The University of British Columbia

Final Examination - April 25, 2015

Mathematics 210

Section 201

Instructor: Carmen Bruni

Closed book examination Time: 2.5 hours

Last Name: _____ , **First:** _____ **Signature** _____

Student Number _____

Special Instructions:

No books, notes or calculators are allowed.

Rules governing examinations

• Each examination candidate must be prepared to produce, upon the request of the invigilator or examiner, his or her UBCcard for identification.

• Candidates are not permitted to ask questions of the examiners or invigilators, except in cases of supposed errors or ambiguities in examination questions, illegible or missing material, or the like.

• No candidate shall be permitted to enter the examination room after the expiration of one-half hour from the scheduled starting time, or to leave during the first half hour of the examination. Should the examination run forty-five (45) minutes or less, no candidate shall be permitted to enter the examination room once the examination has begun.

• Candidates must conduct themselves honestly and in accordance with established rules for a given examination, which will be articulated by the examiner or invigilator prior to the examination commencing. Should dishonest behaviour be observed by the examiner(s) or invigilator(s), pleas of accident or forgetfulness shall not be received.

• Candidates suspected of any of the following, or any other similar practices, may be immediately dismissed from the examination by the examiner/invigilator, and may be subject to disciplinary action:

(a) speaking or communicating with other candidates, unless otherwise authorized;

(b) purposely exposing written papers to the view of other candidates or imaging devices;

(c) purposely viewing the written papers of other candidates;

(d) using or having visible at the place of writing any books, papers or other memory aid devices other than those authorized by the examiner(s); and,

(e) using or operating electronic devices including but not limited to telephones, calculators, computers, or similar devices other than those authorized by the examiner(s)–(electronic devices other than those authorized by the examiner(s) must be completely powered down if present at the place of writing).

• Candidates must not destroy or damage any examination material, must hand in all examination papers, and must not take any examination material from the examination room without permission of the examiner or invigilator.

• Notwithstanding the above, for any mode of examination that does not fall into the traditional, paper-based method, examination candidates shall adhere to any special rules for conduct as established and articulated by the examiner.

• Candidates must follow any additional examination rules or directions communicated by the examiner(s) or invigilator(s).

1		8
2		14
3		8
4		10
5		8
6		10
7		8
8		8
9		8
10		13
11		5
Total		100

[8]**1**. Define, describe or state **four of the five** of the following terms (if you do them all, the first four will be graded).

a. The Fundamental Theorem of Algebra.

b. A Regular Markov (Transition) Matrix.

c. The Index of Coincidence.

d. Eigenvector.

e. The Prime Number Theorem.

[14] **2**. Sage/Python/LaTeX.

a. [**2**] For the following commands, describe the output (write ERROR if the commands give an error). You need not give the exact output for full marks.

i) `RandomGNP(10,0.25)`

ii) `matrix([[1,1],[0,1]]).eigenvectors()`

b. [**2**] Suppose you have a file called "file.txt". Write Python code to open this file in Python and print out the contents.

c. [**1**] In assignment 6, many of you wrote a command similar to

```
def func(s):
  return sqrt(2*s**2)
```

where s was a side length. Consider instead the function defined by

```
def func2(s):
  return sqrt(2)*s
```

Explain why the second code is substantially better than the first.

d. [**2**] Give a docstring for the following function.

```
def string_has_duplicate_characters(s):
'''



'''
unique_char = []
[unique_char.append(x) for x in s if x not in unique_char]
return len(unique_char) == len(s)
```

e. **[2]** Give an exact LaTeX command that will produce the following text

$$\sum_{n=0}^{\infty} \pi^{-n}$$

f. **[5]** Write a Python class for a video game character. Your character has a name and a position (a point in the Cartesian plane stored as an array/list). The default position is $[0, 0]$. The character can move which should be a method of your class that takes in a string (one of UP, DOWN, LEFT, RIGHT) and modifies your character's position one unit accordingly. Do not document your code.

[8] **3**. Decrypt the message

LYQDOK

given that...

a. [**4**] The original message was encrypted using an Affine Cipher with key $(a, b) = (1, 10)$.

b. [**4**] The original message was encrypted using a Vigenere Cipher with keyword WAX.

The following table might be of assistance.

A	B	C	D	E	F	G	H	I	J	K	L	M
00	01	02	03	04	05	06	07	08	09	10	11	12
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

[10] **4**. Let J be the Jordan canonical form given by

$$\begin{bmatrix} 5 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 5 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

a. [**6**] Fill out the following table about J (there are possibly more rows than necessary).

