

# CSci 127: Introduction to Computer Science



[hunter.cuny.edu/csci](http://hunter.cuny.edu/csci)

# Today's Topics



- For-loops
- range()
- Variables
- Characters
- Strings

# Today's Topics



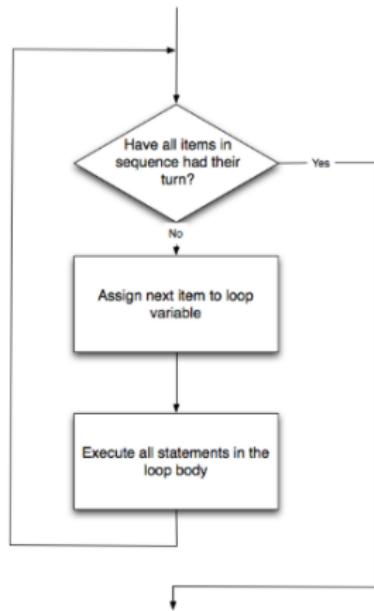
- **For-loops**
- `range()`
- Variables
- Characters
- Strings

## Group Work: predict what will be printed

```
1  for i in range(4):
2      print('The world turned upside down')
3  for j in [0,1,2,3,4,5]:
4      print(j)
5  for count in range(6):
6      print(count)
7  for color in ['red', 'green', 'blue']:
8      print(color)
9  for i in range(2):
10     for j in range(2):
11         print('Look around, ')
12     print('How lucky we are to be alive!')
```

[link to program](#)

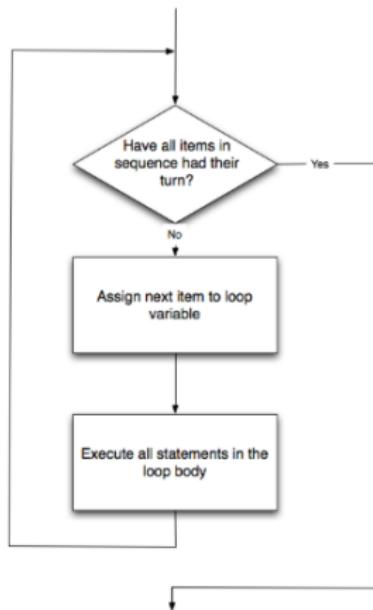
# for-loop



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

*How to Think Like CS, §4.5*

# for-loop



```
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()`.

*How to Think Like CS, §4.5*

# Today's Topics



- For-loops
- `range()`
- Variables
- Characters
- Strings

## More on range(): predict what will be printed

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

[link to range demo](#)

# 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:  $[0,1,2,3,\dots,stop-1]$
- For example, if you want the list  $[0,1,2,3,\dots,100]$ , you would write:

# range()



Simplest version:

- `range(stop)`
- Produces a list:  $[0,1,2,3,\dots,stop-1]$
- For example, if you want the list  $[0,1,2,3,\dots,100]$ , 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 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 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  $\text{start}+k*\text{step}$  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 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:  
 $[start, start+step, start+2*step\dots, last]$   
(where last is the largest  $start+k*step$  less than stop)
- For example, if you want 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)`

# Today's Topics



- For-loops
- `range()`
- **Variables**
- Characters
- Strings

# 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
  - ▶ **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
    - e.g. [3, 1, 4, 5, 9] or
    - ['violet', 'purple', 'indigo']
  - ▶ **class variables**: for complex objects, like turtles.
- In Python (unlike other languages) you don't need to specify the type; it is deduced by its value.

# 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.
- Can't use some words that Python has reserved for itself (e.g. `for`).  
(List of reserved words in *Think CS*, §2.5.)

# Today's Topics



- For-loops
- `range()`
- Variables
- **Characters**
- Strings

# 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	[END 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:

*(There is a link to the ASCII table on the course webpage, under 'Useful Links'.)*

ASCII TABLE		Decimal		Hex Char		Octal Char		Binary		Hex Char		Octal Char		Binary	
Decimal	Char	Dec	Hex	Oct	Bin	Dec	Hex	Oct	Bin	Dec	Hex	Oct	Bin	Dec	Hex
0	\0	0	0000	0	00000000	0	0000	0	00000000	0	0000	0	00000000	0	00000000
1	\1	1	0001	1	00000001	1	0001	1	00000001	1	0001	1	00000001	1	00000001
2	\2	2	0010	2	00000010	2	0010	2	00000010	2	0010	2	00000010	2	00000010
3	\3	3	0011	3	00000011	3	0011	3	00000011	3	0011	3	00000011	3	00000011
4	\4	4	0100	4	00000100	4	0100	4	00000100	4	0100	4	00000100	4	00000100
5	\5	5	0101	5	00000101	5	0101	5	00000101	5	0101	5	00000101	5	00000101
6	\6	6	0110	6	00000110	6	0110	6	00000110	6	0110	6	00000110	6	00000110
7	\7	7	0111	7	00000111	7	0111	7	00000111	7	0111	7	00000111	7	00000111
8	\8	8	1000	8	00001000	8	1000	8	00001000	8	1000	8	00001000	8	00001000
9	\9	9	1001	9	00001001	9	1001	9	00001001	9	1001	9	00001001	9	00001001
10	\a	10	1010	10	00001010	10	1010	10	00001010	10	1010	10	00001010	10	00001010
11	\b	11	1011	11	00001011	11	1011	11	00001011	11	1011	11	00001011	11	00001011
