

THE UNIVERSITY OF MANITOBA

COMP1012:
Computer Programming for Scientists and Engineers
Final Exam (3 hours)

Section (please check one):

☐ A01 (Andres TuTh)

☐ A02 (Boyer MWF)

Name
Student number

December 11, 2014

MARKS

Exam Instructions:

- Marks add up to 50.
- No aids are permitted.
- Answer all questions, and write your answers on the exam itself.
- Write your name, student number, and section on this page and any *separated* pages.
- Place your student card on your desk.

Part A: Predict the output (10 × 1 MARK)

There is a separate problem in each row of the table below. In each one, mentally execute the code fragment on the left and enter the expected output in the box on the right. ***None result in an error.*** Use the last page of the exam for scrap work.

For Graders Only:

A: _____ / 10

B: _____ / 16

C: _____ / 12

D: _____ / 7

E: _____ / 5

Total: _____ / 50

	Code Fragment	Expected output
1. [1]	<code>print 3**1 + 2. // 4</code>	
2. [1]	<code>print tuple(("cattle",))</code>	
3. [1]	<code>print 0**3 <= 3 and 3</code>	
4. [1]	<code>print range(-6,-2,2)</code>	
5. [1]	<code>print [0, 1, 2, 3][1:-1]</code>	
6. [1]	<code>print [jj - 1 // 3. for jj in [5, 1, 3, 1]]</code>	
7. [1]	<code>print array([4, 0, 0, 2])[array([0, 1, 2, 3]) <= 1]</code>	
8. [1]	<code>print "Thursday ".strip().upper() .replace("THURS", "FRI")</code>	
9. [1]	<code>print 2 + 1j**2</code>	
10. [1]	<code>print {1, 2}.union({2, 3}) - {1, 4}</code>	

Name

MARKS

Part B: Programming – (16 MARKS)

B.1 through B.4 together make up a program; you may want to look them all over before starting any of them.

B.1 Function read4Words (4 marks)

[4] Define a function read4Words that prompts the user over and over to enter four words, until the user enters four words, separated by blanks, all on one line. Return a list of the four words, all converted to lowercase, to the calling code. For example, in the session shown below, read4Words returns ["to", "be", "or", "not"] .

```
Enter 4 words on one line, separated by blanks: first word
Your input had 2 words; try again.
Enter 4 words on one line, separated by blanks: I am trying!
Your input had 3 words; try again.
Enter 4 words on one line, separated by blanks: To be or NOT
```

```
def read4Words() :
    """Prompt the user to enter 4 words on a line, separated by
    blanks. Repeat until success."""
```

For marker use only	
Item	Mark
A	
B	
C	
D	
E	
Sum	

Name

MARKS

B.1 through B.4 together make up a program; you may want to look them all over before starting any of them.

B.2 Function allCombos (4 marks)

[4] Write a function allCombos that, given a list of four words, returns a list of all strings with one letter from each word. The first letter should come from the first word, the second letter from the second word, etc. Sort the list of strings into alphabetical order. For example, from ['too', 'many', 'items', 'too'], produce 180 words: ['oaeo', 'oaeo', 'oaeo', ..., 'tytt']. Do not remove duplicates.

```
def allCombos(words4) :  
    """Generate all strings that can be made with the 1st character  
    from the 1st word of words4, the 2nd character from the 2nd  
    word, etc. Return a sorted list."""
```

For marker use only	
Item	Mark
A	
B	
C	
D	
E	
Sum	

Name

MARKS

B.1 through B.4 together make up a program; you may want to look them all over before starting any of them.

B.3 Function filterWords (4 marks)

[4]

Define a function filterWords that finds all strings in a list that are words from a given wordlist. It returns a *tuple* containing unique words sorted into increasing alphabetical order. For example, given the set of 180 combos produced by allCombos for the input ['too', 'many', 'items', 'too'], and a wordlist containing all common English words, filterWords returns ('omit', 'onto').

def filterWords(combos, wordList) :
 """Find all strings in combos that are words in wordList,
 and return a sorted tuple without duplicates."""

For marker use only	
Item	Mark
A	
B	
C	
D	
E	
Sum	

Name

MARKS

B.4 Function main [4 MARKS]

[4] Define a function `main` that uses the functions on the previous pages to find all the common English words that can be made by taking one letter from each of four words provided by the user. It should call each of `read4Words`, `allCombos` and `filterWords` exactly once. The program as a whole should produce output very similar to the sample below. Note the spacing. Assume there will be at least seven strings, and show the first five and the last one, separated by "...". In contrast, show *all* the unique words.

Enter 4 words on one line, separated by blanks: *Handle equipment with care!*

User's words: handle equipment with care!

