Row:	Seat:

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

May 19, 2026

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.**

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:

```
s = "the_City_University_of_New_York"
num = s.count("_")
print('Number is', num)
words = s.split("_")
print('words[1] is', words[1].upper())
actual=[w for w in words if ord(w[0])<91]
print(actual)
short = ""
for word in actual:
    print(word[0], word)
    short = short + word[0]
print("Short version: ", short)
print("Of", words[-2:])
```

Output:

(b) Consider the following shell commands:

```
$ ls
p1.py  p2.py  p50.cpp  photos  world_map.html
$ pwd
/Users/csguest
```

Assuming the commands below are run sequentially, what is the output after each has run:

i.
```
$ mv world_map.html photos
$ ls
```

Output:

ii.
```
$ touch p3.py
$ ls p* | wc -l
```

Output:

iii.
```
$ mkdir hw
$ mv p*.py hw
$ ls
```

Output:

iv.
```
$ cd hw
$ echo "hw:"
$ pwd
```

Output:

2. (a) Check **all that apply**:

   i. What color is the turtle `tess` after this command? `tess.color("#00CD00")`
   ☐ black ☐ white ☐ red ☐ blue ☐ None of these

   ii. Select all the binary numbers **smaller than 8 decimal**:
   ☐ 1001 ☐ 1101 ☐ 0111 ☐ 1010 ☐ 0110

   iii. Select all the **even hexadecimal numbers**:
   ☐ A ☐ B ☐ 90 ☐ 99 ☐ FF

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

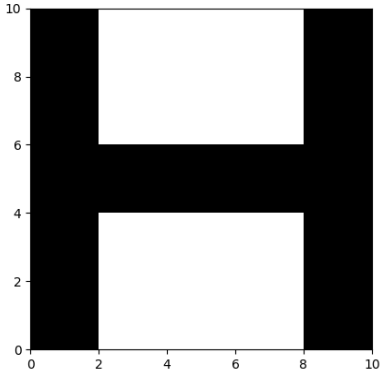
```
import matplotlib.pyplot as plt
import numpy as np
#Set background color to white:

logo = [          ] ( (10,10,3) )
#Draw letter in black:
logo[:,0:2,:] = 0

logo [:, [      ],:]  = 0

logo [ [      ],:,:]  = 0
plt.imshow(logo)
plt.show()
```

Output:



   (c) For each error below, give the line number and the code that would fix the error.

```
1  import pandas as pd
2  fn = 'data.csv"
3  df = read_csv(fn)
4  for c in df.columns:
5      print(c.upper()
```

   i.
```
   fn = 'data.csv"
        ^
SyntaxError: unterminated string literal
```
Line Number: **Code that fixes the error:**

   ii.
```
   df = read_csv(fn)
        ^^^^^^^^
NameError: name 'read_csv' is not defined
```
Line Number: **Correct code:**

   iii.
```
   print(c.upper()
         ^
SyntaxError: '(' was never closed
```
Line Number: **Correct code:**

3. (a) What will make the following statement true:

```
not in1 and in1
```

☐ Only setting `in1 = False`.
☐ Only setting `in1 = True`.
☐ All `in1` values make the statement true.
☐ No `in1` value makes the statement true.

(b) What is the value of `out`?

