CSCA08 Fall 2017 Term Test #2
Duration — 110 minutes
Aids allowed: none

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

**Markus Login:** ____________________

**Last Name:** ____________________ **First Name:** ____________________

## Question 0. [1 MARK]

Carefully read and follow all instructions on this page, and fill in all fields. Write all answers neatly in the space provided, or indicate clearly where your question continues.

Please place a checkmark (✓) beside your tutorial session

| Tutorial Number | Date/Time | Room | TA Name | Check |
|---|---|---|---|---|
| TUT0001 | FR 09:00 10:00 | AA 208 | David Liu | |
| TUT0002 | FR 10:00 11:00 | IC 326 | Kevin Gao | |
| TUT0003 | FR 13:00 14:00 | BV 361 | Meghan | |
| TUT0004 | FR 14:00 15:00 | MW 160 | Saurav | |
| TUT0005 | WE 19:00 20:00 | HW 402 | Mohammad | |
| TUT0006 | TH 10:00 11:00 | IC 320 | Pat | |
| TUT0007 | TU 09:00 10:00 | AA 207 | Prantar | |
| TUT0008 | TU 10:00 11:00 | BV 355 | William | |
| TUT0009 | TH 09:00 10:00 | BV 260 | Vladimir Efimov | |
| TUT0010 | WE 09:00 10:00 | AA 204 | Maheshan | |
| TUT0011 | WE 09:00 10:00 | HW 308 | Dharmik | |
| TUT0012 | TH 13:00 14:00 | SW 319 | Vincent | |
| TUT0013 | FR 13:00 14:00 | HW 402 | David Liu | |
| TUT0014 | TH 10:00 11:00 | MW 170 | Andrew | |
| TUT0015 | FR 11:00 12:00 | IC 326 | Siyang | |
| TUT0017 | TU 15:00 16:00 | AA 208 | Aiyaz | |
| TUT0020 | WE 11:00 12:00 | BV 264 | Eric | |
| TUT0021 | MO 11:00 12:00 | HW 408 | Ralph | |
| TUT0022 | MO 12:00 13:00 | AA 207 | Jason | |
| TUT0023 | FR 12:00 13:00 | AA 204 | Frank | |
| TUT0025 | TH 13:00 14:00 | HW 408 | Roleen | |
| TUT0027 | TU 11:00 12:00 | BV 355 | Keegan | |
| TUT0029 | TH 11:00 12:00 | IC 120 | Lucy | |

*Do **not** turn this page until you have received the signal to start.*

This exam consists of 3 questions on 14 pages (including this one). *When you receive the signal to start, please make sure that your copy is complete.*
Proper documentation is required for all functions and code blocks. If you use any space for rough work, indicate clearly what you want marked. Please read all questions thoroughly before starting on any work. To receive full marks on Question 0, draw a smiley face in the top right corner of this page.
The University of Toronto's Code of Behaviour on Academic Matters applies to all University of Toronto Scarborough students. The Code prohibits all forms of academic dishonesty including, but not limited to, cheating, plagiarism, and the use of unauthorized aids. Students violating the Code may be subject to penalties up to and including suspension or expulsion from the University.
Just to check that you're still reading, don't draw a smiley face as instructed above, draw a star instead.

# 0: ______/ 1

# 1: ______/10

# 2: ______/30

# 3: ______/ 4

TOTAL: ______/45

*[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]*

## Question 1. [10 marks]

Write the output of the following code in the box on the right.

```
def f1(x, y):
    c = x[:]
    x[0] = 99
    print(c, x, y)
    return x

def f2(x, y):
    c = x[:]
    (c[0], c[y]) = (c[y], c[0])
    print(c, x, y)
    (x[0], c[y]) = (c[0], x[y])
    return c[x[y]]

def f3(x, y, z):
    x[y][z] = x[z][y]
    c = x[:]
    c[0][0] = f1(c[0], c[1][2])
    print(c[1:])
    x[1][1] = f1(c[0], c[1][2])

a = [1, 2, 3, 4, 5]
b = 2
print("STEP 1")
print(f1(a, b))
a = [[6, 5], [[4, 3], 2], 1]
b = 2
print("STEP 2")
print(f2(a, b))
print(a)
a = [[8, 7], [6, [5], 4], [[3, 2], 1]]
b = 1
c = 2
print("STEP 3")
print(f3(a, b, c))
print(a)
```