12	\c	12	1100	12	00001100	12	1100	12	00001100	12	1100	12	00001100	12	00001100
13	\d	13	1101	13	00001101	13	1101	13	00001101	13	1101	13	00001101	13	00001101
14	\e	14	1110	14	00001110	14	1110	14	00001110	14	1110	14	00001110	14	00001110
15	\f	15	1111	15	00001111	15	1111	15	00001111	15	1111	15	00001111	15	00001111
16	\n	16	2000	16	00010000	16	2000	16	00010000	16	2000	16	00010000	16	00010000
17	\v	17	2001	17	00010001	17	2001	17	00010001	17	2001	17	00010001	17	00010001
18	\t	18	2010	18	00010010	18	2010	18	00010010	18	2010	18	00010010	18	00010010
19	\r	19	2011	19	00010011	19	2011	19	00010011	19	2011	19	00010011	19	00010011
20	\b	20	2100	20	00010100	20	2100	20	00010100	20	2100	20	00010100	20	00010100
21	\f	21	2101	21	00010101	21	2101	21	00010101	21	2101	21	00010101	21	00010101
22	\n	22	2110	22	00010110	22	2110	22	00010110	22	2110	22	00010110	22	00010110
23	\v	23	2111	23	00010111	23	2111	23	00010111	23	2111	23	00010111	23	00010111
24	\012	24	2200	24	00011000	24	2200	24	00011000	24	2200	24	00011000	24	00011000
25	\015	25	2201	25	00011001	25	2201	25	00011001	25	2201	25	00011001	25	00011001
26	\016	26	2210	26	00011010	26	2210	26	00011010	26	2210	26	00011010	26	00011010
27	\017	27	2211	27	00011011	27	2211	27	00011011	27	2211	27	00011011	27	00011011
28	\020	28	2220	28	00011100	28	2220	28	00011100	28	2220	28	00011100	28	00011100
29	\021	29	2221	29	00011101	29	2221	29	00011101	29	2221	29	00011101	29	00011101
30	\022	30	2222	30	00011110	30	2222	30	00011110	30	2222	30	00011110	30	00011110
31	\023	31	2223	31	00011111	31	2223	31	00011111	31	2223	31	00011111	31	00011111
32	\024	32	2300	32	00011111	32	2300	32	00011111	32	2300	32	00011111	32	00011111
33	\025	33	2301	33	00011111	33	2301	33	00011111	33	2301	33	00011111	33	00011111
34	\026	34	2310	34	00011111	34	2310	34	00011111	34	2310	34	00011111	34	00011111
35	\027	35	2311	35	00011111	35	2311	35	00011111	35	2311	35	00011111	35	00011111
36	\030	36	2320	36	00011111	36	2320	36	00011111	36	2320	36	00011111	36	00011111
37	\031	37	2321	37	00011111	37	2321	37	00011111	37	2321	37	00011111	37	00011111
38	\032	38	2322	38	00011111	38	2322	38	00011111	38	2322	38	00011111	38	00011111
39	\033	39	2323	39	00011111	39	2323	39	00011111	39	2323	39	00011111	39	00011111
40	\034	40	2330	40	00011111	40	2330	40	00011111	40	2330	40	00011111	40	00011111
41	\035	41	2331	41	00011111	41	2331	41	00011111	41	2331	41	00011111	41	00011111
42	\036	42	2332	42	00011111	42	2332	42	00011111	42	2332	42	00011111	42	00011111
43	\037	43	2333	43	00011111	43	2333	43	00011111	43	2333	43	00011111	43	00011111
44	\040	44	2400	44	00011111	44	2400	44	00011111	44	2400	44	00011111	44	00011111
45	\041	45	2401	45	00011111	45	2401	45	00011111	45	2401	45	00011111	45	00011111
46	\042	46	2410	46	00011111	46	2410	46	00011111	46	2410	46	00011111	46	00011111
47	\043	47	2411	47	00011111	47	2411	47	00011111	47	2411	47	00011111	47	00011111
48	\044	48	2420	48	00011111	48	2420	48	00011111	48	2420	48	00011111	48	00011111
49	\045	49	2421	49	00011111	49	2421	49	00011111	49	2421	49	00011111	49	00011111
50	\046	50	2422	50	00011111	50	2422	50	00011111	50	2422	50	00011111	50	00011111
51	\047	51	2423	51	00011111	51	2423	51	00011111	51	2423	51	00011111	51	00011111
52	\050	52	2430	52	00011111	52	2430	52	00011111	52	2430	52	00011111	52	00011111
53	\051	53	2431	53	00011111	53	2431	53	00011111	53	2431	53	00011111	53	00011111
54	\052	54	2432	54	00011111	54	2432	54	00011111	54	2432	54	00011111	54	00011111
55	\053	55	2433	55	00011111	55	2433	55	00011111	55	2433	55	00011111	55	00011111
56	\054	56	2500	56	00011111	56	2500	56	00011111	56	2500	56	00011111	56	00011111
57	\055	57	2501	57	00011111	57	2501	57	00011111	57	2501	57	00011111	57	00011111
58	\056	58	2510	58	00011111	58	2510	58	00011111	58	2510	58	00011111	58	00011111
59	\057	59	2511	59	00011111	59	2511	59	00011111	59	2511	59	00011111	59	00011111
60	\060	60	2520	60	00011111	60	2520	60	00011111	60	2520	60	00011111	60	00011111
61	\061	61	2521	61	00011111	61	2521	61	00011111	61	2521	61	00011111	61	00011111
