

CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

Announcements



- Starting this week, undergraduate teaching assistants (UTAs) will be in 1001E North, assisting with recitation sessions and holding tutoring hours.

Announcements



- Starting this week, undergraduate teaching assistants (UTAs) will be in 1001E North, assisting with recitation sessions and holding tutoring hours.
- Today is the last day to submit the first program ('Hello, World!') to Gradescope.

Announcements



- Starting this week, undergraduate teaching assistants (UTAs) will be in 1001E North, assisting with recitation sessions and holding tutoring hours.
- Today is the last day to submit the first program ('Hello, World!') to Gradescope.
- Last day to complete in-class Quiz 1 is tomorrow (9/7). You must attend a recitation section to take your quiz.

Frequently Asked Questions

From lecture slips & recitation sections.

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?

There is no midterm. Instead there's 14 in-class quizzes.

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.
- Can I submit late homework?

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.
- Can I submit late homework?
No. Instead we drop the 5 lowest grades.

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.
- Can I submit late homework?
No. Instead we drop the 5 lowest grades.
- I missed class. Do you need documentation?

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.
- Can I submit late homework?
No. Instead we drop the 5 lowest grades.
- I missed class. Do you need documentation?
*No. We will automatically drop the 2 lowest quiz grades.
If you will miss ≥ 3 weeks ($> 20\%$ of class), see us about taking a future term.*

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.
- Can I submit late homework?
No. Instead we drop the 5 lowest grades.
- I missed class. Do you need documentation?
*No. We will automatically drop the 2 lowest quiz grades.
If you will miss ≥ 3 weeks ($> 20\%$ of class), see us about taking a future term.*
- Why do I have to work in groups?

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.
- Can I submit late homework?
No. Instead we drop the 5 lowest grades.
- I missed class. Do you need documentation?
*No. We will automatically drop the 2 lowest quiz grades.
If you will miss ≥ 3 weeks ($> 20\%$ of class), see us about taking a future term.*
- Why do I have to work in groups?
It's great practice to explain technical work to others.

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.
- Can I submit late homework?
No. Instead we drop the 5 lowest grades.
- I missed class. Do you need documentation?
*No. We will automatically drop the 2 lowest quiz grades.
If you will miss ≥ 3 weeks ($> 20\%$ of class), see us about taking a future term.*
- Why do I have to work in groups?
It's great practice to explain technical work to others.
- Can I work ahead?

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.
- Can I submit late homework?
No. Instead we drop the 5 lowest grades.
- I missed class. Do you need documentation?
*No. We will automatically drop the 2 lowest quiz grades.
If you will miss ≥ 3 weeks ($> 20\%$ of class), see us about taking a future term.*
- Why do I have to work in groups?
It's great practice to explain technical work to others.
- Can I work ahead?
Yes! All programs are available, on gradescope, 3 weeks before the deadline.

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.
- Can I submit late homework?
No. Instead we drop the 5 lowest grades.
- I missed class. Do you need documentation?
*No. We will automatically drop the 2 lowest quiz grades.
If you will miss ≥ 3 weeks ($> 20\%$ of class), see us about taking a future term.*
- Why do I have to work in groups?
It's great practice to explain technical work to others.
- Can I work ahead?
Yes! All programs are available, on gradescope, 3 weeks before the deadline.
- You said “when you take second semester...” I just took this class for Pathways...

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.
- Can I submit late homework?
No. Instead we drop the 5 lowest grades.
- I missed class. Do you need documentation?
*No. We will automatically drop the 2 lowest quiz grades.
If you will miss ≥ 3 weeks ($> 20\%$ of class), see us about taking a future term.*
- Why do I have to work in groups?
It's great practice to explain technical work to others.
- Can I work ahead?
Yes! All programs are available, on gradescope, 3 weeks before the deadline.
- You said “when you take second semester...” I just took this class for Pathways...
This is Pathways, but we hope that you will be a CS major/minor.

Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Wednesday, 20 December, 9-11am.
- Can I submit late homework?
No. Instead we drop the 5 lowest grades.
- I missed class. Do you need documentation?
*No. We will automatically drop the 2 lowest quiz grades.
If you will miss ≥ 3 weeks ($> 20\%$ of class), see us about taking a future term.*
- Why do I have to work in groups?
It's great practice to explain technical work to others.
- Can I work ahead?
Yes! All programs are available, on gradescope, 3 weeks before the deadline.
- You said “when you take second semester...” I just took this class for Pathways...
This is Pathways, but we hope that you will be a CS major/minor.
- *We also hope: “Get your education don't forget whence you came...”*

Today's Topics



- For-loops
- `range()`
- Variables: ints and strings
- Lists

In Pairs or Triples...

Some review and some novel challenges:

```
1 #Predict what will be printed:
2
3 for i in range(4):
4     print('The world turned upside down')
5
6 for j in [0,1,2,3,4,5]:
7     print(j)
8
9 for count in range(6):
10    print(count)
11
12 for color in ['red', 'green', 'blue']:
13    print(color)
14
15 print()
16 print()
17
18 for i in range(2):
19     for j in range(2):
20         print('Look around,')
21     print('How lucky we are to be alive!')
```

Python Tutor

```
1 #Predict what will be printed:
2
3 for i in range(4):
4     print('The world turned upside down')
5
6 for j in [0,1,2,3,4,5]:
7     print(j)
8
9 for count in range(6):
10    print(count)
11
12 for color in ['red', 'green', 'blue']:
13    print(color)
14
15 print()
16 print()
17
18 for i in range(2):
19     for j in range(2):
20         print('Look around,')
21     print('How lucky we are to be alive!')
```

(Demo with pythonTutor)

Variables

- A **variable** is a reserved memory location for storing a value.



