Row:	Seat:

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

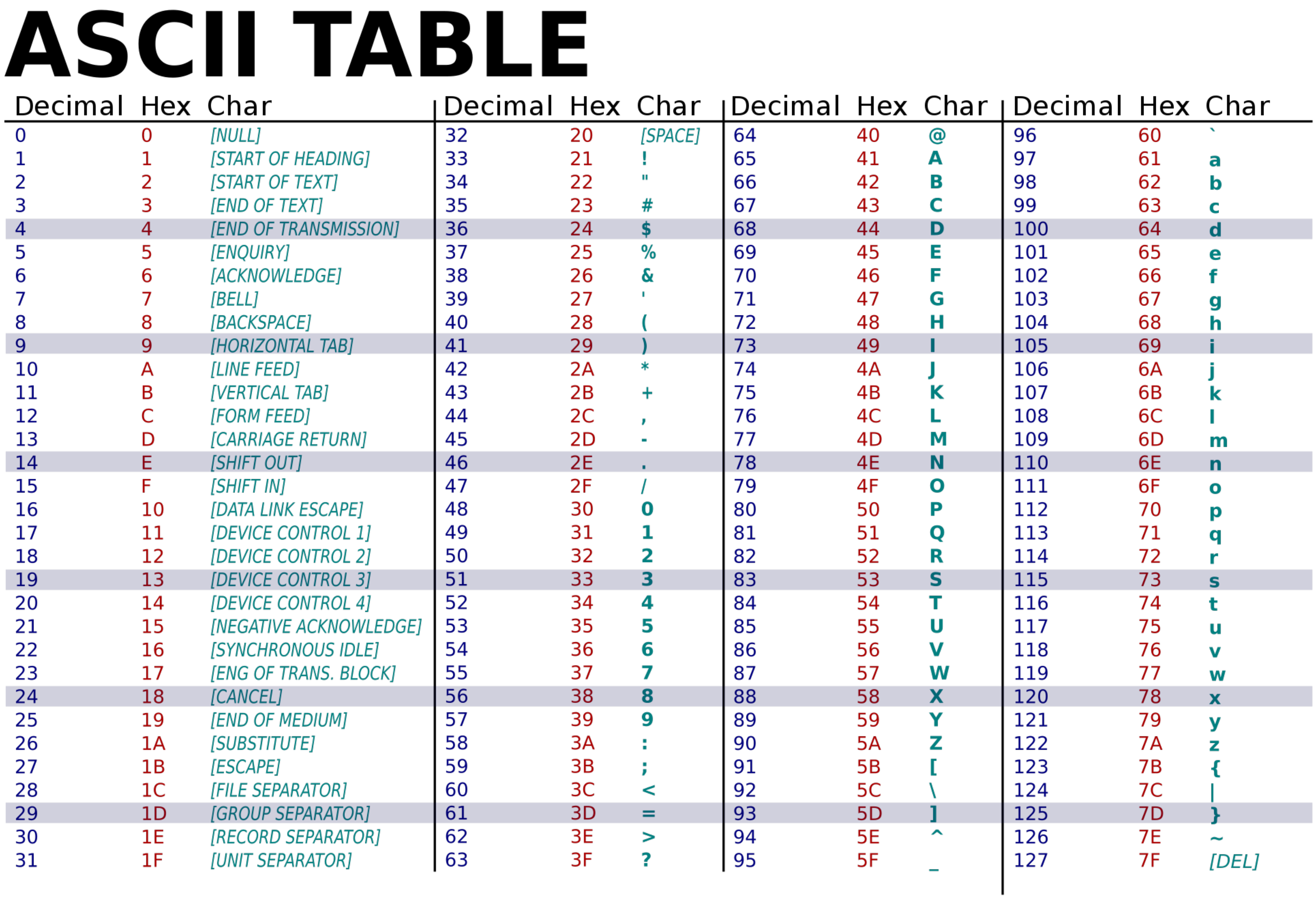
17 May 2022

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) What will the following Python code print:

   i.

   ```
   a = "Jan&Feb&Mar&Apr&May&Jun"
   print(a.count("&"))
   ```

   Output:

   ii.

   ```
   b = a.split("&")
   print(b[0])
   ```

   Output:

   iii.

   ```
   mo = b[-1].upper()
   print(mo)
   ```

   Output:

   iv.

   ```
   for c in mo:
       print(c.lower())
   ```

   Output:

   (b) Consider the following shell commands:

   ```
   $ ls -l
   -rw-r--r--@ 1 ligorio  staff      5308 Mar 21 14:38 quizzes.html
   -rw-r--r--  1 ligorio  staff        413 Apr 20 18:57 zoneDist.csv
   -rw-r--r--@ 1 ligorio  staff      519 Apr 22 15:14 zoneMap.py
   -rw-r--r--  1 ligorio  staff  16455174 Mar 20 19:02 zoning2.html
   -rw-r--r--  1 ligorio  staff  17343896 Mar 20 18:58 zoningIDS.json
   ```

   i. What is the output for:

   ```
   $ ls *zz*
   ```

   Output:

   ii. What is the output for:

   ```
   $ ls -l | grep "Apr"
   ```

   Output:

   iii. What is the output for:

   ```
   $ ls -l | grep "Apr" | wc -l
   ```

   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)	10	2	2	*All Equal*
b)	1100	12	C	*All Equal*
c)	10010	18	12	*All Equal*
d)	100000	34	19	*All Equal*
e)	1111110	250	FE	*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 green.

```
# 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)
```

