Row:	Seat:

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

Fall 2025

Exam Rules

- Show all your work. Your grade will be based on the work shown.
- You may have pens, pencils and one 8 1/2" x 11" reference sheet filled with notes. No other materials are allowed.
- No phones, computers, tablets, calculators, watches, smart glasses, smart pencils, earpods, or other electronic devices are allowed.
- All electronic devices must be turned off and stored in your bag. If you are not able to turn off the Bluetooth/Wifi on your device, put it in your bag at the front of the room.
- **Do not open this exam until instructed to do so.**

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) Fill in the code below to produce the output on the right:

```
day_string = "Monday,Tuesday,Wednesday,Thursday,Friday,Saturday,Sunday"
```

	Python Code:	**Output:**
i.	`print(day_string[ ])`	`Mon`

ii.	`counts = {}`
	`for c in day_string.lower():`
	`  if :`
	`    counts[c] = counts[c]+1`
	`  else:`
	`    counts[c]=1`
	`print("t appears", counts['t'], "times.")`
		`t appears 3 times.`

iii.	`days_list = day_string.`
	`print(days_list[-1].upper())`
		`SUNDAY`

iv.	`short_days = [ for d in days_list]`
	`print(short_days[-2:])`
		`['Sa', 'Su']`

v.	`weekdays = days_list[ ]`
	`print(len(weekdays), "weekdays.")`
		`5 weekdays.`

(b) The commands below are **run sequentially**, what is the output after each has run:

```
$ ls
baruch.png    ccny.mp4       hunter.png    queens.png
$ pwd
/tmp/mock
```

i.
```
$ mkdir pix
$ mv *.png pix
$ ls
```
Output:

ii.
```
$ ls | grep cc
```
Output:

iii.
```
$ cd pix
$ echo "Picture folder:"
$ pwd
```
Output:

iv.
```
$ cp ../ccny.mp4 ccny2.mp4
$ ls | wc -l
```
Output:

2. (a) Check all that apply:

   i. What color is `tess` after this command? `tess.color("#ABABAB")`
   ☐ black ☐ red ☐ white ☐ gray ☐ green

   ii. Select all the **even** binary numbers:
   ☐ 1011 ☐ 1101 ☐ 0111 ☐ 1010 ☐ 1110

   iii. Select the hexadecimal number **larger than 160**:
   ☐ AA ☐ 11 ☐ FF ☐ 55 ☐ DD

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

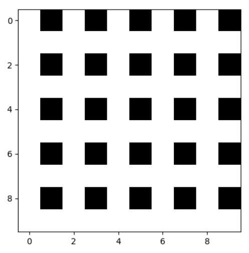
   i.
```
nums = [ 1, 4, 9, 16, 25, 36, 49, 64]

for i in range( ☐ , ☐ ):
    print(nums[i], end=" ")
```
   Output:
```
4 9 16 25
```

   ii.
```
import numpy as np
import matplotlib.pyplot as plt
img = np.ones( (10,10,3) )

img[ ☐ :: ☐ , 1::2, :] = 0

plt.imshow(img)
plt.show()
```
   Output:



   (c) Consider the code:
```
bin_string = input('Enter a binary number: ")
dec_num = 0
for c in bin_string:
    dec_num = dec_num * 2
    if c == '1':
        dec_num++
print(dec_num)
```

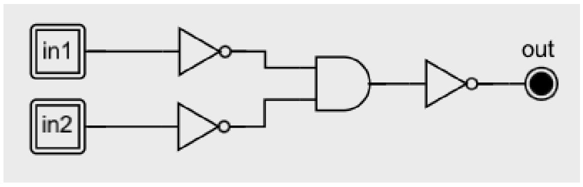
   i. **Circle** the code above and mark line with **(i)** that caused this error:
```
    bin_string = input('Enter a binary number: ")
                                          ^
SyntaxError: unterminated string literal (detected at line 1)
```
   Write the code that would fix the error:

   ii. **Box** the code above and mark line with **(ii)** that caused this error:
```
    dec_num++
             ^
SyntaxError: invalid syntax
```
   Write the code that would fix the error:

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

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

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

   iii. 