62	\062	62	2522	62	00011111	62	2522	62	00011111	62	2522	62	00011111	62	00011111
63	\063	63	2523	63	00011111	63	2523	63	00011111	63	2523	63	00011111	63	00011111
64	\064	64	2530	64	00011111	64	2530	64	00011111	64	2530	64	00011111	64	00011111
65	\065	65	2531	65	00011111	65	2531	65	00011111	65	2531	65	00011111	65	00011111
66	\066	66	2532	66	00011111	66	2532	66	00011111	66	2532	66	00011111	66	00011111
67	\067	67	2533	67	00011111	67	2533	67	00011111	67	2533	67	00011111	67	00011111
68	\070	68	2600	68	00011111	68	2600	68	00011111	68	2600	68	00011111	68	00011111
69	\071	69	2601	69	00011111	69	2601	69	00011111	69	2601	69	00011111	69	00011111
70	\072	70	2610	70	00011111	70	2610	70	00011111	70	2610	70	00011111	70	00011111
71	\073	71	2611	71	00011111	71	2611	71	00011111	71	2611	71	00011111	71	00011111
72	\074	72	2620	72	00011111	72	2620	72	00011111	72	2620	72	00011111	72	00011111
73	\075	73	2621	73	00011111	73	2621	73	00011111	73	2621	73	00011111	73	00011111
74	\076	74	2622	74	00011111	74	2622	74	00011111	74	2622	74	00011111	74	00011111
75	\077	75	2623	75	00011111	75	2623	75	00011111	75	2623	75	00011111	75	00011111
76	\080	76	2630	76	00011111	76	2630	76	00011111	76	2630	76	00011111	76	00011111
77	\081	77	2631	77	00011111	77	2631	77	00011111	77	2631	77	00011111	77	00011111
78	\082	78	2632	78	00011111	78	2632	78	00011111	78	2632	78	00011111	78	00011111
79	\083	79	2633	79	00011111	79	2633	79	00011111	79	2633	79	00011111	79	00011111
80	\084	80	2700												

# Converting from Character to Code:

*(There is a link to the ASCII table on the course webpage, under 'Useful Links'.)*

ASCII TABLE	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
	0	00		128	80		255	FF	
	1	01		129	81		256	00	
	2	02		130	82		257	01	
	3	03		131	83		258	02	
	4	04		132	84		259	03	
	5	05		133	85		260	04	
	6	06		134	86		261	05	
	7	07		135	87		262	06	
	8	08		136	88		263	07	
	9	09		137	89		264	08	
	10	0A		138	8A		265	09	
	11	0B		139	8B		266	0A	
	12	0C		140	8C		267	0B	
	13	0D		141	8D		268	0C	
	14	0E		142	8E		269	0D	
	15	0F		143	8F		270	0E	
	16	10		144	90		271	0F	
	17	11		145	91		272	10	
	18	12		146	92		273	11	
	19	13		147	93		274	12	
	20	14		148	94		275	13	
	21	15		149	95		276	14	
	22	16		150	96		277	15	
	23	17		151	97		278	16	
	24	18		152	98		279	17	
	25	19		153	99		280	18	
	26	1A		154	9A		281	19	
	27	1B		155	9B		282	1A	
	28	1C		156	9C		283	1B	
	29	1D		157	9D		284	1C	
	30	1E		158	9E		285	1D	
	31	1F		159	9F		286	1E	
	32	20		160	A0		287	1F	
	33	21		161	A1		288	20	
	34	22		162	A2		289	21	
	35	23		163	A3		290	22	
	36	24		164	A4		291	23	
	37	25		165	A5		292	24	
	38	26		166	A6		293	25	
	39	27		167	A7		294	26	
	40	28		168	A8		295	27	
	41	29		169	A9		296	28	
	42	2A		170	AA		297	29	
	43	2B		171	AB		298	2A	
	44	2C		172	AC		299	2B	
	45	2D		173	AD		300	2C	
	46	2E		174	AE		301	2D	
	47	2F		175	AF		302	2E	
	48	30		176	B0		303	2F	
	49	31		177	B1		304	30	
	50	32		178	B2		305	31	
	51	33		179	B3		306	32	
	52	34		180	B4		307	33	
	53	35		181	B5		308	34	
	54	36		182	B6		309	35	
	55	37		183	B7		310	36	
	56	38		184	B8		311	37	
	57	39		185	B9		312	38	
	58	3A		186	BA		313	39	
	59	3B		187	BB		314	3A	
	60	3C		188	BC		315	3B	
	61	3D		189	BD		316	3C	
	62	3E		190	BE		317	3D	
	63	3F		191	BF		318	3E	
	64	40		192	C0		319	3F	
	65	41		193	C1		320	40	
	66	42		194	C2		321	41	
	67	43		195	C3		322	42	
	68	44		196	C4		323	43	
	69	45		197	C5		324	44	
	70	46		198	C6		325	45	
	71	47		199	C7		326	46	
	72	48		200	C8		327	47	
	73	49		201	C9		328	48	
	74	4A		202	CA		329	49	
	75	4B		203	CB		330	4A	
	76	4C		204	CC		331	4B	
	77	4D		205	CD		332	4C	
	78	4E		206	CE		333	4D	
	79	4F		207	CF		334	4E	
	80	50		208	D0		335	4F	
	81	51		209	D1		336	50	
	82	52		210	D2		337	51	
	83	53		211	D3		338	52	
	84	54		212	D4		339	53	
	85	55		213	D5		340	54	
	86	56		214	D6		341	55	
	87	57		215	D7		342	56	
	88	58		216	D8		343	57	
	89	59		217	D9		344	58	
	90	5A		218	DA		345	59	
	91	5B		219	DB		346	5A	
	92	5C		220	DC		347	5B	
	93	5D		221	DD		348	5C	
	94	5E		222	DE		349	5D	
