THE AUSTRALIAN NATIONAL UNIVERSITY

Final Examination – November 2016

COMP1730 / COMP6730

Programming for Scientists

Study Period: 15 minutes

Time Allowed: 3 hours

Permitted Materials: One A4 page (1 sheet) with handwritten notes on both sides. NO calculator permitted. Use a blue or black pen.

Questions are NOT equally weighted. The questions and each of their parts are NOT in order from easy to hard.

Some questions or parts of questions are labelled "COMP6730 students only". These should only *be answered by students enrolled in COMP6730. Students in COMP1730 will* not *receive any marks for answering these questions.*

*For **COMP1730 students**, the exam will be marked out of 30, and worth 30% of the final course mark.*
*For **COMP6730 students**, the exam will be marked out of 35, and worth 30% of the final course mark.*

Where not otherwise indicated, questions are followed by framed blank panels into which your answers are to be written. If the space in the answer panel is not enough, use the additional panels at the end of the exam paper. If you use an additional panel, make sure you specify which question and part it is for.

Please write and express yourself clearly – if we cannot read what you have written or understand you have tried to say, you will not receive marks for you answer.

Student Number (NOT your name):	Course (write "1730" or "6730"):

The following are for use by the examiners.

Q1 (5)	Q2 (4)	Q3 (7/8)	Q4 (4)	Q5 (4/6)	Q6 (2/4)	Q7 (4)	Total (30/35)

Student Number: .....

Question 1 [5 marks]

(a) For each of the following expressions, write the value that it evaluates to, and the type of that value, into the table. If evaluating the expression results in a runtime error, you only have to write "error".

Expression	value	type
`5 / 5 / 5`		
`{ 1 : 2, (3, 4) : (5, 6)}[3,4]`		
`{}`		
`5 // 15`		
`(-1 ** 2 == 1)`		
`{1, 3, 5}[1]`		
`0 and 2 in [0,2]`		
`2 + 3, 4`		
`{"acacc"}`		

[3 marks]

(b) Suppose `a = [[1,2,3], [4,5,6], [7,8,9]]`. For each of the following expressions, write the value that it evaluates to into the table. (If the expression results in an error, you only have to write "error".)

Expression	value	Expression	value
`a[:1][1:]`		`a[1][1:]`	
`a[0][:1] + a[2][2:]`		`a[0][1] + a[2][2]`	
`a[:1][1]`		`a[-1][1:] * a[0][1]`	

[2 marks]

Question 2 [4 marks]

Each of the following pieces of python code attempt to compute the sum of numbers read from a file. For each one, indicate whether it is correct; if it is not, explain *precisely* what is wrong with it. (For example, if it has a syntax error, describe which line, or part of a line, is incorrect; if it results in a runtime error, explain where and why the error happens; if it runs but prints the wrong value, give a concrete example.) Note that any number (zero, one, two, three or all four) of them may be correct. You should assume that the file `numbers.txt` exists, and that it is a text file containing one (integer) number per line.

(a)
```
def sum_file(file):
    total = 0
    for line in file:
        total = total + int(line.strip())
    return total

print("The sum is", sum_file("numbers.txt"))
```

(b)
```
file = open("numbers.txt")
total = 0
while line in file:
    total = total + int(file.readline())
file.close()
print("The sum is", total)
```

(c)
```
def sum_file(filename):
    file = open(filename)
    try:
        return sum(file.readlines())
    finally:
        file.close()

print("The sum is", sum_file("numbers.txt"))
```

(d)
```
file = open("numbers.txt")
total = 0
for line in file:
    total = total + int(file.readline())
file.close()
print("The sum is", total)
```

Question 3 [COMP1730: 7 marks; COMP6730: 8 marks]

(a) A function `funC` is defined as follows:

```
def funC(a, b):
    k = a
    total = 0
    while k != b:
        if k != 0:
            total = total + 1/k
            k = k + 1
    return 1/((b - a) * total)
```

For each of the following calls to the function, indicate whether a runtime error will occur, the function will enter an infinite loop, or if it will return a value. (You don't have to calculate the exact value it returns.)

(*i*) `funC(-2, 2)`

(*ii*) `funC(2, 0)`

[2 marks]

(b) A function `funD` is defined as follows:

```
def funD(x):
    if x % 2 == 0
        return 2 * x
    if x < 0:
        return -x
```

What is returned by each of the following calls to the function?