Variables



- A **variable** is a reserved memory location for storing a value.
- Different kinds, or **types**, of values need different amounts of space:
 - ▶ **int**: integer or whole numbers

Variables



- A **variable** is a reserved memory location for storing a value.
- Different kinds, or **types**, of values need different amounts of space:
 - ▶ **int**: integer or whole numbers
 - ▶ **float**: floating point or real numbers

Variables



- A **variable** is a reserved memory location for storing a value.
- Different kinds, or **types**, of values need different amounts of space:
 - ▶ **int**: integer or whole numbers
 - ▶ **float**: floating point or real numbers
 - ▶ **string**: sequence of characters

Variables



- A **variable** is a reserved memory location for storing a value.
- Different kinds, or **types**, of values need different amounts of space:
 - ▶ **int**: integer or whole numbers
 - ▶ **float**: floating point or real numbers
 - ▶ **string**: sequence of characters
 - ▶ **list**: a sequence of items

Variables



- A **variable** is a reserved memory location for storing a value.
- Different kinds, or **types**, of values need different amounts of space:
 - ▶ **int**: integer or whole numbers
 - ▶ **float**: floating point or real numbers
 - ▶ **string**: sequence of characters
 - ▶ **list**: a sequence of items
 - ▶ **class variables**: for complex objects, like turtles.

Variable Names

- There's some rules about valid names for variables.



Variable Names



- There's some rules about valid names for variables.
- Can use the underscore ('_'), upper and lower case letters.

Variable Names



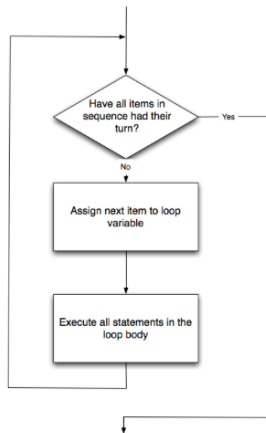
- There's some rules about valid names for variables.
- Can use the underscore ('_'), upper and lower case letters.
- Can also use numbers, just can't start a name with a number.

Variable Names



- There's some rules about valid names for variables.
- Can use the underscore ('_'), upper and lower case letters.
- Can also use numbers, just can't start a name with a number.
- Can't use symbols (like '+' or '*') since used for arithmetic.

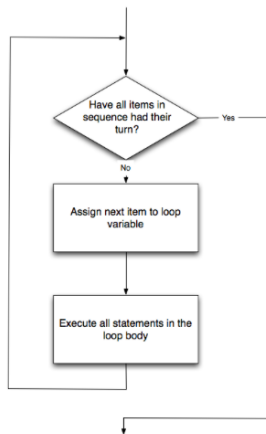
for-loop



```
for i in list:  
    statement1  
    statement2  
    statement3
```

How to Think Like CS, §4.5

for-loop



How to Think Like CS, §4.5

```
for i in list:  
    statement1  
    statement2  
    statement3
```

where `list` is a list of items:

- stated explicitly (e.g. `[1,2,3]`) or
- generated by a function, e.g. `range()`.

In Pairs or Triples...

Some review and some novel challenges:

```
1 #Predict what will be printed:
2
3 for num in [2,4,6,8,10]:
4     print(num)
5
6 sum = 0
7 for x in range(0,12,2):
8     print(x)
9     sum = sum + x
10
11 print(x)
12
13 for c in "ABCD":
14     print(c)
```

Python Tutor

```
1 #Predict what will be printed:
2
3 for num in [2,4,6,8,10]:
4     print(num)
5
6 sum = 0
7 for x in range(0,12,2):
8     print(x)
9     sum = sum + x
10
11 print(x)
12
13 for c in "ABCD":
14     print(c)
```

(Demo with pythonTutor)

range()

Simplest version:



range()



Simplest version:

- `range(stop)`

range()



Simplest version:

- `range(stop)`
- Produces a list: `[0,1,2,3,...,stop-1]`

range()



Simplest version:

- `range(stop)`
- Produces a list: `[0,1,2,3,...,stop-1]`
- For example, if you want the the list `[0,1,2,3,...,100]`, you would write:

range()



Simplest version:

- `range(stop)`
- Produces a list: `[0,1,2,3,...,stop-1]`
- For example, if you want the the list `[0,1,2,3,...,100]`, you would write:

```
range(101)
```

range()

What if you wanted to start somewhere else:



`range()`

What if you wanted to start somewhere else:

- `range(start, stop)`



range()



What if you wanted to start somewhere else:

- `range(start, stop)`
- Produces a list:
`[start, start+1, ..., stop-1]`

range()



What if you wanted to start somewhere else:

- `range(start, stop)`
- Produces a list:
`[start, start+1, ..., stop-1]`
- For example, if you want the the list
`[10, 11, ..., 20]`
you would write:

range()



What if you wanted to start somewhere else:

- `range(start, stop)`
- Produces a list:
`[start, start+1, ..., stop-1]`
- For example, if you want the the list
`[10, 11, ..., 20]`
you would write:

```
range(10, 21)
```