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

Choose one:
☐ `False` ☐ Not defined
☐ `True`

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

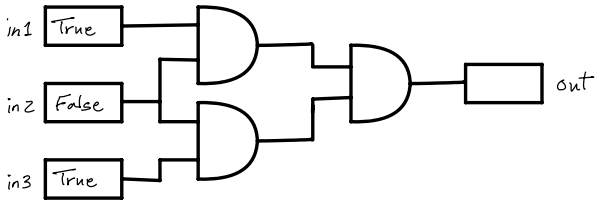
`in1 =` ____

`in2 =` ____

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

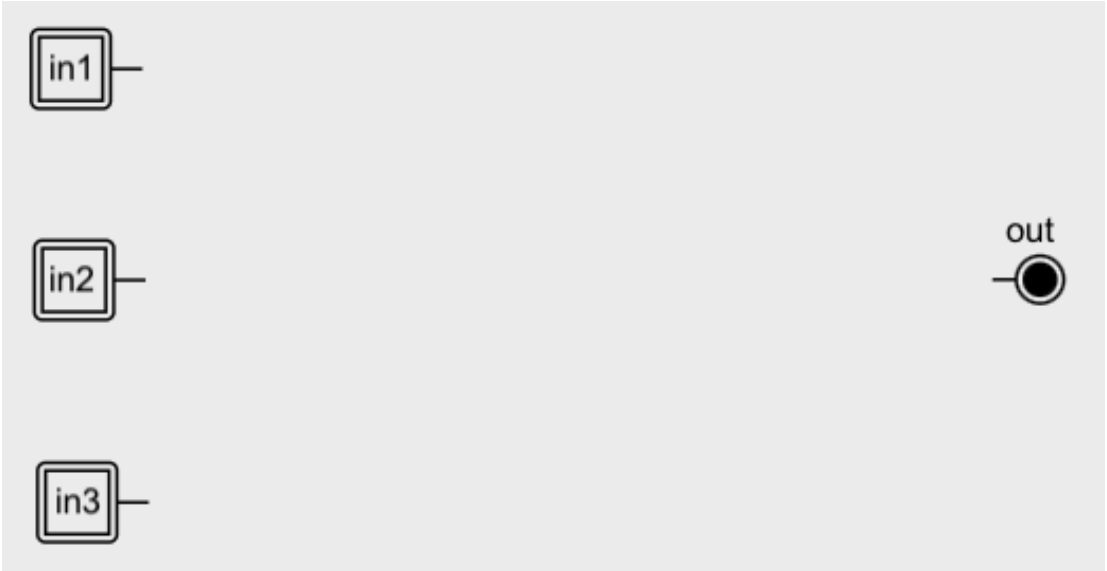
out = `True`

(d) What is the output of this circuit?



Choose one:
☐ `False` ☐ Not defined
☐ `True`

(e) Design a circuit that **exactly implements** the logical expression:
`((not(in1 and in2)) or (not(in2 and not in3))) or in3`

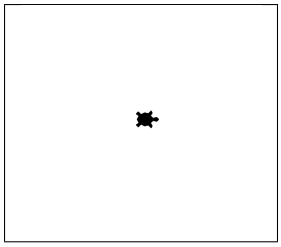


4. (a) Using the turtle and function below, fill in the parameters that yield the output:

```
1  import turtle
2  tess = turtle.Turtle()
3  tess.pensize(5)
4  tess.shape("turtle")
5
6  def drawT(t, commands):
7      if len(commands) != 0:
8          ch = commands[0]
9          if ch == 'F':
10             t.forward(50)
11         elif ch == 'L':
12             t.left(90)
13         elif ch == 'R':
14             t.right(90)
15         elif ch == '^':
16             t.penup()
17         elif ch == 'v':
18             t.pendown()
19         elif ch == 'S':
20             t.stamp()
21         else:
22             print("Error:", ch)
23         drawT(t,commands[1:])
```

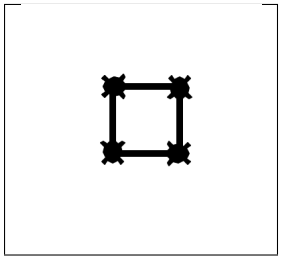
i. `drawT(` ☐)

Output:



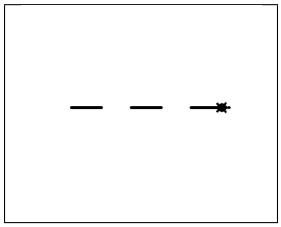
ii. `drawT(` ☐)

Output:



iii. `drawT(` ☐)

Output:



(b) What are the formal parameters for the function `drawT()`: ☐

(c) For which values of `sss`, will `drawT(tess,sss)` print `Error` (check all that apply):

- ☐ `sss = "L^FRvF^R"`
- ☐ `sss = "s"`
- ☐ `sss = "FFFF"`
- ☐ `sss = "BFL"`
- ☐ None of the above.

(d) For which values of `sss`, will `drawT(tess,sss)` **NOT** execute Line 23 (check all that apply):

- ☐ `sss = ""`
- ☐ `sss = "sS"`
- ☐ `sss = "FFFF"`
- ☐ `sss = "^SFSFSv"`
- ☐ None of the above.

5. Design an algorithm that asks the user for a list of locations (given as latitude, longitude, and name) they have visited. Your program should keep track of how many times each location occurs and create only one marker for each location, with the hover text containing the number of visits.

   For example, if the inputted list is:

   ```
   [(40.7678,-73.9645,'Hunter College'),(40.6892,-74.0445, 'Statue of Liberty'),\
    (40.7678,-73.9645,'Hunter College')]
   ```

   your map would display 2 markers: one for Hunter College with a hover text of 2, and one for Statue of Liberty with a hover text of 1.