Eigenvalue	Algebraic Multiplicity	Geometric Multiplicity

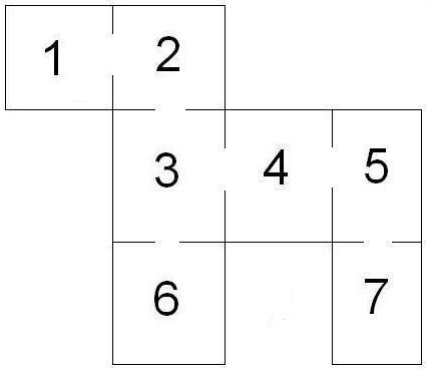
b. [1] What is the determinant of J?

c. [1] What is the trace of J?

d. [1] Is J diagonalizable?

e. [1] Give an eigenvector of J.

[8] **5**. Our graduate student friend from assignment 6 has found himself in another puzzling room. He is stuck in the building with no exit below and once again on a turn has an equal probability of going to the next room or staying put.



a. [**2**] Give the Markov matrix describing the above situation.

b. [**3**] This matrix is regular. What is the minimum power n proving this? Why? (Don't try to compute powers of this matrix!)

c. [**3**] Suppose now that the person starts in rooms 1, 4, or 7 with equal likelihood. What is the state vector after 1 turn?

[10] **6**. Number Theory

a. [**5**] The following text has been encrypted using a Vigenere cipher. Use Kasiski's test to determine a likely keyword length. (Hints: OLE occurs 4 times. The spacing below is to help with counting.)

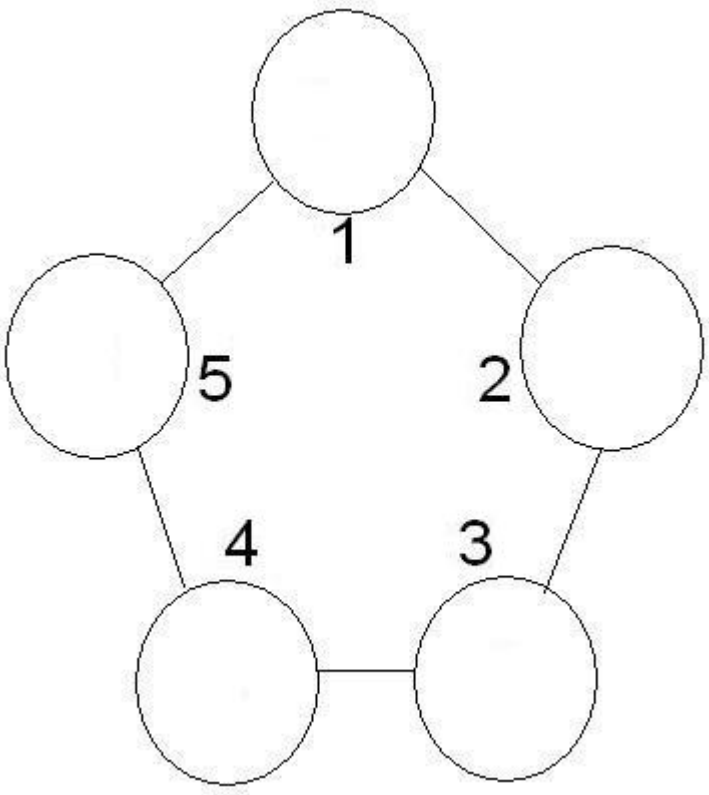
NHYMK	OLENR	UKUYM	JLQIN	MOXIW	TCIMO	MOZOC	OLEXZ
YWANI	IKKUY	CNZCG	AEZOH	MOMDS	DFENC	GHNHY	MKOLE
IWBWI	UMGEB	IDYAO	QCEHX	OSMAH	YZVYO	NCKFQ	ASDYH
IMUFK	WNSIX	UAJLC	XGHYD	NCGHN	HYMKO	LEHJU	PPIIP
YRYFC	XOSHC	CZYHB	EZDXG	NMYON	CXIMA	GFGOL	ZJWZF
CXAZN							

b. [**5**] Find the least positive residue of 3^{100000} modulo 35.

[8] **7**. Graph Theory

a. [**4**] For the graph below, give the adjacency matrix, degree matrix and the Laplace (Kirchoff) matrix.

b. [**4**] Then fill in the circles such that the sum of adjacent vertices is a common multiple of each vertex at each vertex. What does a correct solution correspond to in terms of the adjacency matrix? What is the common multiple and what does this correspond to in terms of the adjacency matrix?



[8] **8**. Choose one of the following (if you do both I will only mark the first one). You will receive zero, one to two points for each good sentence, picture or idea depending on quality.

a. Describe Man-In-The-Middle attacks on public key cryptography schemes and discuss how certificates are used to defend against these attacks.