(*i*) `funD(-2)`

(*ii*) `funD(1)`

[2 marks]

(c) What is printed when the following code is executed? (You must write down all output, in the correct order. Make sure you clearly indicate where there are line breaks.)

```
def funF(a):
    x = a[1:]
    print(x)

x = [3,1,2]
funF(x)
print(x)
```

[1 mark]

(d) Explain the difference between a function and a method.

[1 mark]

(e) Each function call executes in its own namespace. (This is the case not only in python but in the majority of programming languages.) How can you access the values of variables stored in the function's namespace after the function returns?

[1 mark]

(f) **[COMP6730 students only]** What are the benefits of each function call executing in its own namespace?

[1 mark]

Question 4 [4 marks]

(a) For each of the calls to `print` in the following code, write what will be printed into the box after the statement. (In some cases there are several possible outputs, because elements in sets and dictionaries have no order. In those cases, any output that could be printed is acceptable.) If any statement causes an error, assume that it is skipped and execution continues with the following statements (as if the statements were entered interactively).

```
a = { 1 :  '1', 0.0 :  '0', '1' :  0.0 }
b = set(a.values())
a[a[1]] = a[1]
print(b)
```

```
print(a[0.0] + a['1'])
```

```
for key in a.keys():
    b.add(key)
print(b)
```

```
c = {}
for item in a.items():
    c[item[1]] = item[0]
print(c)
```

[2 marks]

(b) Write a function `count_unique_keys` that takes as argument as dictionary and returns the number of distinct keys in it (i.e., not counting repeated keys). Your function should be as simple as possible. (Correct but unnecessarily complicated functions may receive partial marks.)

[2 marks]

Question 5 [COMP1730: 4 marks; COMP6730: 6 marks]

The following function takes as argument a list:

```
def funX(a_list):
    x = a_list.pop(0)
    i = 0
    while i < len(a_list):
        if a_list[i] == x:
            a_list.pop(i)
        else:
            i = i + 1
```

(a) The function does not return a value but modifies the argument list. If the function is called with the list `[1,3,1,2,3,2]`, what is the contents of the list after the function returns?

[1 mark]

(b) Explain in plain English what this function does *in general*. Make your explanation as general and informative as you can. A good answer is one that describes the purpose of the function – something you might put into a comment or docstring – *not* a line by line description of how it works.

[1 mark]

(c) Give two examples of different runtime errors that can occur in the function `funX` above. For each one, give an example of an argument to the function that will cause the error to occur.

[2 marks]

(d) **[COMP6730 students only]** For each of the two runtime errors (exceptions) you identified in (c), explain whether this error should be caught or not, and if it is caught how it should be handled.

[2 marks]

Question 6 [COMP1730: 2 marks; COMP6730: 4 marks]

(a) The location of a file in the file system is specified by a path. There are two kinds of paths: absolute and relative. Give one example of each kind of path. (You can use either posix- or windows-style paths.)

[1 mark]

(b) The floating point number representation has limited precision. Give an example of how you must take this into consideration when writing code that works with floating point numbers.

[1 mark]

(c) **[COMP6730 students only]** Besides limited precision, the floating point number representation has another limitation. What is it?

[1 mark]

(d) **[COMP6730 students only]** Fractional numbers can be represented to arbitrary precision as the ratio of two integers (since the integer type in python is unlimited). Describe at least one reason why it is nevertheless often better to use floating point numbers? (that is, why we should not always use a rational number representation).

[1 mark]

Question 7 [4 marks]

Each of the following pieces of code attempt to define a class namned `ClassX`. The class' constructor takes one argument, which should be assigned to an instance attribute named `value`. For each one, indicate whether it is correct; if it is not, explain *precisely* what is wrong with it. (For example, if it has a syntax error, describe which line, or part of a line, is incorrect; if it results in a runtime error, explain where and why the error happens; if the class definition does not have the specified behaviour, give a concrete example.) Note that any number (zero, one, two, three or all four) of them may be correct.

(a)
```
class ClassX:
    def __init__(x):
        value = x

    self = ClassX(x)
```

(b)
```
class ClassX:
    def __init__(self, x):
        self = x
```

(c)
```
class ClassX

def ClassX.__init__(self, x):
    self.value = x
```

(d)
```
class ClassX:
    def __init__(self, x):
        ClassX.value = x
```