out =

```
in1 = False
in2 = False
```

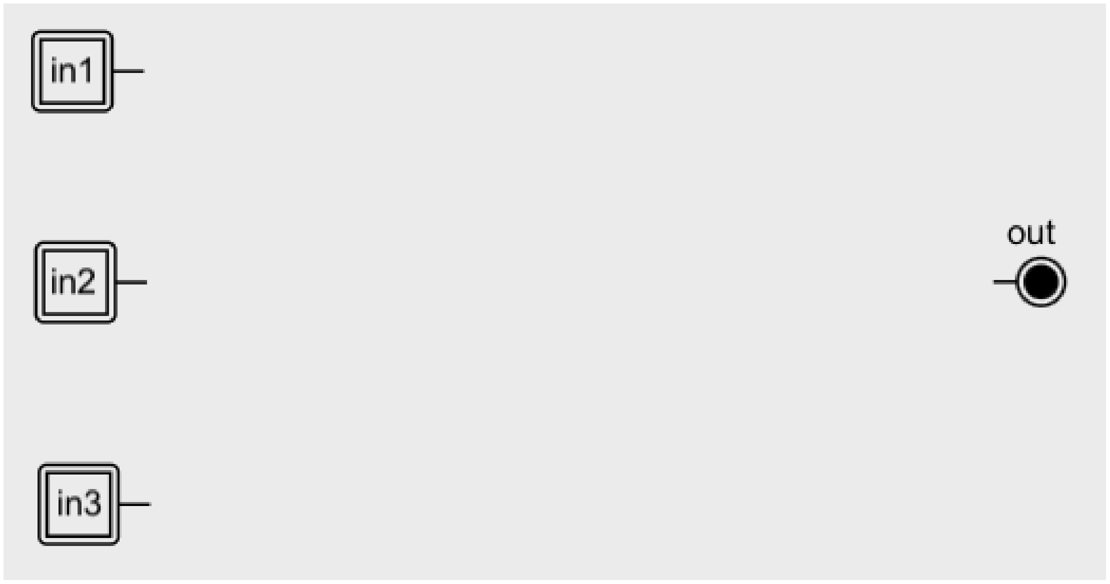
(b) Fill in the values to yield the output:

   i.
```
in1 = 
in2 = 

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

(c) Design a circuit that **exactly implements** the logical expression:

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



4. (a) Draw the output for the function calls:

i. `ramble(tess,5)`

ii. `ramble(tess,100)`

```
 1  import turtle
 2  tess = turtle.Turtle()
 3  tess.shape('turtle')
 4
 5  def ramble(t, len):
 6      if len <= 10:
 7          t.stamp()
 8      elif len%2 == 0:
 9          t.left(90)
10          t.forward(len)
11          ramble(t, len//2)
12      else:
13          t.right(90)
14          t.forward(len)
15          ramble(t, len//2)
```

(b) What are the formal parameters for `ramble()`:

(c) If you call `ramble(tess,5)`, which branches of the function are tested (check all that apply):

- ☐ The block of code at Line 7.
- ☐ The block of code at Lines 9-11.
- ☐ The block of code at Lines 13-15.
- ☐ None of these blocks of code (lines 7, 9-11, 13-15) are visited from this invocation (call).

(d) If you call `ramble(tess,100)`, which branches of the function are tested (check all that apply):

- ☐ The block of code at Line 7.
- ☐ The block of code at Lines 9-11.
- ☐ The block of code at Lines 13-15.
- ☐ None of these blocks of code (lines 7, 9-11, 13-15) are visited from this invocation (call).

5. Write a function `both_cases()` that takes a string, removes all punctuation and spacing, and returns the characters that occurs both in upper and lower case in the string. For example:

   `both_cases("A man, a plan, a canal: Panama")`

   would return `A` and `P` since both `A` and `P` occur both as upper and lower case letters in the inputted string.

Libraries:	
Input:	
Output:	

Design Pattern:

☐ Accumulator ☐ Max/Min ☐ Finding Duplicates ☐ Searching

Principal Mechanisms (select all that apply):

☐ Loop ☐ Conditional (if/else) ☐ Recursion
☐ Indexing/slicing ☐ Dictionary ☐ List Comprehension ☐ Regular Expressions

Process (as a concise and precise LIST OF STEPS / pseudocode):

(Assume libraries have already been imported.)

6. Fill in the Python code below for the function, `animate()`, that animates a hurricane tracker using the Turtle library.

```
def animate(t,lat,lon,wind):
    """
    Takes a turtle, a location, and windspeed.
    Moves the turtle to the location, adjusts color \& pensize based on windspeed
    """
    #Lift the pen up:

    #Move the turtle to (lat, lon) location:

    #Check if wind stronger than 156. If so, change pen size to 5 and color to red:

    #Else, check if wind > 129. If so, change pen size to 4 and color to orange:

    #Else, check if wind > 110. If so, change pen size to 3 and color to yellow:

    #Else, check if wind > 95. If so, change pen size to 2 and color to green:

    #Else, check if wind > 73. If so, change pen size to 2 and color to blue:

    #Else change pen size to 1 and color to white:
```

7. Write a **complete Python program** that creates a DataFrame. Your program should ask the user for:

   - A list of place names,
   - A list of populations, and
   - A name for the output (CSV) file.

   Your program should filter the DataFrame to contain only places with populations larger than 100000 and save the resulting DataFrame to the specified CSV file.

   Hint: Create a dictionary from the inputted lists and then select rows with large populations. To cast an entire column of a DataFrame to be an integer, the following command is useful:

   ```
   df['population'] = df['population'].astype(int)
   ```

8. (a) Consider the following MIPS program:

```
ADDI $s0, $zero, 10
ADDI $s1, $zero, 3
SUB $s2, $s1, $s0
ADD $s3, $s1, $s0
```

After the program runs, what is the value stored in:

$s0 register	$s1 register	$s2 register	$s3 register

(b) Consider the MIPS code:

```
ADDI $sp, $sp, -5
ADDI $t0, $zero, 48
ADDI $s2, $zero, 60
SETUP: SB $t0, 0($sp)
ADDI $sp, $sp, 1
ADDI $t0, $t0, 3
BEQ $t0, $s2, DONE
J SETUP
DONE: ADDI $t0, $zero, 0
SB $t0, 0($sp)
ADDI $sp, $sp, -4
ADDI $v0, $zero, 4
ADDI $a0, $sp, 0
syscall
```

i) How many characters are printed?	
ii) What is the first character printed?	
iii) What is the whole message printed?	
iv) Detail the changes needed to print only the first half of the message:	

9. (a) What is the output:

```
#include <iostream>
using namespace std;
int main()
{
  cout << "Motto:"
       << endl << "Mihi ";
  cout << "Cura,\nFuturi\n";
}
```

Output:

(b) Fill in the missing code to yield the output:

```
#include <iostream>
using namespace std;
int main()
{
    int myst = 5, quest = 10;
    while ((myst < 15) && quest > 0 )
    {
        cout << myst << "\t" << quest << endl;



    }
    return 0;
}
```

Output:

```
5   10
6   8
7   6
8   4
9   2
```

(c) What is the output:

```
#include <iostream>
using namespace std;
int main()
{
    for (int i = 1; i <= 5; i += 2)
    {
        for (int j = 0; j < (5 - i)/2; j++)
            cout << "|";
        for (int j = 0; j < i; j++)
            cout << "#";
        for (int j = 0; j < (5 - i)/2; j++)
            cout << "|";
        cout << endl;
    }
    return 0;
}
```

Output:

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

Python program:

```
for i in range(100,0,-5)
    print(i)
```

C++ program:

(b) Write a C++ program that asks the user for the starting population and prints out the yearly population until it reaches 1000. Each year the population doubles in size.

A sample run:

```
Please enter the starting population: 50
Year 0  50
Year 1  100
Year 2  200
Year 3  400
Year 4  800
```

CSCI 127 Reference Sheet, Fall 2025

Turtles: Let `t` be a turtle.

Function	Description
`t=Turtle.turtle()`	Creates turtle `t`.
`t.forward(x)`	Moves `t` forward `x` steps.
`t.backward(x)`	Moves `t` backward `x` steps.
`t.left(x)`/`t.right(x)`	Turns `t` left/right `x` degrees.
`t.penup()`/`t.pendown()`	Lifts `t`'s pen up/down.
`t.stamp()`	Stamps at `t`'s current location.
`t.goto(x,y)`	Moves `t` to `(x,y)`.

String Methods: Let `s` be a string.

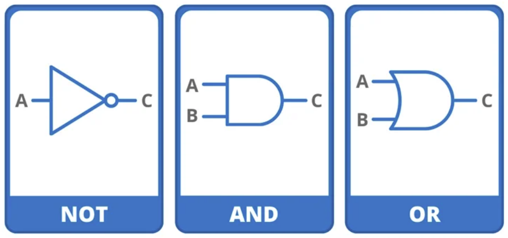
Function	Description
`len(s)`	Returns the length of `s`.
`s.lower()`	Returns `s` as lower case characters.
`s.upper()`	Returns `s` as upper case characters.
`s.count(t)`	Returns count of t in s.
`s.find(t)`	Returns index of t in s (-1 not found).
`s.split(d)`	Splits `s` into list of strings on `d`.
`s.join[lst]`	Joins `lst` into a string, by `s`.
`s[i:j]`	Substring (slice) of s: from i to j-1.
`ord(c)`	Returns Unicode/ASCII of `c`.
`chr(i)`	Returns character of `i`.

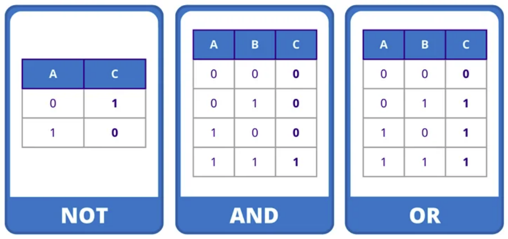
Containers: Lists, Ranges & Dictionaries.

Function	Description
`l = []`	Creates an empty list.
`l = [a,b,c]`	List with 3 elements.
`l.append(elt)`	Adds `elt` to end of list.
`l[i]`	Access element at index `i`.
`range(start,stop,step)`	Range object from `start` to `stop-1`, by `step`.
`zip(l1,l2)`	Combines `l1` & `l2` pairwise.
`[x*x for x in l1]`	List of `l1`'s elements squared. (using list comprehension).
`d = {}`	Creates an empty dictionary.
`d = {k1:v2,k2:v2}`	Dictionary of key/value pairs.
`d[k] = v`	Adds `k:v` to dictionary.
`d[k]`	Access value at key `k`.
`k in d`	Checks if key is in dictionary.
`d.keys() / d.values()`	Returns keys/values of `d`.

Functions:

Function	Description
def fname(x,y):	Defines function, `fname`, with
`command1`	(formal) input parameters, `x` and `y`.
`command2...`	Body of function indented.
`return(v)`	Returns value `v`.
`c = fname(a,b)`	Calls/invokes `fname` with (actual) parameters `a` & `b`, returns to `c`.





NOT

A	C
0	1
1	0

AND

A	B	C
0	0	0
0	1	0
1	0	0
1	1	1

OR

A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

(from truthtablegen.com)

numpy: Let `np` be the `numpy` package.

Function	Description
`arr_z = np.zeros((10,20,3))`	Sets up array for 10x20 black image.
`arr_1 = np.ones((10,20,z))`	Sets up array for 10x20 white image.
`arr[start:stop:step]`	Slice from `start` to `stop-1` by `step`.
`arr = plt.imread('image.png')`	Read in an image.
`plt.imshow(arr)`	Show `arr` as image.
`plt.show()`	
`plt.imsave('image.png', arr)`	Save an array to file.

Pandas: Let `pd` the Pandas package, `df` be a DataFrame, & `s` a Series.

Function	Description
`pd.read_csv(fn)`	Returns a DataFrame with file `fn`.
`pd.DataFrame(d)`	Returns DataFrame from dictionary `d`.
`df.to_csv(fn)`	Writes `df` to `fn`.
`df[col]`	Returns `col` column as a Series.
`df[[col1,col2]]`	Returns DataFrame with `col1` & `col2`.
`df.columns`	List of column names of `df`.
`df.head(n)`/`df.tail(n)`	First/last `n` lines of `df`.
`df.plot(x=col)`	Returns a figure with `col` as x-axis.
`fig.savefig(fn)`	Writes `fig` to `fn`.
`s.min()`/`s.max()`/`s.mean()`	Returns min/max/average of `s`.
`s.value_counts()`	Counts # times each value occurs.
`df.groupby(col)`	Groups `df` by values in `col`.

Plotly Express: Let `px` be the Plotly Express package.

Function	Description
longitude	Degrees east/west from -180 to 180.
latitude	Degrees north/south from -90 to 90.
px.scatter_geo(df,...)	Returns outline map as fig. Keywords args: `lon,lat,size,hover_name,projection,title`.
px.scatter_map(df,...)	Returns tiled map as fig. Keywords args: `lon,lat,size,hover_name,title,zoom`.
fig.show()	Displays map on browser.
fig.write_html(fn)	Writes `fig` to `fn`.

MIPS: Let `rs`, `rt`, & `rd` be registers.

Function	Description
`ADD rd, rs, rt`	Adds values of rs and rt and stores in rd.
`ADDI rd, rs, imm`	Adds values of rs and imm and stores in rd.
`SUB rd, rs, rt`	Subtracts values of rs and rt and stores in rd.
`BEQ rs, rt, target`	If registers `rs == rt`, jump to `target`.
`JUMP target`	Jump to target.

UNIX:

Function	Description
`ls` / `ls -l` / `ls *.py`	Lists files / lists long / lists matching pattern.
`cp x y` / `mv x y`	Copies/renames file x to file y.
`pwd`	Prints path to current directory.
`mkdir x`	Creates directory called `x`.
`cd ../` / `cd /usr/bin`	Changes directory via relative/absolute path.
`echo "message"`	Displays message
`ls|wc -c` / `ls|grep pat`	Uses pipes to count # of files/match `pat`

C++:

Function	Description
`#include <iostream>`	Includes library with `cin`/`cout`.
`using namespace std;`	Use standard names w/o `std::`.
`int main() {...}`	Function definition.
`int x;`	Declares variable `x` to be an integer.
`float y;`	Declares variable `y` to be a float.
`cin >> x;`	Reads input into `x`.
`cout << x;`	Prints `x`.
`for (i=0; i<10; i++){...}`	Basic for-loop.
`while (logicalExpression){...}`	Basic while-loop.
`return(v);`	Returns value `v`.

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]

(From wikipedia commons)