ZJU-UIUC INSTITUTE

Final Examination

(For Students, please fill in your name and ID number and read any instructions below before starting your exam. Please be aware of your obligation not to receive or give aid to others. Don't take the test out of the exam room.)

Name:	ID:

(For instructors, please complete the form below)

Course: CS101	Semester: 2017 Fall Semester	Instructor: Dr. Liao, Zicheng	
Exam Type: Closed-book √ Open-book □ Partly Open-book □ Take Home □			
Exam Date: 12/29/2017	Start Time: 09:00	End Time: 12:00	Duration: 3 hrs
Total number of pages: 10		Number of questions: 100+5	
Specific requirements and instructions to students: This is a close book exam. So no book, no lecture notes, no cell phone during the exam. Do all problems and show your work for full credit.			

(For marker, please fill in the score and grade and sign below.)

Score:	Grade:
Signature:	Date:

(Please go on to the next page for questions)

CS101 Final Exam

- Be sure to enter your information to start
- Do not turn this page until instructed to or 30 minutes after the exam
- You must not communicate with other students during the exam
- You must not use any electronic devices during the exam
- Any violations detected will result in a penalty up to full credit loss
- This is a 3-hour exam

Part A, Multi-choice questions

This part has 15 questions for Python programming language basics (or MATLAB if explicitly stated). There is only one correct answer to each question unless otherwise stated. Please fill in your answers in the blank table at the end of this section.

1. (2 point) A group of eight bits is called:

(A) a byte

(B) a gobble

(C) a bite

(D) a nybble

2. (2 point) Which of the following is NOT a literal in Python?

(A) '5'

(B) 5

(C) a = 5

(D) True

3. (2 point) Which of the following expressions returns False?

(A) 5%2 AND 1

(B) 7//3 == 2 XOR False

(C) 'a' < 'A'

(D) None of the above

4. (2 point) Consider the following code:

```
def fun(a):
    x = 100
    return a
x = 3
x += fun(x)
```

What value is stored in x after the program is run?

(A) 3
(B) 6
(C) 103
(D) 200
(E) None of the above

5. (2 point) Which of the following assignment operation will cause a runtime error?
(A) x = 10
(B) x = y = 10
(C) x * y = 10
(D) x, y = y, 10

6. (2 point) Consider the following code

```
x = 3
s = (‘%i’%(x+1)) * x **(5%x)
print(s)
```

What does this program print?
(A) 3333333333333
(B) 444444444
(C) 333333333
(D) 36

7. (2 point) Which of the following is NOT the correct way of referring to the second last character of the following string in Python?

```
a = ‘FIRE’
```

(A) a[2]
(B) a[-2]
(C) a[2:-1]
(D) a[end-1]

8. (2 point) Consider the following program:

```
a = list(range(10))
a.extend(a)
a.append(a[:])
```

What is the length of a[-1]?
(A) 1
(B) 10
(C) 20
(D) 22
(E) 40

9. (2 point) Consider the following function definition:

```
def fun():
    p = numpy.random.randint(10)
    if p <= 1:
        return p
    else:
        return fun()
```

What is the **expected** number of times the function *fun* to be invoked when calling it?

(A) 10

(B) 5

(C) 4

(D) 1

(E) None of above

10. (2 point) Consider the following code:

```
a = list(range(10))
count = 0
for i in range(1,len(a)+1):
    for j in itertools.combination(a, i):
        count += 1
```

What is the final value of *count*?

(A) 32

(B) 1023

(C) 1024

(C) 2047

(D) 2048

11. (2 point) Given the following 2D matrix :

$$d = \begin{bmatrix} 5 & 2 & 1 & 4 \\ 3 & 1 & 2 & 1 \\ 3 & 4 & 1 & 8 \end{bmatrix}$$

Which of the following code produces the matrix [1 4; 2 1] in MATLAB?

(A) d(0:2, 2:4)

(B) d(1:2, 3:4)’

(C) d(2:3, 2:3)

(D) d(2:3, 2:3)’

(E) None of the above

12. (2 point) Consider the following code:

```
d = list(range(10))
for i in range(10):
    d[10-i] /= d[i]
i = i+1
```

Which error would this code produce?

