Row:	Seat:

Final Exam, Version 3
CSci 127: Introduction to Computer Science
Hunter College, City University of New York

23 May 2022

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, phone, earbuds, 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:								

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	*[NULL]*	32	20	*[SPACE]*	64	40	@	96	60	`
1	1	*[START OF HEADING]*	33	21	!	65	41	A	97	61	a
2	2	*[START OF TEXT]*	34	22	"	66	42	B	98	62	b
3	3	*[END OF TEXT]*	35	23	#	67	43	C	99	63	c
4	4	*[END OF TRANSMISSION]*	36	24	$	68	44	D	100	64	d
5	5	*[ENQUIRY]*	37	25	%	69	45	E	101	65	e
6	6	*[ACKNOWLEDGE]*	38	26	&	70	46	F	102	66	f
7	7	*[BELL]*	39	27	'	71	47	G	103	67	g
8	8	*[BACKSPACE]*	40	28	(	72	48	H	104	68	h
9	9	*[HORIZONTAL TAB]*	41	29	)	73	49	I	105	69	i
10	A	*[LINE FEED]*	42	2A	*	74	4A	J	106	6A	j
11	B	*[VERTICAL TAB]*	43	2B	+	75	4B	K	107	6B	k
12	C	*[FORM FEED]*	44	2C	,	76	4C	L	108	6C	l
13	D	*[CARRIAGE RETURN]*	45	2D	-	77	4D	M	109	6D	m
14	E	*[SHIFT OUT]*	46	2E	.	78	4E	N	110	6E	n
15	F	*[SHIFT IN]*	47	2F	/	79	4F	O	111	6F	o
16	10	*[DATA LINK ESCAPE]*	48	30	0	80	50	P	112	70	p
17	11	*[DEVICE CONTROL 1]*	49	31	1	81	51	Q	113	71	q
18	12	*[DEVICE CONTROL 2]*	50	32	2	82	52	R	114	72	r
19	13	*[DEVICE CONTROL 3]*	51	33	3	83	53	S	115	73	s
20	14	*[DEVICE CONTROL 4]*	52	34	4	84	54	T	116	74	t
21	15	*[NEGATIVE ACKNOWLEDGE]*	53	35	5	85	55	U	117	75	u
22	16	*[SYNCHRONOUS IDLE]*	54	36	6	86	56	V	118	76	v
23	17	*[ENG OF TRANS. BLOCK]*	55	37	7	87	57	W	119	77	w
24	18	*[CANCEL]*	56	38	8	88	58	X	120	78	x
25	19	*[END OF MEDIUM]*	57	39	9	89	59	Y	121	79	y
26	1A	*[SUBSTITUTE]*	58	3A	:	90	5A	Z	122	7A	z
27	1B	*[ESCAPE]*	59	3B	;	91	5B	[	123	7B	{
28	1C	*[FILE SEPARATOR]*	60	3C	<	92	5C	\	124	7C	|
29	1D	*[GROUP SEPARATOR]*	61	3D	=	93	5D	]	125	7D	}
30	1E	*[RECORD SEPARATOR]*	62	3E	>	94	5E	^	126	7E	~
31	1F	*[UNIT SEPARATOR]*	63	3F	?	95	5F	_	127	7F	*[DEL]*

(Image from wikipedia commons)

1. (a) Fill in the code below to produce the Output on the right:

```
workdays = "Monday?Tuesday?Wednesday?Thursday?"
summer_months = "*June*July*August*"
long_weekend = "Friday_Saturday_Sunday"
seasons = "+Spring+Summer+Fall+Winter"
```

i.
```
print( [          ], [          ])
```
Output:
```
Spring Tuesday
```

ii.
```
days = long_weekend[ [      ] ].split( [      ] )

print("Our weekend has", len( [      ] ),"days.")
```
Output:
```
Our weekend has 3 days.
```

iii.
```
for d in [          ]
    print( [          ] )
```
Output:
```
FRIDAY
SATURDAY
SUNDAY
```

(b) Consider the following shell commands:

```
$ pwd
/Users/guest
$ ls
bronx.png  circuit.txt   nand.txt  nyc.png    temp
```

i. What is the output for:
```
$ mkdir logic
$ mv *txt logic
$ ls
```
Output:

ii. What is the output for:
```
$ cd logic
$ ls
```
Output:

iii. What is the output for:
```
$ cd ../temp
$ pwd
```
Output:

2. (a) Select the correct option.