	95	5F		223	DF		350	5E	
	96	60		224	E0		351	5F	
	97	61		225	E1		352	60	
	98	62		226	E2		353	61	
	99	63		227	E3		354	62	
	100	64		228	E4		355	63	
	101	65		229	E5		356	64	
	102	66		230	E6		357	65	
	103	67		231	E7		358	66	
	104	68		232	E8		359	67	
	105	69		233	E9		360	68	
	106	6A		234	EA		361	69	
	107	6B		235	EB		362	6A	
	108	6C		236	EC		363	6B	
	109	6D		237	ED		364	6C	
	110	6E		238	EE		365	6D	
	111	6F		239	EF		366	6E	
	112	70		240	F0		367	6F	
	113	71		241	F1		368	70	
	114	72		242	F2		369	71	
	115	73		243	F3		370	72	
	116	74		244	F4		371	73	
	117	75		245	F5		372	74	
	118	76		246	F6		373	75	
	119	77		247	F7		374	76	
	120	78		248	F8		375	77	
	121	79		249	F9		376	78	
	122	7A		250	FA		377	79	
	123	7B		251	FB		378	7A	
	124	7C		252	FC		379	7B	
	125	7D		253	FD		380	7C	
	126	7E		254	FE		381	7D	
	127	7F		255	FF		382	7E	

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

# Converting from Character to Code:

*(There is a link to the ASCII table on the course webpage, under 'Useful Links'.)*

ASCII TABLE	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
	0	00		1	01	!>	2	02	!>
	2	02		3	03	!>	4	04	!>
	3	03		5	05	!>	6	06	!>
	5	05		7	07	!>	8	08	!>
	7	07		9	09	!>	10	0A	!>
	9	09		11	0B	!>	12	0C	!>
	11	0B		13	0D	!>	14	0E	!>
	13	0D		15	0F	!>	16	10	!>
	15	0F		17	11	!>	18	12	!>
	17	11		19	13	!>	20	14	!>
	19	13		21	15	!>	22	16	!>
	21	15		23	17	!>	24	18	!>
	23	17		25	1A	!>	26	1B	!>
	25	1A		27	1C	!>	28	1D	!>
	27	1C		29	1E	!>	30	1F	!>
	29	1E		31	20	!>	32	21	!>
	31	20		33	23	!>	34	24	!>
	33	23		35	26	!>	36	27	!>
	35	26		37	29	!>	38	2A	!>
	37	29		39	2B	!>	40	2C	!>
	39	2B		41	2D	!>	42	2E	!>
	41	2D		43	2F	!>	44	30	!>
	43	2F		45	32	!>	46	33	!>
	45	32		47	35	!>	48	36	!>
	47	35		49	38	!>	50	39	!>
	49	38		51	3B	!>	52	3C	!>
	51	3B		53	3E	!>	54	3F	!>
	53	3E		55	40	!>	56	41	!>
	55	40		57	43	!>	58	44	!>
	57	43		59	46	!>	60	47	!>
	59	46		61	49	!>	62	4A	!>
	61	49		63	4B	!>	64	4C	!>
	63	4B		65	4D	!>	66	4E	!>
	65	4D		67	4F	!>	68	50	!>
	67	4F		69	52	!>	70	53	!>
	69	52		71	55	!>	72	56	!>
	71	55		73	58	!>	74	59	!>
	73	58		75	5B	!>	76	5C	!>
	75	5B		77	5E	!>	78	5F	!>
	77	5E		79	60	!>	80	61	!>
	79	60		81	63	!>	82	64	!>
	81	63		83	66	!>	84	67	!>
	83	66		85	69	!>	86	6A	!>
	85	69		87	6B	!>	88	6C	!>
	87	6B		89	6D	!>	90	6E	!>
	89	6D		91	6F	!>	92	70	!>
	91	6F		93	73	!>	94	74	!>
	93	73		95	76	!>	96	77	!>
	95	76		97	79	!>	98	7A	!>
	97	79		99	7B	!>	100	7C	!>
	99	7B		101	7D	!>	102	7E	!>
	101	7D		103	7F	!>	104	80	!>
	103	7F		105	83	!>	106	84	!>
	105	83		107	86	!>	108	87	!>
	107	86		109	89	!>	110	8A	!>
	109	89		111	8B	!>	112	8C	!>
	111	8B		113	8D	!>	114	8E	!>
	113	8D		115	8F	!>	116	90	!>
	115	8F		117	93	!>	118	94	!>
	117	93		119	96	!>	120	97	!>
	119	96		121	99	!>	122	9A	!>
	121	99		123	9B	!>	124	9C	!>
	123	9B		125	9D	!>	126	9E	!>
	125	9D		127	9F	!>	128	A0	!>
	127	9F		129	A3	!>	130	A4	!>
	129	A3		131	A6	!>	132	A7	!>
	131	A6		133	A9	!>	134	AA	!>
	133	A9		135	AB	!>	136	AC	!>
	135	AB		137	AD	!>	138	AE	!>
	137	AD		139	AF	!>	140	B0	!>
	139	AF		141	B3	!>	142	B4	!>
	141	B3		143	B6	!>	144	B7	!>
	143	B6		145	B9	!>	146	BA	!>
	145	B9		147	BB	!>	148	BC	!>
	147	BB		149	BD	!>	150	BE	!>
	149	BD		151	BF	!>	152	C0	!>
	151	BF		153	C3	!>	154	C4	!>
	153	C3		155	C6	!>	156	C7	!>
	155	C6		157	C9	!>	158	CA	!>
	157	C9		159	CB	!>	160	CC	!>
	159	CB		161	CD	!>	162	CE	!>
	161	CD		163	CF	!>	164	D0	!>
	163	CF		165	D3	!>	166	D4	!>
	165	D3		167	D6	!>	168	D7	!>
	167	D6		169	D9	!>	170	DA	!>
	169	D9		171	DB	!>	172	DC	!>
	171	DB		173	DD	!>	174	DE	!>
	173	DD		175	DF	!>	176	E0	!>
	175	DF		177	E3	!>	178	E4	!>
	177	E3		179	E6	!>	180	E7	!>
	179	E6		181	E9	!>	182	EA	!>
	181	E9		183	EB	!>	184	EC	!>
	183	EB		185	ED	!>	186	EE	!>
	185	ED		187	FF	!>	188	00	!>
	187	FF		189	03	!>	190	04	!>
	189	03		191	06	!>	192	07	!>
	191	06		193	09	!>	194	0A	!>
	193	09		195	0B	!>	196	0C	!>
	195	0B		197	0D	!>	198	0E	!>
	197	0D		199	0F	!>	200	10	!>
	199	0F		201	13	!>	202	14	!>
	201	13		203	16	!>	204	17	!>
	203	16		205	19	!>	206	1A	!>
	205	19		207	1B	!>	208	1C	!>