OR

b. Discuss the page rank algorithm. Given the following matrix, how can one adapt the algorithm to avoid the issue of cycling (or two-linking)?

$$\begin{bmatrix} 0 & 1/2 & 1/3 & 0 & 0 \\ 1/2 & 1/2 & 1/3 & 0 & 0 \\ 1/2 & 0 & 1/3 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

[8] **9**. Snakes and Ladders. In assignment 6 we discussed some safety measures to guard against poorly designed boards. Here we will continue this task. As a reminder, the board Python class from lectures contained the variables **dice**, **ladders**, **num_players**, **num_squares** and **snakes**. The variables **snakes** and **ladders** were arrays of arrays with a start and end position and the other three variables were stored as integers. (No documentation required for this question).

a. [**2**] Write a Sage/Python function that checks if there is a snake on the final square of the board. Your function should take in one parameter, a Board object and returns true if there is a snake on the final square and false otherwise.

```
def check_final_square(board):
```

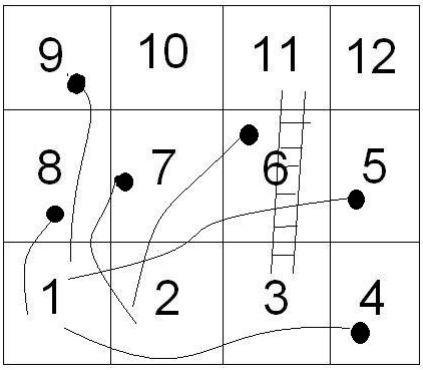
b. [**2**] Write a Sage/Python function that checks if a ladder has an end space that exceeds the board length. Your function should take in one parameter, a Board object and returns true if a ladder has an end square that is not on the game board and false otherwise.

```
def check_ladder_exceeds_board_length(board):
```

c. [**4**] In the assignment, you wrote a function that checked if there were snakes in succession equal to or greater than the number of sides on the dice. Let f be the furthest back tail in this succession, s the start of the succession of snakes and e the end of this succession (see the next page for a concrete example). Write a Sage/Python function that returns true if and only if a player who starts on space f can use ladders to eventually end up past the succession of snakes. You may assume that every square is either blank, contains a ladder (either base or top) or contains a snake (either head or tail) and you may assume that the parameters f, s, and e are correct for the game board.

In the example on the next page, assume board.dice = 6. Now f=1, s=4, e=9 and here snakes (signaled by the dots that slide down to the tails) are from spaces 4 to 9

inclusive. Normally our piece would never be able to get to space 12 however there is a ladder on square 3 going to 11 that avoids this entire successive snake situation and is always reachable since (at least) one of the snakes from 4-9 goes to square 1 which is before square 3.




```
def skip_snakes(board, f, s, e):
```

[13] **10**. Recursion and Looping.

a. **[2]** State the two cardinal rules of recursion.

b. A popular machine learning algorithm is called 2-means clustering. This algorithm takes a set of data points in the Cartesian plane and groups them into two clusters. Given a set of data points D (for example, $\mathbf{D = [(0.5,0.5), (0.25,0.33), (0.9,0.01)]}$) in the box $[0,1] \times [0,1]$, the algorithm starts with two random points $P = (p_1, p_2)$ and $Q = (q_1, q_2)$ in the aforementioned box and computes the distance between the data points and P and Q. If the point is closer to P than Q, include the point in a list S_P otherwise include the point in a list S_Q. Then iterate again by choosing the centroid of the points of S_P as your new P and similarly for Q. Stop after n iterations. (No need for documentation in these questions.)

i) **[3]** Write code that returns the lists S_P and S_Q. Your function should take in D, P, Q.

```
def closer_to_P_or_Q(D,P,Q):
    SP = []; SQ = []




    return (SP,SQ)
```

ii) [**4**] Write code that computes the centroid (the point formed by taking the average of the x-coordinates and y-coordinates of a set of points) of a set of points in the Cartesian plane. You should return a tuple (x, y) corresponding to the centroid.

```
def centroid(points):
```

iii) [**4**] Using the previous parts, complete the code below where n is the number of iterations.

```
def two_means_clustering(D,n):
    P = (random(), random())
    Q = (random(), random())
    SP = []
    SQ = []









    return (SP, SQ)
```

[5] **11**. Alice and Bob have a friend Carol. They would like to use a modified Diffie-Hellman key exchange to share a private key. Create a three way Diffie-Hellman protocol to help Alice, Bob and Carol share a private key. Your method must work against Eve (an eavesdropper) but not Mallory (a non-passive eavesdropper). A diagram might help with your solution.

This page has been intentionally left blank and may be detached and used for rough work.