Output:

Part (e) [3 MARKS]

```
def greeting(s):
    global s
    s = "Happy " + s
    print(s)

res = greeting(s)
print(res)
```

Output:

On this page, please write nothing except your name.

Family Name(s): ____

Given Name(s): ____