`range()`

What if you wanted to count by twos, or
some other number:



range()

What if you wanted to count by twos, or some other number:

- `range(start, stop, step)`



range()

What if you wanted to count by twos, or some other number:

- `range(start, stop, step)`
- Produces a list:
[start, start+step, start+2*step..., last]
(where last is the largest start+k*step less than stop)



range()



What if you wanted to count by twos, or some other number:

- `range(start, stop, step)`
- Produces a list:
`[start, start+step, start+2*step..., last]`
(where last is the largest `start+k*step` less than stop)
- For example, if you want the the list `[5,10,...,50]` you would write:

range()



What if you wanted to count by twos, or some other number:

- `range(start, stop, step)`
- Produces a list:
`[start, start+step, start+2*step..., last]`
(where last is the largest `start+k*step` less than stop)
- For example, if you want the the list `[5,10,...,50]` you would write:

```
range(5, 51, 5)
```

In summary: `range()`



The three versions:

In summary: `range()`



The three versions:

- `range(stop)`

In summary: `range()`



The three versions:

- `range(stop)`
- `range(start, stop)`

In summary: `range()`



The three versions:

- `range(stop)`
- `range(start, stop)`
- `range(start, stop, step)`

Standardized Code for Characters

American Standard Code for Information Interchange (ASCII), 1960.

Standardized Code for Characters

American Standard Code for Information Interchange (ASCII), 1960.
(New version called: Unicode).

Standardized Code for Characters

American Standard Code for Information Interchange (ASCII), 1960.
(New version called: Unicode).

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]

(wiki)

Converting from Character to Code:

- ord(c): returns Unicode (ASCII) of the character.

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	00		104	68	p	192	C0	€	255	FF	ÿ
1	01		105	69	q	193	C1		256	00	
2	02		106	6A	r	194	C2		257	01	
3	03		107	6B	s	195	C3		258	02	
4	04		108	6C	t	196	C4		259	03	
5	05		109	6D	u	197	C5		260	04	
6	06		110	6E	v	198	C6		261	05	
7	07		111	6F	w	199	C7		262	06	
8	08		112	70	x	200	C8		263	07	
9	09		113	71	y	201	C9		264	08	
10	0A	�	114	72	z	202	CA		265	09	
11	0B		115	73	{	203	CB		266	0A	�
12	0C	�	116	74		204	CC		267	0B	
13	0D		117	75	}	205	CD		268	0C	�
14	0E		118	76	~	206	CE		269	0D	
15	0F		119	77		207	CF		270	0E	�
16	10		120	78		208	D0		271	0F	
17	11		121	79		209	D1		272	10	
18	12		122	7A		210	D2		273	11	
19	13		123	7B		211	D3		274	12	
20	14		124	7C		212	D4		275	13	
21	15		125	7D		213	D5		276	14	
22	16		126	7E		214	D6		277	15	
23	17		127	7F		215	D7		278	16	
24	18					216	D8		279	17	
25	19					217	D9		280	18	
26	1A					218	DA	¡	281	19	
27	1B					219	DB	¢	282	1A	
28	1C					220	DC	£	283	1B	
29	1D					221	DD	¤	284	1C	
30	1E					222	DE	¥	285	1D	
31	1F					223	DF	¦	286	1E	
32	20					224	DD	§	287	1F	
33	21					225	DD	¨	288	20	
34	22					226	DD	©	289	21	
35	23					227	DD	ª	290	22	
36	24					228	DD	«	291	23	
37	25					229	DD	¬	292	24	
38	26					230	DD	­	293	25	
39	27					231	DD	®	294	26	
40	28					232	DD	¯	295	27	
41	29					233	DD	°	296	28	
42	2A	�				234	DD	±	297	29	
43	2B					235	DD	²	298	2A	�
44	2C	�				236	DD	³	299	2B	
45	2D					237	DD	´	300	2C	�
46	2E	�				238	DD	µ	301	2D	
47	2F					239	DD	¶	302	2E	�
48	30					240	DD	·	303	2F	
49	31					241	DD	¸	304	30	
50	32					242	DD	¹	305	31	
51	33					243	DD	º	306	32	
52	34					244	DD	»	307	33	
53	35					245	DD	¼	308	34	
54	36					246	DD	½	309	35	
55	37					247	DD	¾	310	36	
56	38	�				248	DD	¿	311	37	
57	39					249	DD		312	38	�
58	3A					250	DD		313	39	
59	3B					251	DD		314	3A	
60	3C					252	DD		315	3B	
61	3D					253	DD		316	3C	
62	3E					254	DD		317	3D	
63	3F					255	DD		318	3E	

Converting from Character to Code:

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	00		16	10	P	32	20	[48	30	0
1	01	SOH	17	11	Q	33	21	\	49	31	1
2	02	STX	18	12	R	34	22]	50	32	2
3	03	ETX	19	13	S	35	23	^	51	33	3
4	04	END	20	14	T	36	24	_	52	34	4
5	05	SO	21	15	U	37	25	`	53	35	5
6	06	SI	22	16	V	38	26	{	54	36	6
7	07	BELL	23	17	W	39	27		55	37	7
8	08	BS	24	18	X	40	28	~	56	38	8
9	09	HT	25	19	Y	41	29	DEL	127	7F	DEL
10	0A	LF	26	1A	Z	42	2A				
11	0B	VT	27	1B	[43	2B				
12	0C	FF	28	1C	\	44	2C				
13	0D	CR	29	1D]	45	2D				
14	0E		30	1E	^	46	2E				
15	0F		31	1F	_	47	2F				
16	10	`	32	20	[48	30	0			
17	11	1	33	21	\	49	31	1			
18	12	2	34	22]	50	32	2			
19	13	3	35	23	^	51	33	3			
20	14	4	36	24	_	52	34	4			
21	15	5	37	25	`	53	35	5			
22	16	6	38	26	{	54	36	6			
23	17	7	39	27		55	37	7			
24	18	8	40	28	~	56	38	8			
25	19	9	41	29		57	39	9			
26	1A	:	42	2A		58	3A	:			
27	1B	;	43	2B		59	3B	;			
28	1C	<	44	2C		5A	3A	<			
29	1D	=	45	2D		5B	3B	=			
30	1E	>	46	2E		5C	3C	>			
31	1F	?	47	2F		5D	3D	?			
32	20	`	48	30	0	5E	3E	`			
33	21	1	49	31	1	5F	3F	1			
34	22	2	50	32	2						
35	23	3	51	33	3						
36	24	4	52	34	4						
37	25	5	53	35	5						
38	26	6	54	36	6						
39	27	7	55	37	7						
40	28	8	56	38	8						
41	29	9	57	39	9						
42	2A	:	58	3A	:						
43	2B	;	59	3B	;						
44	2C	<	5A	3C	<						
45	2D	=	5B	3D	=						
46	2E	>	5C	3E	>						
47	2F	?	5D	3F	?						
48	30	0	5E	3E	`						
49	31	1	5F	3F	1						
50	32	2									
51	33	3									
52	34	4									
53	35	5									
54	36	6									
55	37	7									
56	38	8									
57	39	9									
58	3A	:									
59	3B	;									
60	3C	<									
61	3D	=									
62	3E	>									
63	3F	?									
64	40	`									
65	41	A									
66	42	B									
67	43	C									
68	44	D									
69	45	E									
70	46	F									
71	47	`									
72	48	a									
73	49	b									
74	4A	c									
75	4B	d									
76	4C	e									
77	4D	f									
78	4E	g									
79	4F	h									
80	50	i									
81	51	j									
82	52	k									
83	53	l									
84	54	m									
85	55	n									
86	56	o									
87	57	p									
88	58	q									
89	59	r									
90	5A	s									
91	5B	t									
92	5C	u									
93	5D	v									
94	5E	w									
95	5F	x									
96	60	y									
97	61	z									
98	62	[
99	63	\									
100	64]									
101	65	^									
102	66	_									
103	67	`									
104	68	{									
105	69										
106	6A	~									
107	6B										
108	6C										
109	6D										
110	6E										
111	6F										
112	70										
113	71										
114	72										
115	73										
116	74										
117	75										
118	76										
119	77										
120	78										
121	79										
122	7A										
123	7B										
124	7C										
125	7D										
126	7E										
127	7F	DEL									

- `ord(c)`: returns Unicode (ASCII) of the character.
- Example: `ord('a')` returns 97.

Converting from Character to Code:

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	00		10	0A	LF	20	14	SP	30	1E	0
1	01	SOH	11	0B	VT	21	15	!	31	1F	1
2	02	STX	12	0C	FF	22	16	"	32	20	2
3	03	ETX	13	0D	CR	23	17	#	33	21	3
4	04	END	14	0E	SO	24	18	\$	34	22	4
5	05	ESC	15	0F	SH	25	19	%	35	23	5
6	06	ACK	16	10	HT	26	1A	&	36	24	6
7	07	BEL	17	11	BT	27	1B	'	37	25	7
8	08	BS	18	12	BO	28	1C	(38	26	8
9	09	HT	19	13	BU	29	1D)	39	27	9
10	0A	LF	1A	14	BD	2A	1E	*	3A	28	*
11	0B	VT	1B	15	BC	2B	1F	+	3B	29	+
12	0C	FF	1C	16	BB	2C	20	,	3C	2A	,
13	0D	CR	1D	17	BA	2D	21	-	3D	2B	-
14	0E	SO	1E	18	B9	2E	22	.	3E	2C	.
15	0F	SH	1F	19	B8	2F	23	:	3F	2D	:
16	10	HT	20	1A	B7	30	24	;	40	2E	;
17	11	BT	21	1B	B6	31	25	<	41	2F	<
18	12	BO	22	1C	B5	32	26	=	42	30	=
19	13	BU	23	1D	B4	33	27	>	43	31	>
20	14	BD	24	1E	B3	34	28	?	44	32	?
21	15	BC	25	1F	B2	35	29	@	45	33	@
22	16	BA	26	20	B1	36	2A	A	46	34	A
23	17	B9	27	21	B0	37	2B	B	47	35	B
24	18	B8	28	22	A7	38	2C	C	48	36	C
25	19	B7	29	23	A6	39	2D	D	49	37	D
26	1A	B6	2A	24	A5	3A	2E	E	4A	38	E
27	1B	B5	2B	25	A4	3B	2F	F	4B	39	F
28	1C	B4	2C	26	97	3C	30	[4C	3A	[
29	1D	B3	2D	27	96	3D	31	\	4D	3B	\
30	1E	B2	2E	28	95	3E	32]	4E	3C]
31	1F	B1	2F	29	94	3F	33	^	4F	3D	^
32	20	B0	30	2A	93	40	34	_	50	3E	_
33	21	A7	31	2B	92	41	35	`	51	3F	`
34	22	A6	32	2C	83	42	36	{	52	40	{
35	23	A5	33	2D	82	43	37		53	41	
36	24	A4	34	2E	81	44	38	~	54	42	~
37	25	A3	35	2F	7C	45	39	DEL	55	43	DEL
38	26	A2	36	30	7B	46	3A		56	44	
39	27	A1	37	31	7A	47	3B		57	45	
40	28	A0	38	32	79	48	3C		58	46	
41	29	97	39	33	78	49	3D		59	47	
42	2A	96	3A	34	77	4A	3E		5A	48	
43	2B	95	3B	35	76	4B	3F		5B	49	
44	2C	94	3C	36	75	4C	40		5C	4A	
45	2D	93	3D	37	74	4D	41		5D	4B	
46	2E	92	3E	38	73	4E	42		5E	4C	
47	2F	91	3F	39	72	4F	43		5F	4D	
48	30	83	40	3A	71	50	44		60	4E	
49	31	82	41	3B	70	51	45		61	4F	
50	32	81	42	3C	6F	52	46		62	50	
51	33	80	43	3D	6E	53	47		63	51	
52	34	7C	44	3E	6D	54	48		64	52	
53	35	7B	45	3F	6C	55	49		65	53	
54	36	7A	46	40	6B	56	4A		66	54	
55	37	79	47	41	6A	57	4B		67	55	
56	38	78	48	42	69	58	4C		68	56	
57	39	77	49	43	68	59	4D		69	57	
58	3A	76	4A	44	67	5A	4E		6A	58	
59	3B	75	4B	45	66	5B	4F		6B	59	
60	3C	74	4C	46	65	5C	50		6C	5A	
61	3D	73	4D	47	64	5D	51		6D	5B	
62	3E	72	4E	48	63	5E	52		6E	5C	
63	3F	71	4F	49	62	5F	53		6F	5D	
64	40	6F	50	4A	61	60	54		70	5E	
65	41	6E	51	4B	60	61	55		71	5F	
66	42	6D	52	4C	5F	62	56		72	60	
67	43	6C	53	4D	5E	63	57		73	61	
68	44	6B	54	4E	5D	64	58		74	62	
69	45	6A	55	4F	5C	65	59		75	63	
70	46	69	56	50	5B	66	5A		76	64	
71	47	68	57	51	5A	67	5B		77	65	
72	48	67	58	52	59	68	5C		78	66	
73	49	66	59	53	58	69	5D		79	67	
74	4A	65	5A	54	57	6A	5E		7A	68	
75	4B	64	5B	55	56	6B	5F		7B	69	
76	4C	63	5C	56	55	6C	60		7C	6A	
77	4D	62	5D	57	54	6D	61		7D	6B	
78	4E	61	5E	58	53	6E	62		7E	6C	
79	4F	60	5F	59	52	6F	63		7F	6D	
80	50	5F	60	5A	51	70	64				
81	51	5E	5F	5B	50	71	65				
82	52	5D	5E	5C	4F	72	66				
83	53	5C	5D	5D	4E	73	67				
84	54	5B	5C	5E	4D	74	68				
85	55	5A	5B	5F	4C	75	69				
86	56	59	5A	60	4B	76	6A				
87	57	58	59	5F	4A	77	6B				
88	58	57	58	5E	49	78	6C				
89	59	56	57	5D	48	79	6D				
90	5A	55	56	5C	47	7A	6E				
91	5B	54	55	5B	46	7B	6F				
92	5C	53	54	5A	45	7C	70				
93	5D	52	53	59	44	7D	71				
94	5E	51	52	58	43	7E	72				
95	5F	50	51	57	42		73				
96	60	4F	50	56	41		74				
97	61	4E	4F	55	40		75				
98	62	4D	4E	54	3F		76				
99	63	4C	4D	53	3E		77				
100	64	4B	4C	52	3D		78				

- `ord(c)`: returns Unicode (ASCII) of the character.
- Example: `ord('a')` returns 97.
- `chr(x)`: returns the character whose Unicode is x.

Converting from Character to Code:

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	00		10	0A	LF	20	14	SP	30	1E	0
1	01	SOH	11	0B	VT	21	15	!	31	1F	1
2	02	STX	12	0C	FF	22	16	"	32	20	2
3	03	ETX	13	0D	CR	23	17	#	33	21	3
4	04	END	14	0E	SO	24	18	\$	34	22	4
5	05	ESC	15	0F	SH	25	19	%	35	23	5
6	06	ACK	16	10	SI	26	1A	&	36	24	6
7	07	BEL	17	11	DL	27	1B	'	37	25	7
8	08	BS	18	12	IL	28	1C	(38	26	8
9	09	HT	19	13	PL	29	1D)	39	27	9
10	0A	LF	20	14	ST	30	1E	*	40	28	10
11	0B	VT	21	15	ET	31	1F	+	41	29	11
12	0C	FF	22	16	EN	32	20	,	42	2A	12
13	0D	CR	23	17	SO	33	21	-	43	2B	13
14	0E	SO	24	18	SB	34	22	.	44	2C	14
15	0F	SH	25	19	RF	35	23	:	45	2D	15
16	10	SI	26	1A	SS	36	24	;	46	2E	16
17	11	DL	27	1B	ST	37	25	<	47	2F	17
18	12	IL	28	1C	ET	38	26	=	48	30	18
19	13	PL	29	1D	EN	39	27	>	49	31	19
20	14	ST	30	1E	SO	40	28	?	50	32	20
21	15	ET	31	1F	SB	41	29	@	51	33	21
22	16	EN	32	20	RF	42	2A	A	52	34	22
23	17	SO	33	21	SS	43	2B	B	53	35	23
24	18	SB	34	22	ST	44	2C	C	54	36	24
25	19	RF	35	23	ET	45	2D	D	55	37	25
26	1A	SS	36	24	EN	46	2E	E	56	38	26
27	1B	ST	37	25	SO	47	2F	F	57	39	27
28	1C	ET	38	26	SB	48	30		58	3A	28
29	1D	EN	39	27	RF	49	31		59	3B	29
30	1E	SO	40	28	SS	50	32		60	3C	30
31	1F	SB	41	29	ST	51	33		61	3D	31
32	20	RF	42	2A	ET	52	34		62	3E	32
33	21	SS	43	2B	EN	53	35		63	3F	33
34	22	ST	44	2C	SO	54	36		64	40	34
35	23	ET	45	2D	SB	55	37		65	41	35
36	24	EN	46	2E	RF	56	38		66	42	36
37	25	SO	47	2F	SS	57	39		67	43	37
38	26	SB	48	30	ST	58	3A		68	44	38
39	27	RF	49	31	ET	59	3B		69	45	39
40	28	SS	50	32	EN	60	3C		70	46	40
41	29	ST	51	33	SO	61	3D		71	47	41
42	2A	ET	52	34	SB	62	3E		72	48	42
43	2B	EN	53	35	RF	63	3F		73	49	43
44	2C	SO	54	36	SS	64	40		74	4A	44
45	2D	SB	55	37	ST	65	41		75	4B	45
46	2E	RF	56	38	ET	66	42		76	4C	46
47	2F	SS	57	39	EN	67	43		77	4D	47
48	30	ST	58	3A	SO	68	44		78	4E	48
49	31	ET	59	3B	SB	69	45		79	4F	49
50	32	EN	60	3C	RF	70	46		80	50	50
51	33	SO	61	3D	SS	71	47		81	51	51
52	34	SB	62	3E	ST	72	48		82	52	52
53	35	RF	63	3F	ET	73	49		83	53	53
54	36	SS	64	40	EN	74	4A		84	54	54
55	37	ST	65	41	SO	75	4B		85	55	55
56	38	ET	66	42	SB	76	4C		86	56	56
57	39	EN	67	43	RF	77	4D		87	57	57
58	3A	SO	68	44	SS	78	4E		88	58	58
59	3B	SB	69	45	ST	79	4F		89	59	59
60	3C	RF	70	46	ET	80	50		90	5A	60
61	3D	SS	71	47	EN	81	51		91	5B	61
62	3E	ST	72	48	SO	82	52		92	5C	62
63	3F	ET	73	49	SB	83	53		93	5D	63
64	40	EN	74	4A	RF	84	54		94	5E	64
65	41	SO	75	4B	SS	85	55		95	5F	65
66	42	SB	76	4C	ST	86	56		96	60	66
67	43	RF	77	4D	ET	87	57		97	61	67
68	44	SS	78	4E	EN	88	58		98	62	68
69	45	ST	79	4F	SO	89	59		99	63	69
70	46	ET	80	50	SB	90	5A		100	64	70
71	47	EN	81	51	RF	91	5B		101	65	71
72	48	SO	82	52	SS	92	5C		102	66	72
73	49	SB	83	53	ST	93	5D		103	67	73
74	4A	RF	84	54	ET	94	5E		104	68	74
75	4B	SS	85	55	EN	95	5F		105	69	75
76	4C	ST	86	56	SO	96	60		106	6A	76
77	4D	ET	87	57	SB	97	61		107	6B	77
78	4E	EN	88	58	RF	98	62		108	6C	78
79	4F	SO	89	59	SS	99	63		109	6D	79
80	50	SB	90	5A	ST	100	64		110	6E	80
81	51	RF	91	5B	ET	110	6E		111	6F	81
82	52	SS	92	5C	EN	111	6F		112	70	82
83	53	ST	93	5D	SO	112	70		113	71	83
84	54	ET	94	5E	SB	113	71		114	72	84
85	55	EN	95	5F	RF	114	72		115	73	85
86	56	SO	96	60	SS	115	73		116	74	86
87	57	SB	97	61	ST	116	74		117	75	87
88	58	RF	98	62	ET	117	75		118	76	88
89	59	SS	99	63	EN	118	76		119	77	89
90	5A	ST	100	64	SO	119	77		120	78	90
91	5B	ET	101	65	SB	120	78		121	79	91
92	5C	EN	102	66	RF	121	79		122	7A	92
93	5D	SO	103	67	SS	122	7A		123	7B	93
94	5E	SB	104	68	ST	123	7B		124	7C	94
95	5F	RF	105	69	ET	124	7C		125	7D	95
96	60	SS	106	6A	EN	125	7D		126	7E	96
97	61	ST	107	6B	SO	126	7E		127	7F	97
98	62	ET	108	6C	SB	127	7F				
99	63	EN	109	6D	RF						
100	64	SO	110	6E	SS						

- `ord(c)`: returns Unicode (ASCII) of the character.
- Example: `ord('a')` returns 97.
- `chr(x)`: returns the character whose Unicode is x.
- Example: `chr(97)` returns 'a'.

