Final Exam, Version 3

CSci 127: Introduction to Computer Science

Hunter College, City University of New York

19 December 2018

Exam Rules

- Show all your work. Your grade will be based on the work shown.
- The exam is closed book and closed notes with the exception of an 8 1/2" x 11" piece of paper filled with notes, programs, etc.
- When taking the exam, you may have with you pens and pencils, and your note sheet.
- You may not use a computer, calculator, tablet, smart watch, or other electronic device.
- Do not open this exam until instructed to do so.

Hunter College regards acts of academic dishonesty (e.g., plagiarism, cheating on examinations, obtaining unfair advantage, and falsification of records and official documents) as serious offenses against the values of intellectual honesty. The College is committed to enforcing the CUNY Policy on Academic Integrity and will pursue cases of academic dishonesty according to the Hunter College Academic Integrity Procedures.

I understand that all cases of academic dishonesty will be reported to the Dean of Students and will result in sanctions.

Name:

EmpID:

Email:

Signature:

1. (a) What will the following Python code print:

i.

```
s = "avram,henriette;dolciani,mary;rees,mina"
a = s[6:11]
print(a.upper())
```

Output:

ii.

```
names = s.split(’;’)
print(names[-1])
```

Output:

iii.

```
b,c = names[0],names[1]
print(b[:5])
```

Output:

iv.

```
for n in names:
    w = n.split(’,’)
    print(w[1],w[0])
```

Output:

(b) Consider the following shell commands:

```
$ ls
nyc.csv  p53.cpp  p54.cpp  p55.cpp  trees.csv
```

i. What is the output for:

```
$ ls *.cpp
```

Output:

ii. What is the output for:

```
$ ls *.cpp |  wc -l
```

Output:

iii. What is the output for:

```
$ mkdir ccProgs
$ echo "Created folder: ccProgs"
```

Output:

2. (a) For each row below containing a binary, decimal, and hexadecimal number, circle the **largest value** in the row (or "All Equal" if all three entries have the same value):

	Binary:	Decimal:	Hexadecimal:	All Equal
a)	11	10	9	*All Equal*
b)	111	5	5	*All Equal*
c)	101010	32	21	*All Equal*
d)	1000000	64	40	*All Equal*
e)	11111110	254	FF	*All Equal*

(b) Fill in the code below to make an image in which a pixel is white if it has an entry of 0 in the array `elevations`. Otherwise, the pixel should be colored blue.

```
# Takes elevation data of NYC and displays coastlines
import numpy as np
import matplotlib.pyplot as plt
elevations = np.loadtxt('elevationsNYC.txt')
#Base image size on shape (dimensions) of the elevations:
mapShape = elevations.shape + (3,)
floodMap = np.zeros(mapShape)

for row in range(mapShape[0]):
    for col in range(mapShape[1]):
```

```
#Save the image:
plt.imsave('floodMap.png', floodMap)
```

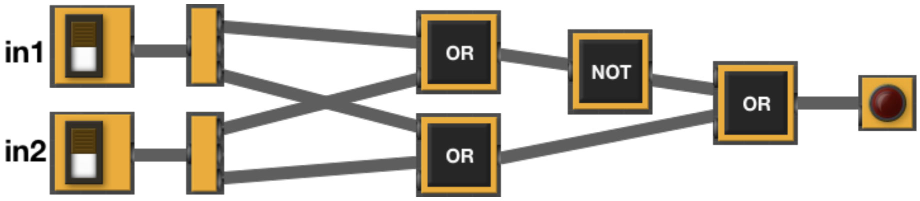
3. (a) What is the value (True/False):

i.
```
in1 = False
in2 = True
out = in1 and in2
```
out =

ii.
```
in1 = True
in2 = True
out = not in1 or (in2 and not in1)
```
out =

iii.
```
in1 = False
in2 = True and not in1
in3 = in1 and in2
out = in1 or not in3
```
out =

iv.



```
in1 = True
in2 = False
```
out =

(b) Design a circuit that implements the logical expression:

```
(in1 or (in1 and not in2)) or (in3 and not in3)
```

4. (a) For the following code:

```
def v3(anoop, madison):
    if  anoop >  madison:
        return madison
    else:
        return -1
```

```
def startV3(shelly):
    jack = 5
    rachael = 20
    alexandra = v1(jack+shelly,rachael)
    return alexandra
```

i. What are the formal parameters for `v3()`:

ii. What are the formal parameters for `startV3()`:

iii. What does `startV3(20)` return:

(b) Given the function definition:

```
def sorted(ls):
    for i in range(4):
        print(ls)
        for j in range(3):
            if ls[j] > ls[j+1]:
                 ls[j],ls[j+1] = ls[j+1],ls[j]
```