	207	1B		209	1D	!>	210	1E	!>
	209	1D		211	1F	!>	212	20	!>
	211	1F		213	23	!>	214	24	!>
	213	23		215	26	!>	216	27	!>
	215	26		217	29	!>	218	2A	!>
	217	29		219	2B	!>	220	2C	!>
	219	2B		221	2D	!>	222	2E	!>
	221	2D		223	2F	!>	224	30	!>
	223	2F		225	33	!>	226	34	!>
	225	33		227	36	!>	228	37	!>
	227	36		229	39	!>	230	3A	!>
	229	39		231	3B	!>	232	3C	!>
	231	3B		233	3D	!>	234	3E	!>
	233	3D		235	3F	!>	236	40	!>
	235	3F		237	43	!>	238	44	!>
	237	43		239	46	!>	240	47	!>
	239	46		241	49	!>	242	4A	!>
	241	49		243	4B	!>	244	4C	!>
	243	4B		245	4D	!>	246	4E	!>
	245	4D		247	4F	!>	248	50	!>
	247	4F		249	53	!>	250	54	!>
	249	53		251	56	!>	252	57	!>
	251	56		253	59	!>	254	5A	!>
	253	59		255	5B	!>	256	5C	!>
	255	5B		257	5D	!>	258	5E	!>
	257	5D		259	5F	!>	260	60	!>
	259	5F		261	63	!>	262	64	!>
	261	63		263	66	!>	264	67	!>
	263	66		265	69	!>	266	6A	!>
	265	69		267	6B	!>	268	6C	!>
	267	6B		269	6D	!>	270	6E	!>
	269	6D		271	6F	!>	272	70	!>
	271	6F		273	73	!>	274	74	!>
	273	73		275	76	!>	276	77	!>
	275	76		277	79	!>	278	7A	!>
	277	79		279	7B	!>	280	7C	!>
	279	7B		281	7D	!>	282	7E	!>
	281	7D		283	7F	!>	284	80	!>
	283	7F		285	83	!>	286	84	!>
	285	83		287	86	!>	288	87	!>
	287	86		289	89	!>	290	8A	!>
	289	89		291	8B	!>	292	8C	!>
	291	8B		293	8D	!>	294	8E	!>
	293	8D		295	8F	!>	296	90	!>
	295	8F		297	93	!>	298	94	!>
	297	93		299	96	!>	300	97	!>
	299	96		301	99	!>	302	9A	!>
	301	99		303	9B	!>	304	9C	!>
	303	9B		305	9D	!>	306	9E	!>
	305	9D		307	9F	!>	308	A0	!>
	307	9F		309	A3	!>	310	A4	!>
	309	A3		311	A6	!>	312	A7	!>
	311	A6		313	A9	!>	314	AA	!>
	313	A9		315	AB	!>	316	AC	!>
	315	AB		31					

# Converting from Character to Code:

*(There is a link to the ASCII table on the course webpage, under 'Useful Links'.)*

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	00	\0	32	20	\t	64	40	\n
1	01	\1	33	21	\a	65	41	A
2	02	\2	34	22	\b	66	42	B
3	03	\3	35	23	\f	67	43	F
4	04	\4	36	24	\n	68	44	\n
5	05	\5	37	25	\r	69	45	\r
6	06	\6	38	26	\v	70	46	\v
7	07	\7	39	27	\b	71	47	\b
8	08	\8	40	28	\t	72	48	\t
9	09	\9	41	29	\n	73	49	\n
10	0A	\10	42	2A	\r	74	4A	\r
11	0B	\11	43	2B	\v	75	4B	\v
12	0C	\12	44	2C	\b	76	4C	\b
13	0D	\13	45	2D	\t	77	4D	\t
14	0E	\14	46	2E	\n	78	4E	\n
15	0F	\15	47	2F	\r	79	4F	\r
16	10	\16	48	30	\t	80	50	\t
17	11	\17	49	31	\n	81	51	\n
18	12	\18	4A	32	\r	82	52	\r
19	13	\19	4B	33	\v	83	53	\v
20	14	\20	4C	34	\b	84	54	\b
21	15	\21	4D	35	\t	85	55	\t
22	16	\22	4E	36	\n	86	56	\n
23	17	\23	4F	37	\r	87	57	\r
24	18	\24	50	38	\v	88	58	\v
25	19	\25	51	39	\b	89	59	\b
26	1A	\26	52	3A	\t	90	5A	\t
27	1B	\27	53	3B	\n	91	5B	\n
28	1C	\28	54	3C	\r	92	5C	\r
29	1D	\29	55	3D	\v	93	5D	\v
30	1E	\2A	56	3E	\b	94	5E	\b
31	1F	\2B	57	3F	\t	95	5F	\t
32	20	\2C	58	40	\n	96	60	\n
33	21	\2D	59	41	\r	97	61	\r
34	22	\2E	5A	42	\v	98	62	\v
35	23	\2F	5B	43	\b	99	63	\b
36	24	\30	5C	44	\t	100	64	\t
37	25	\31	5D	45	\n	101	65	\n
38	26	\32	5E	46	\r	102	66	\r
39	27	\33	5F	47	\v	103	67	\v
40	28	\34	60	48	\b	104	68	\b
41	29	\35	61	49	\t	105	69	\t
42	2A	\36	62	4A	\n	106	6A	\n
43	2B	\37	63	4B	\r	107	6B	\r
44	2C	\38	64	4C	\v	108	6C	\v
45	2D	\39	65	4D	\b	109	6D	\b
46	2E	\3A	66	4E	\t	110	6E	\t
47	2F	\3B	67	4F	\n	111	6F	\n
48	30	\3C	68	50	\r	112	70	\r
49	31	\3D	69	51	\v	113	71	\v
50	32	\3E	6A	52	\b	114	72	\b
51	33	\3F	6B	53	\t	115	73	\t
52	34	\30	6C	54	\n	116	74	\n
53	35	\31	6D	55	\r	117	75	\r
54	36	\32	6E	56	\v	118	76	\v
55	37	\33	6F	57	\b	119	77	\b
56	38	\34	70	58	\t	120	78	\t
57	39	\35	71	59	\n	121	79	\n
58	3A	\36	72	5A	\r	122	7A	\r
59	3B	\37	73	5B	\v	123	7B	\v
60	3C	\38	74	5C	\b	124	7C	\b
61	3D	\39	75	5D	\t	125	7D	\t
62	3E	\3A	76	5E	\n	126	7E	\n
63	3F	\3B	77	5F	\r	127	7F	\r
64	40	\3C	78	60	\v	128	80	\v
65	41	\3D	79	61	\b	129	81	\b
66	42	\3E	7A	62	\t	130	82	\t
67	43	\3F	7B	63	\n	131	83	\n
68	44	\30	7C	64	\r	132	84	\r
69	45	\31	7D	65	\v	133	85	\v
70	46	\32	7E	66	\b	134	86	\b
71	47	\33	7F	67	\t	135	87	\t
72	48	\34	80	68	\n	136	88	\n
73	49	\35	81	69	\r	137	89	\r
74	4A	\36	82	6A	\v	138	8A	\v
75	4B	\37	83	6B	\b	139	8B	\b
76	4C	\38	84	6C	\t	140	8C	\t