Converting from Character to Code:

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	00		10	0A	LF	20	14	SP	30	1E	0
1	01	SOH	11	0B	VT	21	15	!	31	1F	1
2	02	STX	12	0C	FF	22	16	"	32	20	2
3	03	ETX	13	0D	CR	23	17	#	33	21	3
4	04	END	14	0E	SO	24	18	\$	34	22	4
5	05	ESC	15	0F	SH	25	19	%	35	23	5
6	06	ACK	16	10	SI	26	1A	&	36	24	6
7	07	BEL	17	11	DL	27	1B	'	37	25	7
8	08	BS	18	12	IL	28	1C	(38	26	8
9	09	HT	19	13	PL	29	1D)	39	27	9
10	0A	LF	20	14	ST	30	1E	*	40	28	10
11	0B	VT	21	15	ET	31	1F	+	41	29	11
12	0C	FF	22	16	EN	32	20	,	42	2A	12
13	0D	CR	23	17	SO	33	21	-	43	2B	13
14	0E	SO	24	18	SB	34	22	.	44	2C	14
15	0F	SH	25	19	RF	35	23	:	45	2D	15
16	10	SI	26	1A	SS	36	24	;	46	2E	16
17	11	DL	27	1B	ST	37	25	<	47	2F	17
18	12	IL	28	1C	ET	38	26	=	48	30	18
19	13	PL	29	1D	EN	39	27	>	49	31	19
20	14	ST	30	1E	SO	40	28	?	50	32	20
21	15	ET	31	1F	SB	41	29	@	51	33	21
22	16	EN	32	20	RF	42	2A	A	52	34	22
23	17	SO	33	21	SS	43	2B	B	53	35	23
24	18	SB	34	22	ST	44	2C	C	54	36	24
25	19	RF	35	23	ET	45	2D	D	55	37	25
26	1A	SS	36	24	EN	46	2E	E	56	38	26
27	1B	ST	37	25	SO	47	2F	F	57	39	27
28	1C	ET	38	26	SB	48	30		58	3A	28
29	1D	EN	39	27	RF	49	31		59	3B	29
30	1E	SO	40	28	SS	50	32		60	3C	30
31	1F	SB	41	29	ST	51	33		61	3D	31
32	20	RF	42	2A	ET	52	34		62	3E	32
33	21	SS	43	2B	EN	53	35		63	3F	33
34	22	ST	44	2C	SO	54	36		64	40	34
35	23	ET	45	2D	SB	55	37		65	41	35
36	24	EN	46	2E	RF	56	38		66	42	36
37	25	SO	47	2F	SS	57	39		67	43	37
38	26	SB	48	30	ST	58	3A		68	44	38
39	27	RF	49	31	ET	59	3B		69	45	39
40	28	SS	50	32	EN	60	3C		70	46	40
41	29	ST	51	33	SO	61	3D		71	47	41
42	2A	ET	52	34	SB	62	3E		72	48	42
43	2B	EN	53	35	RF	63	3F		73	49	43
44	2C	SO	54	36	SS	64	40		74	4A	44
45	2D	SB	55	37	ST	65	41		75	4B	45
46	2E	RF	56	38	ET	66	42		76	4C	46
47	2F	SS	57	39	EN	67	43		77	4D	47
48	30	ST	58	3A	SO	68	44		78	4E	48
49	31	ET	59	3B	SB	69	45		79	4F	49
50	32	EN	60	3C	RF	70	46		80	50	50
51	33	SO	61	3D	SS	71	47		81	51	51
52	34	SB	62	3E	ST	72	48		82	52	52
53	35	RF	63	3F	ET	73	49		83	53	53
54	36	SS	64	40	EN	74	4A		84	54	54
55	37	ST	65	41	SO	75	4B		85	55	55
56	38	ET	66	42	SB	76	4C		86	56	56
57	39	EN	67	43	RF	77	4D		87	57	57
58	3A	SO	68	44	SS	78	4E		88	58	58
59	3B	SB	69	45	ST	79	4F		89	59	59
60	3C	RF	70	46	ET	80	50		90	5A	60
61	3D	SS	71	47	EN	81	51		91	5B	61
62	3E	ST	72	48	SO	82	52		92	5C	62
63	3F	ET	73	49	SB	83	53		93	5D	63
64	40	EN	74	4A	RF	84	54		94	5E	64
65	41	SO	75	4B	SS	85	55		95	5F	65
66	42	SB	76	4C	ST	86	56		96	60	66
67	43	RF	77	4D	ET	87	57		97	61	67
68	44	SS	78	4E	EN	88	58		98	62	68
69	45	ST	79	4F	SO	89	59		99	63	69
70	46	ET	80	50	SB	90	5A		100	64	70
71	47	EN	81	51	RF	91	5B		101	65	71
72	48	SO	82	52	SS	92	5C		102	66	72
73	49	SB	83	53	ST	93	5D		103	67	73
74	4A	RF	84	54	ET	94	5E		104	68	74
75	4B	SS	85	55	EN	95	5F		105	69	75
76	4C	ST	86	56	SO	96	60		106	6A	76
77	4D	ET	87	57	SB	97	61		107	6B	77
78	4E	EN	88	58	RF	98	62		108	6C	78
79	4F	SO	89	59	SS	99	63		109	6D	79
80	50	SB	90	5A	ST	100	64		110	6E	80
81	51	RF	91	5B	ET	110	6E		111	6F	81
82	52	SS	92	5C	EN	111	6F		112	70	82
83	53	ST	93	5D	SO	112	70		113	71	83
84	54	ET	94	5E	SB	113	71		114	72	84
85	55	EN	95	5F	RF	114	72		115	73	85
86	56	SO	96	60	SS	115	73		116	74	86
87	57	SB	97	61	ST	116	74		117	75	87
88	58	RF	98	62	ET	117	75		118	76	88
89	59	SS	99	63	EN	118	76		119	77	89
90	5A	ST	100	64	SO	119	77		120	78	90
91	5B	ET	101	65	SB	120	78		121	79	91
92	5C	EN	102	66	RF	121	79		122	7A	92
93	5D	SO	103	67	SS	122	7A		123	7B	93
94	5E	SB	104	68	ST	123	7B		124	7C	94
95	5F	RF	105	69	ET	124	7C		125	7D	95
96	60	SS	106	6A	EN	125	7D		126	7E	96
97	61	ST	107	6B	SO	126	7E		127	7F	97
98	62	ET	108	6C	SB	127	7F				
99	63	EN	109	6D	RF						
100	64	SO	110	6E	SS						