Sorted strings: aeh! aeh! aeha aeha aehc ... nuwr

Unique words: amir ante emir epic heir lite lute nite

```
def main() :  
    """Find all words that can be made from 4 given words."""  
    wordList = getWords("2of12inf.txt") # do NOT write getWords
```

For marker use only	
Item	Mark
A	
B	
C	
D	
E	
Sum	

Name

MARKS

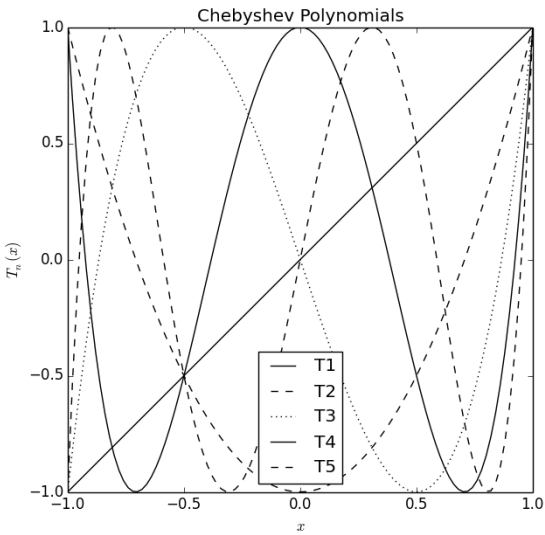
Part C: Programming – (12 MARKS)

C.1 Chebyshev Polynomials [6 MARKS]

[6] Write a script (it doesn't have to have any input nor any functions) to plot the Chebyshev polynomials $T_n(x)$ for n varying from 1 to 5. Use a loop over values of n . Use 101 equally spaced x values from -1 to 1. Do not use any loops related to x ; use numpy array calculations instead. Use black as your colour for all curves, and use solid, dashed and dotted lines. Label the plot as shown. The following definition enables you to determine $T_n(x)$.

$$T_n(x) = \begin{cases} 1 & \text{if } n = 0 \\ x & \text{if } n = 1 \\ 2xT_{n-1}(x) - T_{n-2}(x) & \text{otherwise} \end{cases}$$

```
# no leading comments are required
import matplotlib.pyplot as plt
import numpy as np
```



For marker use only	
Item	Mark
A	
B	
C	
D	
E	
Sum	

Name

MARKS

C.2 Simulate Dice [6 MARKS]

[6]

Write a script with a single function `main` that uses 1000 simulations to estimate the probability that m takes the value 6, where m is defined as follows:



- roll a single fair 6-sided die repeatedly, for as long as the result is an odd number;
- let r be the number of rolls until the first even number: if r is divisible by 3, let m be the sum of the r rolls; otherwise let m be the value of the last roll.

Use at least two global constants. Set the random seed to 2014. Sample output:

Based on 1000 simulations, the probability that $m = 6$ is 0.298

Programmed by <your name>

```
# no leading comments are required; use at least two constants
# define a main function
import numpy as np
import random
```

For marker use only	
Item	Mark
A	
B	
C	
D	
E	
Sum	

Name

COMP1012:
Final Exam (3 hours)

MARKS

Part D: Multiple Choice + Expression (7 x 1 MARK)

For each of the following 6 multiple-choice questions, circle the *single best* answer.

- [1]

1.

Given the following assignments, which of the expressions below has the value 3?

```
list1 = range(5)  
list2 = [list1, list1, 3]  
list1[0] = list2
```

a) list1[2]
b) list2[1][2]
c) list1[0][0][2]
d) list2[0][0][0][2]
e) list1[0][0][0][2]
- [1]

2.

Which of the following format specifiers can be used to print a complex number?

a) %c
b) %d
c) %f
d) %g
e) %r
- [1]

3.

Considering the following sequence of instructions, what is the result of draw(5, '@')?

```
def draw(SIZE, CHAR) :  
    for row in range(SIZE + 1) :  
        text = ""  
        for col in range(min(row + 1, SIZE)) :  
            text += CHAR  
        print text
```

a) @@@@
@@@@
@@@@
@@@
@@
@

b) @
@@
@@@
@@@@
@@@@@
@@@@@

c) @
@@
@@@
@@@@
@@@@@

d) @@@@
@@@@
@@@
@@
@

e) @@@@
@@@@
@@@@
@@@@
@@@@

[1]

4.