   i. What color is tina after this command? `tina.color(1.0,0.0,1.0)`
   ☐ black ☐ red ☐ white ☐ gray ☐ purple

   ii. Select the SMALLEST Binary number:
   ☐ 1011 ☐ 1101 ☐ 1111 ☐ 1010 ☐ 1110

   iii. Select the LARGEST Hexadecimal number:
   ☐ AA ☐ BA ☐ DC ☐ CC ☐ CD

   iv. What is the binary number equivalent to decimal 14?
   ☐ 1011 ☐ 1101 ☐ 1111 ☐ 1010 ☐ 1110

   v. What is the hexadecimal number equivalent to decimal 170?
   ☐ AA ☐ BA ☐ DC ☐ CC ☐ CD

   (b) Fill in the code to produce the Output on the right:

```
nums = [ 23, 45, 76, 23, 98, 45 , 11, 4, 33, 29, 5, 66]
```

i.
```
for i in range( [ ] , [ ] ):
    print(nums[i], end=" ")
```

Output:
```
23 98 45 11 4 33 29
```

ii.
```
for j in range( [ ] , [ ] , [ ] ):
    print(nums[j], end=" ")
```

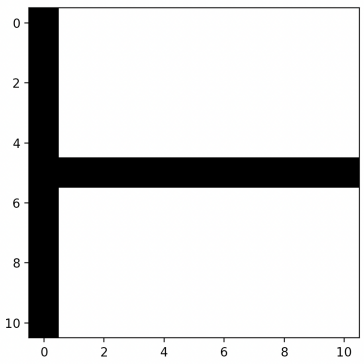
Output:
```
45 45 29
```

Output:

iii.
```
import numpy as np
import matplotlib.pyplot as plt
img = np.ones( (11,11,3) )

img[ [ ] , [ ] , :]  = 0 # black row

img[ [ ] , [ ] , :]  = 0 # black column
plt.imshow(img)
plt.show()
```

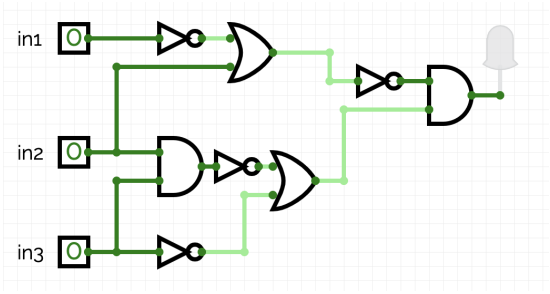


3. (a) What is the value (True/False):

i.
```
in1 = False
in2 = False
out =  (not in1 and in2) or (not in1 or in2)
```
☐ True ☐ False

ii.
```
in1 = True
in2 = False
in3 = ( not in1 ) or ( not in2 )
out = (not in1 or not in2) and (not in2 and in3)
```
☐ True ☐ False

iii.




```
in1 = True
in2 = False
in3 = True
```
☐ True ☐ False

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

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

4. Consider the following functions:

```
def whoop(n, smile):
   for i in range(1,n+1):
      screech(i, smile)
      print()
```

```
def screech(i, smirk):
    for j in range(i):
      print(smirk, end='  ')


def main():
      whoop(3, '^_^')
```

(a) What are the formal parameters for `screech()`?

(b) What are the actual parameters for `whoop()`?

(c) How many calls are made to `screech()` after calling `main()`?

(d) What is the output after calling `main()`?

Output:

5. Design an algorithm that asks the user for the name of an image file and the quarter ['TL', 'TR', 'BL', 'BR'] they wish to "black-out", where 'TL' stands for Top Left, 'BL' stands for Bottom Right and so on. The algorithm then saves a new image where that quarter of the image is black. The name of the new image is 'XXblack.png' where XX is replaced by one of ['TL', 'TR', 'BL', 'BR'] that the user entered. You must write detailed **pseudocode** as a precise list of steps that completely and precisely describe the algorithm.

Libraries (if any):

Input:

Output:

Principal Mechanisms (select all that apply):

☐ Single Loop ☐ Nested Loop ☐ Conditional (if/else) statement
☐ Indexing / Slicing ☐ `split()` ☐ `input()`

Process (as a concise and precise LIST OF STEPS / pseudocode):
(Assume libraries, if any, have already been imported.)