- `ord(c)`: returns Unicode (ASCII) of the character.
- Example: `ord('a')` returns 97.
- `chr(x)`: returns the character whose Unicode is x.
- Example: `chr(97)` returns 'a'.

In Pairs or Triples...

Some review and some novel challenges:

```
1 #Predict what will be printed:
2
3 for c in range(65,90):
4     print(chr(c))
5
6 message = "I love Python"
7 newMessage = ""
8 for c in message:
9     print(ord(c)) #Print the Unicode of each number
10    print(chr(ord(c)+1)) #Print the next character
11    newMessage = newMessage + chr(ord(c)+1) #add to the new message
12 print("The coded message is", newMessage)
13
14 word = "zebra"
15 codedWord = ""
16 for ch in word:
17     offset = ord(ch) - ord('a') + 1 #how many letters past 'a'
18     wrap = offset % 26 #if larger than 26, wrap back to 0
19     newChar = chr(ord('a') + wrap) #compute the new letter
20     print(wrap, chr(ord('a') + wrap)) #print the wrap & new lett
21     codedWord = codedWord + newChar #add the newChar to the coded w
22
23 print("The coded word (with wrap) is", codedWord)
```

Python Tutor

```
1 #Predict what will be printed:
2
3 for c in range(65,90):
4     print(chr(c))
5
6 message = "I love Python"
7 newMessage = ""
8 for c in message:
9     print(ord(c))    #Print the Unicode of each number
10    print(chr(ord(c)+1))    #Print the next character
11    newMessage = newMessage + chr(ord(c)+1) #Add to the new message
12 print("The coded message is", newMessage)
13
14 word = "zebra"
15 codedWord = ""
16 for ch in word:
17     offset = ord(ch) - ord('a') + 1 #how many letters past 'a'
18     wrap = offset % 26 #if larger than 26, wrap back to 0
19     newChar = chr(ord('a') + wrap) #compute the new letter
20     print(wrap, chr(ord('a') + wrap)) #print the wrap & new lett
21     codedWord = codedWord + newChar #add the newChar to the coded w
22
23 print("The coded word (with wrap) is", codedWord)
```

(Demo with pythonTutor)

User Input

Covered in detail in Lab 2:

```
→ 1 mess = input('Please enter a message: ')\n   2 print("You entered", mess)
```

(Demo with pythonTutor)

Side Note: '+' for numbers and strings



- `x = 3 + 5` stores the number 8 in memory location `x`.

Side Note: '+' for numbers and strings



- $x = 3 + 5$ stores the number 8 in memory location x .
- $x = x + 1$ increases x by 1.

Side Note: '+' for numbers and strings



- `x = 3 + 5` stores the number 8 in memory location `x`.
- `x = x + 1` increases `x` by 1.
- `s = "hi" + "Mom"` stores "hiMom" in memory locations `s`.

Side Note: '+' for numbers and strings



- `x = 3 + 5` stores the number 8 in memory location `x`.
- `x = x + 1` increases `x` by 1.
- `s = "hi" + "Mom"` stores "hiMom" in memory locations `s`.
- `s = s + "A"` adds the letter `x` to the end of the strings `s`.

Recap

- For-loops



Recap



- For-loops
- `range()`

Recap



- For-loops
- `range()`
- Variables: ints and strings

Recap



- For-loops
- `range()`
- Variables: ints and strings
- Some arithmetic

Recap



- For-loops
- `range()`
- Variables: ints and strings
- Some arithmetic
- String concatenation

Recap



- For-loops
- `range()`
- Variables: ints and strings
- Some arithmetic
- String concatenation
- Functions: `ord()` and `chr()`