Which of the following statements is *not* correct?

a) Lists, tuples, strings and numpy arrays all use the same notation for slices.
b) A string is like a tuple of characters, in that both are immutable.
c) Among lists, tuples, strings and numpy arrays, only tuples *cannot* be concatenated with the '+' operator.
d) One difference between a list and a numpy array is that data items in the array all have the same data type.
e) Among lists, tuples, strings and numpy arrays, you can subtract 5 only from an array.
- Page 8 of 12

Name

MARKS

[1]

5. Which of the formatted prints below would produce this output?

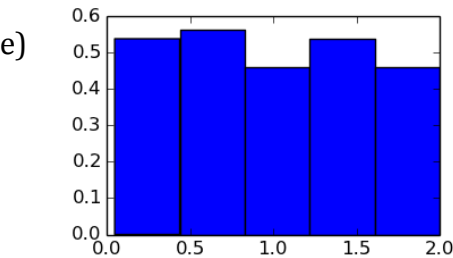
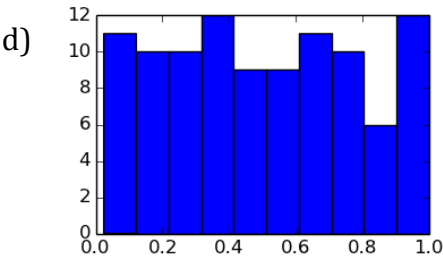
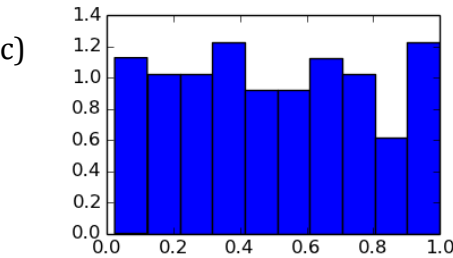
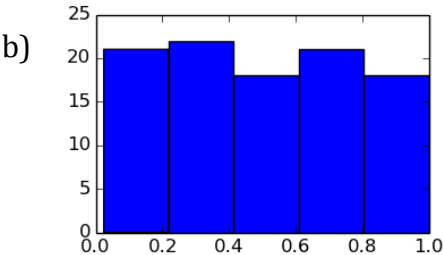
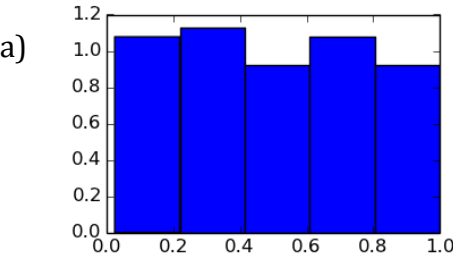
Output: |-2.8571e-01|

- a) `print "Output: |%-20.4e|" % (-2./7)`
- b) `print "Output: |%-20.4f|" % (-2./7)`
- c) `print "Output: |%-20g|" % (-2./7)`
- d) `print "Output: |%20.4f|" % (-2./7)`
- e) `print "Output: |%20.4e|" % (-2./7)`

[1]

6. Which plot could result from the following code?

```
import numpy as np
import matplotlib.pyplot as plt
plt.figure()
numbers = np.random.random(100)
plt.hist(numbers)
plt.show()
```



[1]

7. [1 marks] Using good coding practices and QuizMaster rules, write a Python expression for this expression, assuming `math` has already been imported:

$$\left[\frac{\sqrt{a}}{-e^{\tan(10)}\pi^{\frac{10}{[x]}}} \right]$$

Put expression here

COMP1012:
Final Exam (3 hours)

MARKS

Part E: Short Answer (5 MARKS)

[5] The code below applies the sieve of Eratosthenes to find and return all the prime numbers up to a limit. The code was originally correct, but 6 errors have been added to it. Each error is one of the following: *missing/extra/incorrect item, where item is punctuation, constant, operation, library, function or variable* (including mismatched brackets or quotes). Only numbered lines have errors, and no line has more than one error. *The comments are correct.* Find the errors. In the table at the bottom, give the line number of each error, say what is wrong and how you would fix it. As an example, one error has been done for you. Find five more.

```
#.....findPrimes
1 def findPrimes(limit)
    """Find all prime numbers up to integer limit > 1 and return as array"""
2     maybePrime = np.zeros(limit + 1) != 0 # limit + 1 Trues
3     maybePrime[2] = False # 2 Falses at the start
4     maxFactor = int((limit + 1)**0.5) # largest small factor up to limit
5     for candidate in range(2, maxFactor) :
6         if maybePrime[candidate] :
7             maybePrime[2 * candidate : candidate : ] = False
            # all composites up to limit have False in maybePrime, as at least
            # one of their prime factors has been found. The Trues are prime.
8     primes = np.range( len(maybePrime) )[maybePrime]
9     return
```

(1) line 1... ':' is missing in the **def** statement; add it after the closing parenthesis ')

(2)

(3)

(4)

(5)

(6)

Name

MARKS

Extra space

Name

MARKS

Extra space