Libraries: (if any)	
Input:	
Output:	

Design Pattern:

☐ Accumulator ☐ Max/Min ☐ Finding Duplicates ☐ Searching

Principal Mechanisms (select all that apply):

☐ Loop ☐ Conditional (if/else) ☐ Recursion ☐ Indexing/slicing
☐ `input()` ☐ 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 following functions that are part of a program that draws with turtles:

   - `getData()`: gets the color and shape of a turtle and the number of sides of a polygon
   - `getTurtle()`: returns a turtle with color and shape
   - `drawPolygon()`: draws a polygon with n sides using turtle t

```
import turtle
def getData():
    """
    Asks the user for the color and shape of a turtle
    and the number of sides of a polygon.
    Returns the color and shape as strings and the sides as integer.
    """
```

```
def getTurtle(color, shape):
    """
    Returns a turtle with color and shape
    """
```

```
def drawPolygon(t, n):
    """
    Draws a polygon with n sides using turtle t

    """
```

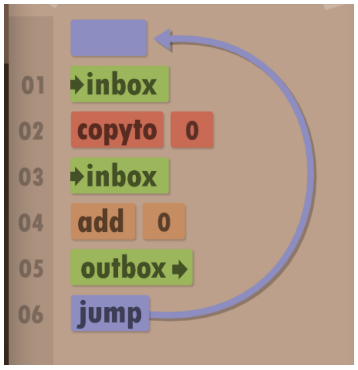
7. Write a **complete program**, based on the Queens Library dataset. Your program should:

   - Ask for the name of the input CSV file and the name of the output HTML file.
   - Use Plotly Express to create a map with the libraries open on Sunday. When the user hovers over the marker, the Sunday hours (e.g. `'Su'` column) should appear in the pop-up.
   - Save the map to output file specified by the user.

First rows of Queens Library dataset:

name	Address	City	Mn	Tu	We	Th	Fr	Sa	Su
Rosedale	144-20 243 Stree	Rosedale	12:00 PM - 8:00	1:00 PM - 6:00 P	10:00 AM - 6:00	12:00 PM - 8:00	10:00 AM - 6:00	10:00 AM - 5:00	Closed - Closed
Bayside	214-20 Northern	Bayside	9:00 AM - 8:00 P	1:00 PM - 6:00 P	10:00 AM - 6:00	12:00 PM - 8:00	10:00 AM - 6:00	10:00 AM - 5:00	Closed - Closed
Central Library-	89-11 Merrick Bo	Jamaica	9:00 AM - 9:00 P	9:00 AM - 7:00 P	9:00 AM - 7:00 P	9:00 AM - 7:00 P	9:00 AM - 7:00 P	10:00 AM - 5:00	12:00 PM - 5:00
Hollis	202-05 Hillside A	Hollis	12:00 PM - 8:00	1:00 PM - 6:00 P	10:00 AM - 6:00	12:00 PM - 8:00	10:00 AM - 6:00	10:00 AM - 5:00	Closed - Closed
Queens Library f	2002 Cornaga Av	Far Rockaway	2:30 PM - 6:00 P	2:30 PM - 6:00 P	2:30 PM - 6:00 P	2:30 PM - 6:00 P	2:30 PM - 6:00 P	Closed - Closed	Closed - Closed
Douglaston/Little	249-01 Northern	Little Neck	12:00 PM - 8:00	1:00 PM - 6:00 P	10:00 AM - 6:00	12:00 PM - 8:00	10:00 AM - 6:00	10:00 AM - 5:00	Closed - Closed
Queensboro Hill	60-05 Main Stree	Flushing	12:00 PM - 8:00	1:00 PM - 6:00 P	10:00 AM - 6:00	12:00 PM - 8:00	10:00 AM - 6:00	10:00 AM - 5:00	Closed - Closed
Howard Beach	92-06 156 Avenu	Howard Beach	12:00 PM - 8:00	1:00 PM - 6:00 P	10:00 AM - 6:00	12:00 PM - 8:00	10:00 AM - 6:00	10:00 AM - 5:00	Closed - Closed

8. (a) What does the Human Resource Machine (HRM) code output with the following input:



Output:

Note: if the input is a letter rather than a number, it counts as zero.