77	4D	\39	85	6D	\n	141	8D	\n
78	4E	\3A	86	6E	\r	142	8E	\r
79	4F	\3B	87	6F	\v	143	8F	\v
80	50	\3C	88	70	\b	144	90	\b
81	51	\3D	89	71	\t	145	91	\t
82	52	\3E	8A	72	\n	146	92	\n
83	53	\3F	8B	73	\r	147	93	\r
84	54	\30	8C	74	\v	148	94	\v
85	55	\31	8D	75	\b	149	95	\b
86	56	\32	8E	76	\t	150	96	\t
87	57	\33	8F	77	\n	151	97	\n
88	58	\34	90	78	\r	152	98	\r
89	59	\35	91	79	\v	153	99	\v
90	5A	\36	92	7A	\b	154	9A	\b
91	5B	\37	93	7B	\t	155	9B	\t
92	5C	\38	94	7C	\n	156	9C	\n
93	5D	\39	95	7D	\r	157	9D	\r
94	5E	\3A	96	7E	\v	158	9E	\v
95	5F	\3B	97	7F	\b	159	9F	\b
96	60	\3C	98	80	\t	160	100	\t
97	61	\3D	99	81	\n	161	101	\n
98	62	\3E	9A	82	\r	162	102	\r
99	63	\3F	9B	83	\v	163	103	\v
100	64	\30	9C	84	\b	164	104	\b
101	65	\31	9D	85	\t	165	105	\t
102	66	\32	9E	86	\n	166	106	\n
103	67	\33	9F	87	\r	167	107	\r
104	68	\34	100	88	\v	168	108	\v
105	69	\35	101	89	\b	169	109	\b
106	6A	\36	102	8A	\t	170	10A	\t
107	6B	\37	103	8B	\n	171	10B	\n
108	6C	\38	104	8C	\r	172	10C	\r
109	6D	\39	105	8D	\v	173	10D	\v
110	6E	\3A	106	8E	\b	174	10E	\b
111	6F	\3B	107	8F	\t	175	10F	\t
112	70	\3C	108	90	\n	176	110	\n
113	71	\3D	109	91	\r	177	111	\r
114	72	\3E	110	92	\v	178	112	\v
115	73	\3F	111	93	\b	179	113	\b
116	74	\30	112	94	\t	180	114	\t
117	75	\31	113	95	\n	181	115	\n
118	76	\32	114	96	\r	182	116	\r
119	77	\33	115	97	\v	183	117	\v
120	78	\34	116	98	\b	184	118	\b
121	79	\35	117	99	\t	185	119	\t
122	7A	\36	118	100	\n	186	120	\n
123	7B	\37	119	101	\r	187	121	\r
124	7C	\38	120	102	\v	188	122	\v
125	7D	\39	121	103	\b	189	123	\b
126	7E	\3A	122	104	\t	190	124	\t
127	7F	\3B	123	105	\n	191	125	\n
128	80	\3C	124	106	\r	192	126	\r
129	81	\3D	125	107	\v	193	127	\v
130	82	\3E	126	108	\b	194	128	\b
131	83	\3F	127	109	\t	195	129	\t
132	84	\30	128	110	\n	196	130	\n
133	85	\31	129	111	\r	197	131	\r
134	86	\32	130	112	\v	198	132	\v
135	87	\33	131	113	\b	199	133	\b
136	88	\34	132	114	\t	200	134	\t
137	89	\35	133	115	\n	201	135	\n
138	8A	\36	134	116	\r	202	136	\r
139	8B	\37	135	117	\v	203	137	\v
140	8C	\38	136	118	\b	204	138	\b
141	8D	\39	137	119	\t	205	139	\t
142	8E	\3A	138	120	\n	206	140	\n
143	8F	\3B	139	121	\r	207	141	\r
144	90	\3C	140	122	\v	208	142	\v
145	91	\3D	141	123	\b	209	143	\b
146	92	\3E	142	124	\t	210	144	\t
147	93	\3F	143	125	\n	211	145	\n
148	94	\30	144	126	\r	212	146	\r
149	95	\31	145	127	\v	213	147	\v
150	96	\32	146	128	\b	214	148	\b
151	97	\33	147	129	\t	215	149	\t
152	98	\34	148	130	\n	216	150	\n
153	99	\35	149	131	\r	217	151	\r
154	9A	\36	150	132	\v	218	152	\v
155	9B	\37	151	133	\b	219	153	\b
156	9C	\38	152	134	\t	220	154	\t
157	9D	\39	153	135	\n	221	155	\n
158	9E	\3A	154	136	\r	222	156	\r
159	9F	\3B	155	137	\v	223	157	\v
160	100	\3C	156	138	\b	224	158	\b
161	101	\3D	157	139	\t	225	159	\t
162	102	\3E	158	140	\n	226	160	\n
163	103	\3F	159	141	\r	227	161	\r
164	104	\30	160	142	\v	228	162	\v
165	105	\31	161	143	\b	229	163	\b
166	106	\32	162	144	\t	230	164	\t
167	107	\33	163	145	\n	231	165	\n
168	108	\34	164	146	\r	232	166	\r
169	109	\35	165	147	\v	233	167	\v
170	110	\36	166	148	\b	234	168	\b
171	111	\37	167	149	\t	235	169	\t
172	112	\38	168	150	\n	236	170	\n
173	113	\39	169	151	\r	237	171	\r
174	114	\3A	170	152	\v	238	172	\v
175	115	\3B	171	153	\b	239	173	\b
176	116	\3C	172	154	\t	240	174	\t
177	117	\3D</						

# Converting from Character to Code:

*(There is a link to the ASCII table on the course webpage, under 'Useful Links'.)*

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	00	\0	32	20	\t	64	40	\n
1	01	\1	33	21	\a	65	41	A
2	02	\2	34	22	\b	66	42	B
3	03	\3	35	23	\f	67	43	F
4	04	\4	36	24	\n	68	44	\n
5	05	\5	37	25	\r	69	45	\r
6	06	\6	38	26	\v	70	46	\v
7	07	\7	39	27	\b	71	47	\b
8	08	\8	40	28	\t	72	48	\t
9	09	\9	41	29	\n	73	49	\n
10	0A	\10	42	2A	\f	74	4A	\f
11	0B	\11	43	2B	\r	75	4B	\r
12	0C	\12	44	2C	\v	76	4C	\v
13	0D	\13	45	2D	\b	77	4D	\b
14	0E	\14	46	2E	\t	78	4E	\t
15	0F	\15	47	2F	\n	79	4F	\n
16	10	\16	48	30	\t	80	50	\t
17	11	\17	49	31	\n	81	51	\n
18	12	\18	4A	32	\f	82	52	\f
19	13	\19	4B	33	\r	83	53	\r