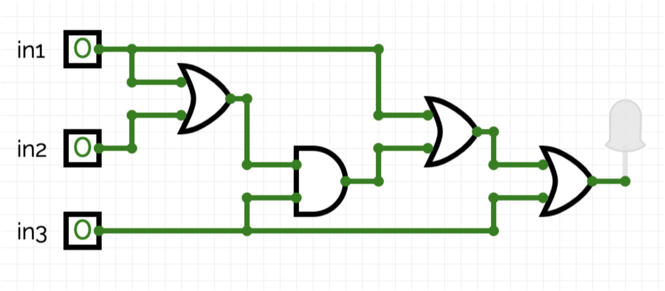
3. (a) What is the value (True/False):

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

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

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

iv.



```
in1 = False
in2 = True
in3 = False
```
out = □

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

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

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

```
import turtle

def mystery1(tess, x, y):
    for i in range(2):
        tess.forward(x)
        tess.left(90)
        tess.forward(y)
        tess.left(90)

def mystery2(tina, s):
    mystery1(tina, s, s)


taj = turtle.Turtle()
```

i. `mystery1(taj, 100, 20)`

ii. `mystery2(taj, 100)`

(b) Given the function definitions:

```
def enigma(n):
    for i in range(1,n+1):
        help(i)
        print()
```

```
def help(x):
    for j in range(x):
        print((x+j)*2,end=' ')
```

i. What is the output for `enigma(5)`?

5. Design an algorithm that asks the user for the name of a text file containing a grid of numbers and loads it into a 2D array of integers (think like an image without the color channel), then outputs the index `(row, col)` of the SMALLEST number in the array.

   Libraries:

   Input:

   Output:

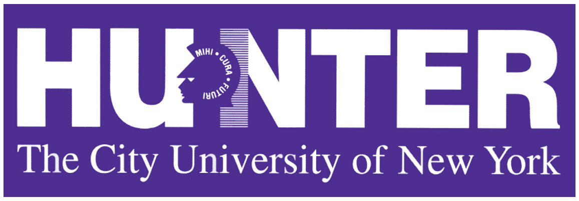
   Design Pattern:
   □ Search □ Find Min □ Find Max □ Find All

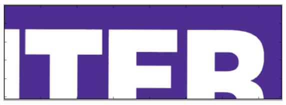
   Principal Mechanisms (select all that apply):
   □ Single Loop □ Nested Loop □ Conditional (if/else) statement
   □ Indexing / Slicing □ `split()` □ `input()`

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

6. Write a **complete Python program** that asks the user for the name of a .png (image) file and displays the upper right quarter of the image.

   For example if the image is `hunterLogo.png` (left), the displayed image would be (right):





7. Fill in the following functions that are part of a program that maps GIS data from NYC OpenData CSV files:

   - `getData()`: asks the user for the name of the CSV and returns a DataFrame of the contents.
   - `getLocale()`: asks the user for latitude and longitude of the user's current location and returns those floating points numbers, and
   - `computeDist()`: computes the squared distance between two points (x1,y1) and (x2,y2):

   $$(x1 - x2)^2 + (y1 - y2)^2$$

```
import pandas as pd
def getData():
     """
     Asks the user for the name of the CSV and
     Returns a dataframe of the contents.
     """
```

```
def getLocale():
     """
     Asks the user for latitude and longitude of the user's current location and
     Returns those floating points numbers.
     """
```

```
def computeDist(x1,y1,x2,y2):
     """
     Computes the squared distance between two points (x1,y1) and (x2,y2) and
     Returns (x1-x2)^2 + (y1-y2)^2
     """
```

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

Output:

(b) Modify the program to print out 99 copies of the character '!'. Shade in the box for each line that needs to be changed and rewrite the instruction below.

```
□ ADDI $sp, $sp, -6        # Set up stack

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

□ ADDI $t0, $zero, 33      # Set $t0 at 33 (!)

□ ADDI $s2, $zero, 5       # Use to test when you reach 5

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

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

□ SUB $s2, $s2, $s3        # Decrease the counter by 1

□ BEQ $s2, $zero, DONE     # Jump to done if $s0 == 0

□ 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, -5        # 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(int i = 0; i <=30; □ ){
       cout << i*2  << endl;
    }
    return 0;
}
```

Output:

```
0
20
40
60
```

(b)

```
#include <iostream>
using namespace std;
int main()
{
    int count = 5;
    int num = 2;

    while(count □ && num □ ){
        cout << count << " " << num << endl;
        count -=1;
        if(count % 2 == 0)
            num -=1;
    }
    return 0;
}
```

Output:

```
5 2
4 1
3 1
2 0
1 0
```

(c)

```
#include <iostream>
using namespace std;
int main(){

    for (int i = 5; □ ; i--){
         cout << "Still counting!" << endl;
    }
    return 0;
}
```

Output:

```
Still counting!
Still counting!
Still counting!
Still counting!
Still counting!
Still counting!
Still counting!
Still counting!
Still counting!
Still counting!
```

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

```
#Python Loops, V3:
for i in range(0,50,5):
    print(i)
```

(b) Write a **complete C++ program** to compute the ticket price to enter the Museum of Natural History. Your program must ask the user for their age and print "Child: $12.50" if the age entered is 12 or less, "Adult: $22.00" if the age entered is less than 65, and "Senior: $17.00" otherwise.