6. Consider `boeing.csv` from the "Military Stocks during Russia-Ukraine War " dataset from kaggle, reporting the Boeing Company's stock prices (in USD $) from January 2010 to May 2022 **Each row in the dataset corresponds to the stock values for one day of trading**. A snapshot of the data is given in the image below:

Date	Open	High	Low	Close	Volume
2010-01-04	55.720001	56.389999	54.799999	56.180000	6186700
2010-01-05	56.250000	58.279999	56.000000	58.020000	8867800
2010-01-06	58.230000	59.990002	57.880001	59.779999	8836500
2010-01-07	59.509998	62.310001	59.020000	62.200001	14379100
■ ■ ■					
2022-04-28	156.610001	156.789993	149.000000	154.220001	13518800
2022-04-29	153.440002	157.029999	148.520004	148.839996	10880300
2022-05-02	148.020004	149.449997	143.380005	148.610001	12390700

Fill in the Python program below:

```
#Import the libraries for plotting and data frames

[                                                    ]

#Prompt user for input file name:

fin = [                                              ]

#Read input data into data frame:

boeing = [                                           ]

#Print the average opening value

print( [                                             ] )

#Print the lowest closing value

print( [                                             ] )

#Create a new column called "Range" that computes
#the difference between the highest and lowest value of the stock

boeing["Range"] [                                    ]

#Plot the newly computed range against the date

boeing.[                                             ]

plt.show()
```

7. Fill in the following functions that are part of a program that averages the color in an image:

   - `getData()`: asks the user for the name of an image file and returns a numpy array of the pixels
   - `getAvg()`: computes and returns the average (r, g, b) values in img
   - `avgImg()`: returns an image of size rows, cols, with color r, g, b

```
import numpy as np
import matplotlib.pyplot as plt
def getData():
    """
    Asks the user for the name of an image file
    Returns a numpy array of the pixels
    """
```

```
def getAvg(img):
    """
    Computes and returns the average (r, g, b) values in img
    """
```

```
def avgImg(rows, cols, r, g, b):
    """
    Creates and returns an image of size rows, cols, with color r, g, b

    """
```

8. (a) What is printed by the MIPS program below:

Output:

(b) Modify the program to print out "ZYXWV". Shade in the box for each line that needs to be changed and rewrite the instruction below, or add instructions where necessary.

```
☐ ADDI $sp, $sp, -10      # Set up stack

☐ ADDI $s3, $zero, 1      # Store 1 in a register

☐ ADDI $t0, $zero, 90      # Set $t0 at 90 (Z)

☐ ADDI $s2, $zero, 10      # Use to test when you reach 10

☐ SETUP: SB $t0, 0($sp)    # Next letter in $t0

☐ ADDI $sp, $sp, 1         # Increment the stack

☐ ADDI $s3, $s3, 1         # Increment the counter by 1

☐ BEQ $s3, $s2, DONE       # Jump to done if $s3 == 10

☐ 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, -9      # 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. Fill in the C++ programs below to produce the Output on the right.

(a)

```
#include <iostream>
using namespace std;
int main()
{
    for( [      ] ; i <=15; [      ] ){
       cout << i-1 << endl;
    }
    return 0;
}
```

Output:

```
3
5
7
9
11
13
```

(b)

```
#include <iostream>
using namespace std;
int main()
{
     int n=12, m=-5;

    while(n+m [      ] ){
        cout << n << " " << m << endl;
        n-=2;
        m++;
    }
    return 0;
}
```

Output:

```
12 -5
10 -4
8 -3
6 -2
4 -1
2 0
0 1
```

(c)

```
#include <iostream>
using namespace std;
int main(){
for ( [                              ] ){
    for( [                              ] ){
            cout << i << j-i << " ";
        }
        cout << endl;
    }
    return 0;
}
```

Output:

```
88 87 86 85 84 83 82 81 80
77 76 75 74 73 72 71 70
66 65 64 63 62 61 60
55 54 53 52 51 50
44 43 42 41 40
33 32 31 30
```

10. (a) Write a **complete C++ program** that repeatedly asks the user for two integers until their sum is even, then it outputs the sum:

```
//include library and namespace
```

```
//main function signature
```

```
{
  //variable initialization
```

```
  //repeatedly ask for two integers until sum is even
```

```
  //output sum
```

```
    return 0;
}
```

(b) Write a **complete C++ program** that asks the user for an amount and computes the number of years it takes to triple the amount, if it is subject to an increase of 5% each year.

```
//include library and namespace

//main function signature

{
  //declare variables

  //obtain input

  //compute number of years it takes to triple amount at 5% yearly increase

  //Output number of years and tripled amount

   return 0;
}
```

SCRATCH PAPER (page left intentionally blank)

SCRATCH PAPER (page left intentionally blank)