20	14	\20	4C	34	\v	84	54	\v
21	15	\21	4D	35	\b	85	55	\b
22	16	\22	4E	36	\t	86	56	\t
23	17	\23	4F	37	\n	87	57	\n
24	18	\24	50	38	\t	88	58	\t
25	19	\25	51	39	\n	89	59	\n
26	1A	\26	52	3A	\f	90	5A	\f
27	1B	\27	53	3B	\r	91	5B	\r
28	1C	\28	54	3C	\v	92	5C	\v
29	1D	\29	55	3D	\b	93	5D	\b
30	1E	\20	56	3E	\t	94	5E	\t
31	1F	\21	57	3F	\n	95	5F	\n
32	20	\22	58	40	\t	96	60	\t
33	21	\23	59	41	\n	97	61	'a'
34	22	\24	5A	42	\f	98	62	'\f'
35	23	\25	5B	43	\r	99	63	'\r'
36	24	\26	5C	44	\v	100	64	'\v'
37	25	\27	5D	45	\b	101	65	'\b'
38	26	\28	5E	46	\t	102	66	'\t'
39	27	\29	5F	47	\n	103	67	'\n'
40	28	\20	60	48	\t	104	68	'\t'
41	29	\21	61	49	\n	105	69	'\n'
42	2A	\22	62	4A	\f	106	6A	'\f'
43	2B	\23	63	4B	\r	107	6B	'\r'
44	2C	\24	64	4C	\v	108	6C	'\v'
45	2D	\25	65	4D	\b	109	6D	'\b'
46	2E	\26	66	4E	\t	110	6E	'\t'
47	2F	\27	67	4F	\n	111	6F	'\n'
48	30	\28	68	50	\t	112	70	'\t'
49	31	\29	69	51	\n	113	71	'\n'
50	32	\20	6A	52	\f	114	72	'\f'
51	33	\21	6B	53	\r	115	73	'\r'
52	34	\22	6C	54	\v	116	74	'\v'
53	35	\23	6D	55	\b	117	75	'\b'
54	36	\24	6E	56	\t	118	76	'\t'
55	37	\25	6F	57	\n	119	77	'\n'
56	38	\26	70	58	\t	120	78	'\t'
57	39	\27	71	59	\n	121	79	'\n'
58	3A	\28	72	5A	\f	122	7A	'\f'
59	3B	\29	73	5B	\r	123	7B	'\r'
60	3C	\20	74	5C	\v	124	7C	'\v'
61	3D	\21	75	5D	\b	125	7D	'\b'
62	3E	\22	76	5E	\t	126	7E	'\t'
63	3F	\23	77	5F	\n	127	7F	'\n'
64	40	\24	78	60	\t	128	80	'\t'
65	41	\25	79	61	\n	129	81	'\n'
66	42	\26	7A	62	\f	130	82	'\f'
67	43	\27	7B	63	\r	131	83	'\r'
68	44	\28	7C	64	\v	132	84	'\v'
69	45	\29	7D	65	\b	133	85	'\b'
70	46	\20	7E	66	\t	134	86	'\t'
71	47	\21	7F	67	\n	135	87	'\n'
72	48	\22	80	68	\t	136	88	'\t'
73	49	\23	81	69	\n	137	89	'\n'
74	4A	\24	82	6A	\f	138	8A	'\f'
75	4B	\25	83	6B	\r	139	8B	'\r'
76	4C	\26	84	6C	\v	140	8C	'\v'
77	4D	\27	85	6D	\b	141	8D	'\b'
78	4E	\28	86	6E	\t	142	8E	'\t'
79	4F	\29	87	6F	\n	143	8F	'\n'
80	50	\20	88	70	\t	144	90	'\t'
81	51	\21	89	71	\n	145	91	'\n'
82	52	\22	8A	72	\f	146	92	'\f'
83	53	\23	8B	73	\r	147	93	'\r'
84	54	\24	8C	74	\v	148	94	'\v'
85	55	\25	8D	75	\b	149	95	'\b'
86	56	\26	8E	76	\t	150	96	'\t'
87	57	\27	8F	77	\n	151	97	'\n'
88	58	\28	90	78	\t	152	98	'\t'
89	59	\29	91	79	\n	153	99	'\n'
90	5A	\20	92	7A	\f	154	100	'\f'
91	5B	\21	93	7B	\r	155	101	'\r'
92	5C	\22	94	7C	\v	156	102	'\v'
93	5D	\23	95	7D	\b	157	103	'\b'
94	5E	\24	96	7E	\t	158	104	'\t'
95	5F	\25	97	7F	\n	159	105	'\n'
96	60	\26	98	80	\t	160	106	'\t'
97	61	\27	99	81	\n	161	107	'\n'
98	62	\28	9A	82	\f	162	108	'\f'
99	63	\29	9B	83	\r	163	109	'\r'
100	64	\20	9C	84	\v	164	110	'\v'
101	65	\21	9D	85	\b	165	111	'\b'
102	66	\22	9E	86	\t	166	112	'\t'
103	67	\23	9F	87	\n	167	113	'\n'
104	68	\24	90	88	\t	168	114	'\t'
105	69	\25	91	89	\n	169	115	'\n'
106	6A	\26	92	8A	\f	170	116	'\f'
107	6B	\27	93	8B	\r	171	117	'\r'
108	6C	\28	94	8C	\v	172	118	'\v'
109	6D	\29	95	8D	\b	173	119	'\b'
110	6E	\20	96	8E	\t	174	120	'\t'
111	6F	\21	97	8F	\n	175	121	'\n'
112	70	\22	98	90	\t	176	122	'\t'
113	71	\23	99	91	\n	177	123	'\n'
114	72	\24	9A	92	\f	178	124	'\f'
115	73	\25	9B	93	\r	179	125	'\r'
116	74	\26	9C	94	\v	180	126	'\v'
117	75	\27	9D	95	\b	181	127	'\b'
118	76	\28	9E	96	\t	182	128	'\t'
119	77	\29	9F	97	\n	183	129	'\n'
120	78	\20	90	98	\t	184	130	'\t'
121	79	\21	91	99	\n	185	131	'\n'
122	7A	\22	92	9A	\f	186	132	'\f'
123	7B	\23	93	9B	\r	187	133	'\r'
124	7C	\24	94	9C	\v	188	134	'\v'
125	7D	\25	95	9D	\b	189	135	'\b'
126	7E	\26	96	9E	\t	190	136	'\t'
127	7F	\27	97	9F	\n	191	137	'\n'
128	80	\28	98	90	\t	192	138	'\t'
129	81	\29	99	91	\n	193	139	'\n'
130	82	\20	9A	92	\f	194	140	'\f'
131	83	\21	9B	93	\r	195	141	'\r'
132	84	\22	9C	94	\v	196	142	'\v'
133	85	\23	9D	95	\b	197	143	'\b'
134	86	\24	9E	96	\t	198	144	'\t'
135	87	\25	9F	97	\n	199	145	'\n'
136	88	\26	90	98	\t	200	146	'\t'
137	89	\27	91	99	\n	201	147	'\n'
138	8A	\28	92	9A	\f	202	148	'\f'
139	8B	\29	93	9B	\r	203	149	'\r'
140	8C	\20	94	9C	\v	204	150	'\v'
141	8D	\21	95	9D	\b	205	151	'\b'
142	8E	\22	96	9E	\t	206	152	'\t'
143	8F	\23	97	9F	\n	207	153	'\n'
144	90	\24	98	90	\t	208	154	'\t'