(b) Consider the following MIPS program:

```
ADDI $s0, $zero, 2
ADDI $s1, $zero, 4
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

(c) Consider the MIPS code:

```
1  ADDI $sp, $sp, -6
2  ADDI $t0, $zero, 72
3  ADDI $s2, $zero, 67
4  SETUP: SB $t0, 0($sp)
5  ADDI $sp, $sp, 1
6  ADDI $t0, $t0, -1
7  BEQ $t0, $s2, DONE
8  J SETUP
9  DONE: ADDI $t0, $zero, 0
10 SB $t0, 0($sp)
11 ADDI $sp, $sp, -5
12 ADDI $v0, $zero, 4
13 ADDI $a0, $sp, 0
14 syscall
```

i) How many characters are printed?	
ii) What is the first character printed?	
iii) What is the message printed?	
iv) List what you need to change to print the string "HIJKL":	

9. (a) Fill in the missing code to yield the output:

```
#include <iostream>
using namespace std;
int main()
{
    cout << "We shape out tools\nand ";
    
    cout << "\t--John Culkin, 1967" << endl;
    return 0;
}
```

Output:

We shape out tools
and thereafter they shape us.
–John Culkin, 1967

(b) What is the output:

```
#include <iostream>
using namespace std;
int main()
{
    for (int i = 5; i >= 0; i--){
        if (i < 3)
            cout << "step ";
        cout << i << endl;
    }
}
```

Output:

(c) What is the output:

```
#include <iostream>
using namespace std;
int main()
{
    int size = 5;
    for (int i = 1; i <= size; i++)
    {
        for (int j = 0; j < i; j++)
            cout << "*";
        if (i % 2 == 0)
            cout << "0";
        else
            cout << "1";
        cout << endl;
    }
    return 0;
}
```

Output:

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

Python program:

```
num = -1
while (num < 0) or (num % 10 != 0):
    num = int(input("Enter again: "))
print("Your number:", num)
```

C++ program:

(b) Write a **complete** 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
```

SCRATCH PAPER

SCRATCH PAPER

CSCI 127 Reference Sheet, Spring 2026

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` back `x` steps.
`t.left(x)/t.right(x)`	Turns `t` left/right `x` deg.
`t.penup()/t.pendown()`	Lifts `t`'s pen up/down.
`t.stamp()`	Stamps at `t`'s location.
`t.goto(x,y)`	Moves `t` to `(x,y)`.
`t.colormode(255)`	Enable 255 mode.
`t.color(s)/t.color(x,y,z)`	Changes `t`'s color.

String Methods: Let `s` be a string.

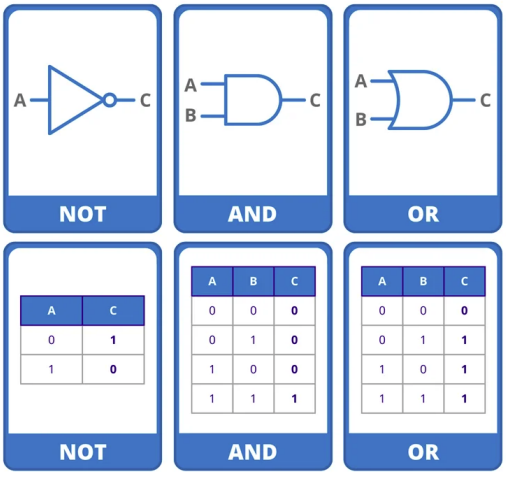
Function	Description
`len(s)`	Returns the length of `s`.
`s.lower()`	Returns `s` as lower case character(s).
`s.upper()`	Returns `s` as upper case character(s).
`s.count(t)`	Returns count of t in s.
`s.find(t)`	Returns index of t in s (-1 not found).
`s.split(d)`	Splits `s` into list of strings on `d`.
`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:v1,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`.



(from truthtablegen.com)

Control Structures:

Function	Description
`for i in range(0,100,5):`	Basic for-loop.
`for c in str_var:`	Looping through a string.
`if x==0:... elif x>0:... else:...`	Basic if-statement.

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((10,20,3))`	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 of dictionary `d`.
`df.to_csv(file_name)`	Writes `df` to `file_name`.
`df[col]`	Returns `col` column as a Series.
`df[[c1,c2]]`	Returns DataFrame with `c1` & `c2`.
`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`.
`df.dropna(subset=[c1,c2])`	Drops rows with missing c1 or c2 values.

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 -l / 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`.

(From wikipedia commons)

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]