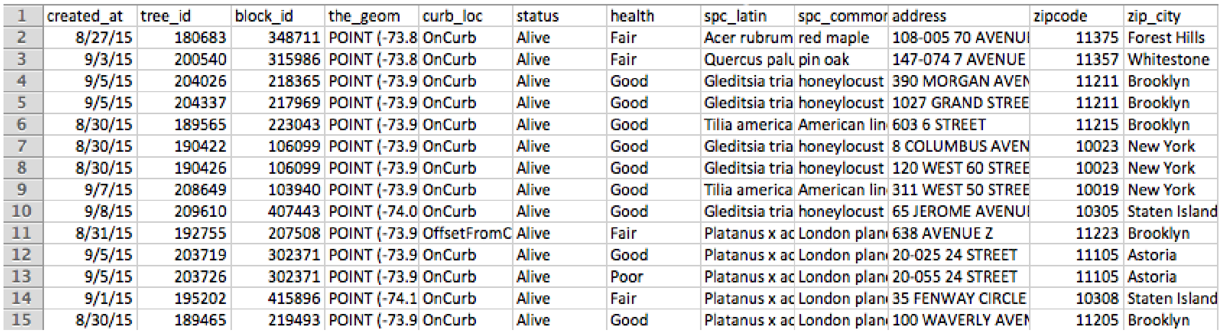
i. What is the output for `sorted([20,10,0,5])`?

ls[0]	ls[1]	ls[2]	ls[3]

ii. What is the output for `sorted(["Nicky","Maria","Ferdi","Andrey"])`?

ls[0]	ls[1]	ls[2]	ls[3]

5. Design an algorithm that prints out all the street trees in your zip code from the NYC Urban Forest OpenData. Specify the inputs and outputs for your algorithm and give the design in pseudocode. In your pseudocode, specify any libraries that you would need for your design.



1	created_at	tree_id	block_id	the_geom	curb_loc	status	health	spc_latin	spc_commor	address	zipcode	zip_city
2	8/27/15	180683	348711	POINT (-73.8	OnCurb	Alive	Fair	Acer rubrum	red maple	108-005 70 AVENUI	11375	Forest Hills
3	9/3/15	200540	315986	POINT (-73.8	OnCurb	Alive	Fair	Quercus palu	pin oak	147-074 7 AVENUE	11357	Whitestone
4	9/5/15	204026	218365	POINT (-73.9	OnCurb	Alive	Good	Gleditsia tria	honeylocust	390 MORGAN AVEN	11211	Brooklyn
5	9/5/15	204337	217969	POINT (-73.9	OnCurb	Alive	Good	Gleditsia tria	honeylocust	1027 GRAND STREE	11211	Brooklyn
6	8/30/15	189565	223043	POINT (-73.9	OnCurb	Alive	Good	Tilia america	American lin	603 6 STREET	11215	Brooklyn
7	8/30/15	190422	106099	POINT (-73.9	OnCurb	Alive	Good	Gleditsia tria	honeylocust	8 COLUMBUS AVEN	10023	New York
8	8/30/15	190426	106099	POINT (-73.9	OnCurb	Alive	Good	Gleditsia tria	honeylocust	120 WEST 60 STREE	10023	New York
9	9/7/15	208649	103940	POINT (-73.9	OnCurb	Alive	Good	Tilia america	American lin	311 WEST 50 STREE	10019	New York
10	9/8/15	209610	407443	POINT (-74.0	OnCurb	Alive	Good	Gleditsia tria	honeylocust	65 JEROME AVENUI	10305	Staten Island
11	8/31/15	192755	207508	POINT (-73.9	OffsetFromC	Alive	Fair	Platanus x ac	London plan	638 AVENUE Z	11223	Brooklyn
12	9/5/15	203719	302371	POINT (-73.9	OnCurb	Alive	Good	Platanus x ac	London plan	20-025 24 STREET	11105	Astoria
13	9/5/15	203726	302371	POINT (-73.9	OnCurb	Alive	Poor	Platanus x ac	London plan	20-055 24 STREET	11105	Astoria
14	9/1/15	195202	415896	POINT (-74.1	OnCurb	Alive	Fair	Platanus x ac	London plan	35 FENWAY CIRCLE	10308	Staten Island
15	8/30/15	189465	219493	POINT (-73.9	OnCurb	Alive	Good	Platanus x ac	London plan	100 WAVERLY AVEN	11205	Brooklyn

Input:

Output:

Process:

6. Fill in the Python program that will:
    - prompt the user for the name of a CSV file,
    - prompt the user for the name of a column in that CSV file,
    - print out the maximum value of the column, and
    - displays a bar plot of the column entered (with `"Year"` as the x-axis).

```
#P6,V3: prints max of a column in a CSV file & makes a plot

#Import the libraries for data frames and displaying images:



#Prompt user for file name:



#Prompt user for column name:



df = pd.read_csv(fileName)
#Compute maximum value of the column:



print("Maximum is ", M)

#Display a bar plot of "Year" vs. column entered by user:
```