(A) NameError

(B) IndexError

(C) IndentationError

(D) ZeroDivisionError

(E) None of the Above

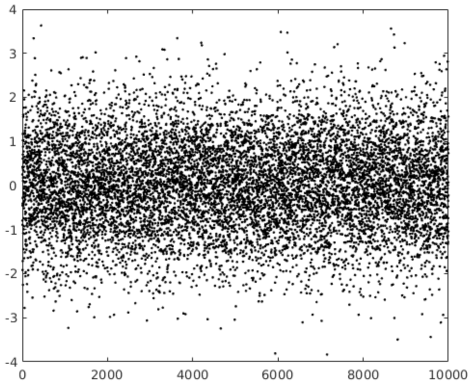
13. (2 point) Which of the following will produce this plot?

(A) plot(randi(10000,1), ‘-’);

(B) plot(randn(10000,1), ‘.’);

(C) plot(rand(10000,1), ‘.’);

(D) plot(rand([-4,4], 10000,1), ‘.’);



14. (2 point) Which of the following strategies suffers from getting stuck at local optimum in optimization?

(A) divide-and-conquer

(B) random walk

(C) hill-climbing

(D) brute-force search

15. (2 point) Given two variables X and Y that have correlation corr(X,Y) = -0.98, which of the following statements are incorrect?

(A) X and Y are highly correlated

(B) X and Y are hardly correlated

(C) the value of Y tends to become larger as X becomes smaller

(D) The increase of value in X causes the value of Y to decrease

(E) We can fairly reliably predict the value of X from Y

(F) There is a strong causal relation from Y to X.

(There are possibly multiple correct answers to this question. Try pick them all)

Your Final Answers for Part A here:

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
11	**12**	**13**	**14**	**15**					

Part B, quick questions (10 points; 2 for each)

1. What's the minimum possible number of bits for representing a Boolean type in Python? ______________________
2. In computer programming languages, an 8-bit unsigned integer represents nonnegative integer numbers starting from 0. What is the largest number it can represents? ______________________
3. What is the advantage and disadvantage of interpreted languages versus compiling languages? (enumerate 1-2 for each; no need to elaborate)
4. What is the difference of mutable types versus immutable types in Python? (enumerate 1-2 points; no need to elaborate)
5. Why do we need to have data *types* in programming languages?

Part C, programming hackthon

This part has 5 programming problems. You may choose to use Python or Matlab according to the problem specification. Please carefully read the problem definitions.

1. (10 points) ***String Operation***

A complex number c is represented as `A + Bj` in python, where `A` is a float number for the real part and `B` is a float number for the imaginary part. Assuming you are writing a program to load complex numbers from a text file, each line is a complex number in plain text (ASCII). To make your code easy to read, you decide to write a function **Str2ComplexNumber(s)**, which takes a string ***s*** of a complex number (e.g., '3.14 + 2j') as argument and returns the corresponding complex number (e.g., 3.14 + 2j). Write your function definition below in Python.

2. (10 points) ***Find Median***

Write a function **FindMedian(L)** that takes a list of numbers **L** as argument, and returns the median. For a list of N numbers, the median is the (int(N/2) + 1)th largest number in the list if N is an odd number, or the average of the N/2 th and (N/2 + 1)th largest number if N is an even number. You may choose to use Python or MATLAB (you shall NOT use their built-in function `median`).

3. (10 points) ***Randomness and Simulation***

In a Poker game, we have a deck of 52 cards -- each card is worth of some points ranging from 1 to 13, and there are 4 identical cards for each of them (52 = 13x4). Now, write a function **PokerGame**() to simulate the process of randomly drawing 3 cards from a deck *with replacement*, and return **True** if the sum of points on the 3 cards equals 13, and **False** otherwise. You may choose to write this in Python or in MATLAB.

4. (15 points) ***Image Processing***

An image is represented as a 2D array of pixels, and for grayscale images each pixel is denoted by a single number for its grayness (from 0 to 1). So a grayscale image can be conveniently stored and processed as a matrix in Matlab. Now given a grayscale image I, its horizontal gradient I_x is defined for every pixel (except the last column) as follows:

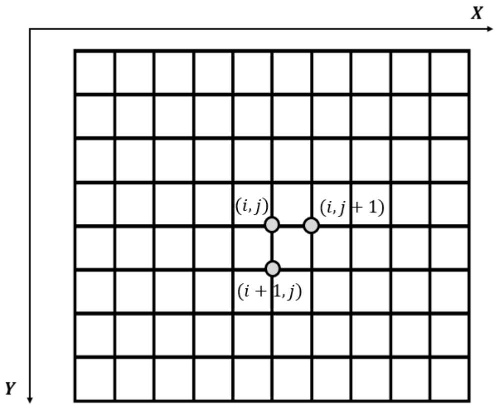
$$I_x(i,j) = I(i,j+1) - I(i,j)$$

where (i,j) are the row and column indices as shown in the figure. Similarly, the vertical gradient I_y is defined for every pixel (except the last row) as follows:

$$I_y(i,j) = I(i+1,j) - I(i,j)$$

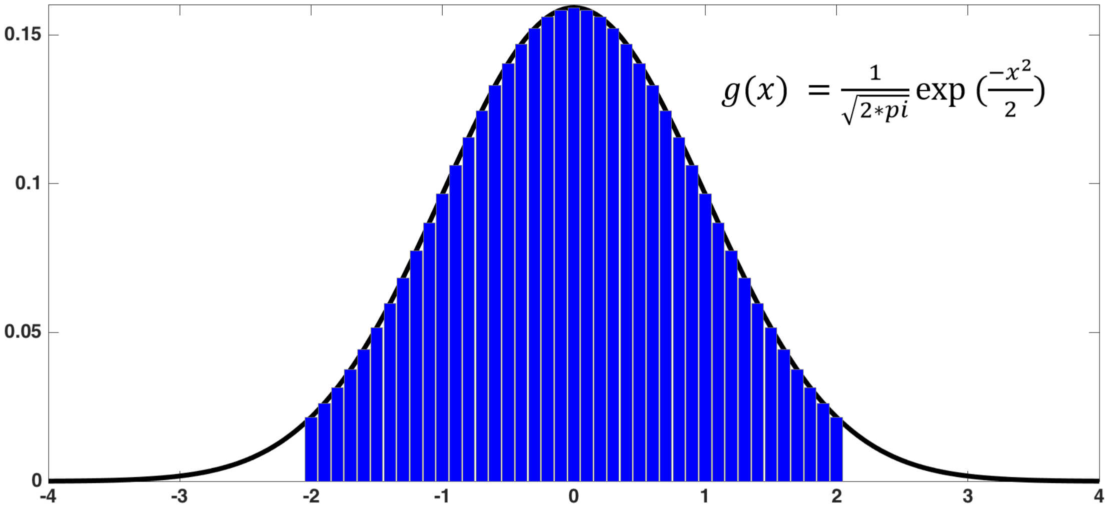
Now write a MATLAB function **ComputeLargeGradients(I)** that takes a grayscale image **I** as input. The function should compute the gradient field I_x and I_y, and return the indices of pixels whose gradient magnitude ($\sqrt{I_x(i,j)^2 + I_y(i,j)^2}$) is in the highest 5 percentile.

(Hints: 1. MATLAB function `prctile` operates row/column-wise if the input is a matrix; 2. `A(:)` converts matrix `A` into a column vector; `[x,y] = find(A)` returns the x/y indices of all non-zero entries in A)



5. ***Statistics & Computational Approximation***

The standard normal distribution function $g(x) \ = \frac{1}{\sqrt{2*pi}} \exp\,(\frac{-x^2}{2})$ defines the *probability density* $g(x)$ of drawing a random value $x \in [-\infty, +\infty]$ (higher probability density means better chance to be drawn). Now given the probability density function, the probability of drawing a number in the range $[a, b]$ (a<b) equals to the ***area*** the function curve encloses in the range from a to b. For example, for a = -2 and b = 2, it is roughly the dark area shown in the Figure below.



To mathematically calculate the exact area can be tricky, and the solution only exists for a set of special functions. A more practical way is the computational approximate approach we introduced -- that is, to subdivide the range [a, b] into many equally spaced intervals, and approximate the area by the sum of areas of the bins whose width is the interval size and the height is determined by the function $g(x)$.

(a) Write a function **ComputeProbilityInBetween(a,b)** to compute and return the probability of drawing a number in the range of [a, b] from standard normal distribution. You may choose to use Python or MATLAB. (15 points)

(b) Now, write another function **ComputeProbilityAbove(a)** to compute and return the probability of drawing a number larger than **a** from the standard normal distribution. (5 points **BONUS**)