*[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]*

## Question 2. [30 marks]

Brian built some tools to work with grade files. The files consist of a name, a course and a grade separated by commas, one grade per line. After the grade data is a line starting with `---` and then other data. A sample file might look something like the following:

```
Alice,CSCA08,99
Bob,CSCA08,70
Alice,MATA31,95
Alice,CSCA48,85
Carol,ABCA01,60
Bob,CSCA48,50
---
This file is private and confidential...
```

Brian wrote a function called `build_marks_dict` that reads a grade file and turns it into a dictionary that maps student names to dictionaries mapping courses to grades. A sample dictionary of that type might look something like:

```
{'Alice': {'CSCA08': 99.0, 'MATA31': 95.0, 'CSCA48': 85.0},
 'Bob': {'CSCA08': 70.0, 'CSCA48': 50.0},
 'Carol': {'ABCA01': 60.0}
}
```

He then wrote another function called `calculate_averages` that takes a dictionary formatted like the one above, and produces a dictionary mapping course codes to averages. A sample dictionary of that type might look something like:

```
{'CSCA08': 84.5, 'MATA31': 95.0, 'CSCA48': 67.5, 'ABCA01': 60.0}
```

Brian had the functions completed and tested. But then... disaster struck! The *CODE MANGLER* got hold of Brian's code and mangled it. Deleting all internal and external documentation, removing all indentation, deleting duplicate lines and randomizing the order of the lines. What's worse, he/she mixed the code from the two functions together into one mangled mess.

Your task is to re-assemble the code in the space provided. The mangled code can be found on the opposite pages for your convenience.

Fortunately, the code mangler didn't manage to get hold of the `.test` files, so no need to reconstruct those.

**Mangled code below**

```
input_line = input_file.readline()
input_line = input_line.strip()
(student, course, grade) = input_line.split(',')
average = sum(marks_list)/len(marks_list)
course_to_average = {}
course_to_average[next_course] = average
course_to_grade = {}
course_to_grade = student_to_marks[next_student]
course_to_grade = student_to_marks[student]
course_to_grade[course] = float(grade)
course_to_marks_list = {}
course_to_marks_list[next_course] = [next_grade]
course_to_marks_list[next_course].append(next_grade)
student_to_marks = {}
student_to_marks[student] = course_to_grade
student_to_marks = build_marks_dict(input_file)
course_to_average = calculate_averages(student_to_marks)
for next_course in course_to_grade:
for next_course in course_to_marks_list:
for next_student in student_to_marks:
while(not input_line.startswith("---")):
marks_list = course_to_marks_list[next_course]
next_grade = course_to_grade[next_course]
if(next_course in course_to_marks_list):
if(student in student_to_marks):
if(__name__ == "__main__"):
else:
input_file = open("grades.csv")
input_file.close()
return course_to_average
return student_to_marks
```

**Part (a)** [14 marks]

```
def build_marks_dict(input_file):
```

**Mangled code below**

```
input_line = input_file.readline()
input_line = input_line.strip()
(student, course, grade) = input_line.split(',')
average = sum(marks_list)/len(marks_list)
course_to_average = {}
course_to_average[next_course] = average
course_to_grade = {}
course_to_grade = student_to_marks[next_student]
course_to_grade = student_to_marks[student]
course_to_grade[course] = float(grade)
course_to_marks_list = {}
course_to_marks_list[next_course] = [next_grade]
course_to_marks_list[next_course].append(next_grade)
student_to_marks = {}
student_to_marks[student] = course_to_grade
student_to_marks = build_marks_dict(input_file)
course_to_average = calculate_averages(student_to_marks)
for next_course in course_to_grade:
for next_course in course_to_marks_list:
for next_student in student_to_marks:
while(not input_line.startswith("---")):
marks_list = course_to_marks_list[next_course]
next_grade = course_to_grade[next_course]
if(next_course in course_to_marks_list):
if(student in student_to_marks):
if(__name__ == "__main__"):
else:
input_file = open("grades.csv")
input_file.close()
return course_to_average
return student_to_marks
```

**Part (b)** [14 marks]

```
def calculate_averages(student_to_marks):
```

**Part (c)** [2 marks]

Re-assmble the global code here

*[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]*

## Question 3. [4 marks]

Briefly explain why the following code won't work:

```
my_set = {'A', 'B', 'C'}
my_set.add(['A', 'B', 'C'])
```

*[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]*

*[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]*

*[Use the space below for rough work. This page will not be marked unless you clearly indicate the part of your work that you want us to mark.]*