7. Complete the following program, by writing the functions:
   - `getInput()`: returns the number of turtles the user entered,
   - `setUp()`: sets up a graphics window and turtle, and
   - `drawLines()`: repeat 10 times: n steps, turn left 60 degrees.

```
import turtle
def getInput():
     """
     Prompts for a whole number.
     Returns the number entered.
     """



def setUp():
     """
     Creates a graphics window and turtle.  Returns both.
     """



def drawLines(t,n):
     """
     Takes a turtle and n as input.
     Repeats 10 times: n steps, turn left 60 degrees.
     """



def main():
    n = getInput()  #get number of lines to be drawn
    w,t = setUp()   #sets up a graphics window and turtle
    drawLines(t,n)  #repeat 10 times: n steps, turn left 60 degrees
if __name__ == ’__main__’:
    main()
```

8. (a) What is the output for a run of this MIPS program:

```
#Loop through every other letter:
ADDI $sp, $sp, -6   # Set up stack
ADDI $t0, $zero, 65 # Start $t0 at 65 (A)
ADDI $s2, $zero, 75 # Use to test when you reach 75 (K)
SETUP: SB $t0, 0($sp) # Next letter in $t0
ADDI $sp, $sp, 1    # Increment the stack
ADDI $t0, $t0, 2    # Increment the letter
BEQ $t0, $s2, DONE  # Jump to done if $t0 == 75
J SETUP             # If not, jump back to SETUP for loop
DONE: ADDI $t0, $zero, 0 # Null (0) to terminate string
SB $t0, 0($sp)      # Add null to stack
ADDI $sp, $sp, -6   # Set up stack to print
ADDI $v0, $zero, 4  # 4 is for print string
ADDI $a0, $sp, 0    # Set $a0 to stack pointer for printing
syscall  # print to the log
```

Output:

(b) Indicate what modifications are needed to the MIPS program (repeated below) so that it prints out the first 10 upper case letters: `ABCDEFGHIJ` ?

```
#Loop through every other letter:
ADDI $sp, $sp, -6   # Set up stack
ADDI $t0, $zero, 65 # Start $t0 at 65 (A)
ADDI $s2, $zero, 75 # Use to test when you reach 75 (K)
SETUP: SB $t0, 0($sp) # Next letter in $t0
ADDI $sp, $sp, 1    # Increment the stack
ADDI $t0, $t0, 2    # Increment the letter
BEQ $t0, $s2, DONE  # Jump to done if $t0 == 75
J SETUP             # If not, jump back to SETUP for loop
DONE: ADDI $t0, $zero, 0 # Null (0) to terminate string
SB $t0, 0($sp)      # Add null to stack
ADDI $sp, $sp, -6   # Set up stack to print
ADDI $v0, $zero, 4  # 4 is for print string
ADDI $a0, $sp, 0    # Set $a0 to stack pointer for printing
syscall  # print to the log
```

9. What is the output of the following C++ programs?

(a)

```
//Lyrics by Lopez & Lopez
#include <iostream>
using namespace std;
int main()
{
  cout << "Let the storm rage ";
  cout << "on\nThe cold never ";
  cout << bothered me anyway";
  cout << endl;
  return(0);
}
```

Output:

(b)

```
//More Elsa
#include <iostream>
using namespace std;
int main()
{
  int count = 2;
  while (count > 0) {
    cout <<"Let it go, ";
    count--;
  }
  cout << "\nThat perfect girl ";
  cout << "is gone\n";
  return(0);
}
```

Output:

(c)

```
//Stars and srtipes
#include <iostream>
using namespace std;
int main()
{
  int i, j;
  for (i = 0; i < 5; i++)
  {
    for (j = 0; j < 5; j++)
       if (j % 2 == 0)
         cout << "*";
       else
         cout << "-";
    cout << endl;
  }
  return(0);
}
```

Output:

10. (a) Translate the following program into a **complete C++ program**:

```
#Python Loops, V3:
for i in range(0,101,2):
    print(i)
```

(b) Write a **complete C++ program** that asks the user for a whole number between -31 and 31 and prints out the number in "two's complement" notation, using the following algorithm:

i. Ask the user for a number, n.

ii. If the number is negative, print a 1 and let x = 32 + n.

iii. If the number is not negative, print a 0 and let x = n.

iv. Let b = 16.

v. While $b > 0.5$:

- If $x >= b$ then print 1, otherwise print 0
- Let x be the remainder of dividing x by b.
- Let b be b/2.