145	91	\25	99	91	\n	209	155	'\n'
146	92	\26	9A	92	\f	210	156	'\f'
147	93	\27	9B	93	\r	211	157	'\r'
148	94	\28	9C	94	\v	212	158	'\v'
149	95	\29	9D	95	\b	213	159	'\b'
150	96	\20	9E	96	\t	214	160	'\t'
151	97	\21	9F	97	\n	215	161	'\n'
152	98	\22	90	98	\t	216	162	'\t'
153	99	\23	91	99	\n	217	163	'\n'
154	9A	\24	92	9A	\f	218	164	'\f'
155	9B	\25	93	9B	\r	219	165	'\r'
156	9C	\26	94	9C	\v	220	166	'\v'
157	9D	\27	95	9D	\b	221	167	'\b'
158	9E	\28	96	9E	\t	222	168	'\t'
159	9F	\29	97	9F	\n	223	169	'\n'
160	90	\20	98	90	\t	224	170	'\t'
161	91	\21	99	91	\n	225	171	'\n'
162	92	\22	9A	92	\f	226	172	'\f'
163	93	\23	9B	93	\r	227	173	'\r'
164	94	\24	9C	94	\v	228	174	'\v'
165	95	\25	9D	95	\b	229	175	'\b'
166	96	\26	9E	96	\t	230	176	'\t'
167	97	\27	9F	97	\n	231	177	'\n'
168	98	\28	90	98	\t	232	178	'\t'
169	99	\29	91	99	\n	233	179	'\n'
170	9A	\20	92	9A	\f	234	180	'\f'
171	9B	\21	93	9B	\r	235	181	'\r'
172	9C	\22	94	9C	\v	236	182	'\v'
173	9D	\23	95	9D	\b	237	183	'\b'
174	9E	\24	96	9E	\t	238	184	'\t'
175	9F	\25	97	9F	\n	239	185	'\n'
176	90	\26	98	90	\t	240	186	'\t'
177	91	\						

# Converting from Character to Code:

*(There is a link to the ASCII table on the course webpage, under 'Useful Links'.)*

Decimal Num Char	Octal Num Char	Hex Num Char	Decimal Num Char	Octal Num Char	Hex Num Char
'\000'	'000'	'000'	'\001'	'001'	'001'
'\002'	'002'	'002'	'\003'	'003'	'003'
'\004'	'004'	'004'	'\005'	'005'	'005'
'\006'	'006'	'006'	'\007'	'007'	'007'
'\010'	'010'	'00A'	'\011'	'011'	'00B'
'\012'	'012'	'00C'	'\013'	'013'	'00D'
'\014'	'014'	'00E'	'\015'	'015'	'00F'
'\016'	'016'	'010'	'\017'	'017'	'011'
'\020'	'020'	'012'	'\021'	'021'	'013'
'\022'	'022'	'014'	'\023'	'023'	'015'
'\024'	'024'	'016'	'\025'	'025'	'017'
'\026'	'026'	'018'	'\027'	'027'	'019'
'\030'	'030'	'01A'	'\031'	'031'	'01B'
'\032'	'032'	'01C'	'\033'	'033'	'01D'
'\034'	'034'	'01E'	'\035'	'035'	'01F'
'\036'	'036'	'020'	'\037'	'037'	'021'
'\040'	'040'	'022'	'\041'	'041'	'023'
'\042'	'042'	'024'	'\043'	'043'	'025'
'\044'	'044'	'026'	'\045'	'045'	'027'
'\046'	'046'	'028'	'\047'	'047'	'029'
'\048'	'048'	'02A'	'\049'	'049'	'02B'
'\050'	'050'	'02C'	'\051'	'051'	'02D'
'\052'	'052'	'02E'	'\053'	'053'	'02F'
'\054'	'054'	'030'	'\055'	'055'	'031'
'\056'	'056'	'032'	'\057'	'057'	'033'
'\060'	'060'	'034'	'\061'	'061'	'035'
'\062'	'062'	'036'	'\063'	'063'	'037'
'\064'	'064'	'038'	'\065'	'065'	'039'
'\066'	'066'	'03A'	'\067'	'067'	'03B'
'\068'	'068'	'03C'	'\069'	'069'	'03D'
'\070'	'070'	'03E'	'\071'	'071'	'03F'
'\072'	'072'	'040'	'\073'	'073'	'041'
'\074'	'074'	'042'	'\075'	'075'	'043'
'\076'	'076'	'044'	'\077'	'077'	'045'
'\080'	'080'	'048'	'\081'	'081'	'049'
'\082'	'082'	'04A'	'\083'	'083'	'04B'
'\084'	'084'	'04C'	'\085'	'085'	'04D'
'\086'	'086'	'04E'	'\087'	'087'	'04F'
'\088'	'088'	'050'	'\089'	'089'	'051'
'\090'	'090'	'052'	'\091'	'091'	'053'
'\092'	'092'	'054'	'\093'	'093'	'055'
'\094'	'094'	'056'	'\095'	'095'	'057'
'\096'	'096'	'058'	'\097'	'097'	'059'
'\098'	'098'	'05A'	'\099'	'099'	'05B'
'\09A'	'09A'	'05C'	'\09B'	'09B'	'05D'
'\09C'	'09C'	'05E'	'\09D'	'09D'	'05F'
'\09E'	'09E'	'060'	'\09F'	'09F'	'061'
'\0A0'	'0A0'	'062'	'\0A1'	'0A1'	'063'
'\0A2'	'0A2'	'064'	'\0A3'	'0A3'	'065'
'\0A4'	'0A4'	'066'	'\0A5'	'0A5'	'067'
'\0A6'	'0A6'	'068'	'\0A7'	'0A7'	'069'
'\0A8'	'0A8'	'06A'	'\0A9'	'0A9'	'06B'
'\0AA'	'0AA'	'06C'	'\0AB'	'0AB'	'06D'
'\0AC'	'0AC'	'06E'	'\0AD'	'0AD'	'06F'
'\0AE'	'0AE'	'070'	'\0AF'	'0AF'	'071'
'\0B0'	'0B0'	'072'	'\0B1'	'0B1'	'073'
'\0B2'	'0B2'	'074'	'\0B3'	'0B3'	'075'
'\0B4'	'0B4'	'076'	'\0B5'	'0B5'	'077'
'\0B6'	'0B6'	'078'	'\0B7'	'0B7'	'079'
'\0B8'	'0B8'	'07A'	'\0B9'	'0B9'	'07B'
'\0BA'	'0BA'	'07C'	'\0BB'	'0BB'	'07D'
'\0BC'	'0BC'	'07E'	'\0BD'	'0BD'	'07F'
'\0BE'	'0BE'	'080'	'\0BF'	'0BF'	'081'
'\0C0'	'0C0'	'082'	'\0C1'	'0C1'	'083'
'\0C2'	'0C2'	'084'	'\0C3'	'0C3'	'085'
'\0C4'	'0C4'	'086'	'\0C5'	'0C5'	'087'
'\0C6'	'0C6'	'088'	'\0C7'	'0C7'	'089'
'\0C8'	'0C8'	'08A'	'\0C9'	'0C9'	'08B'
'\0CA'	'0CA'	'08C'	'\0CB'	'0CB'	'08D'
'\0CC'	'0CC'	'08E'	'\0CD'	'0CD'	'08F'
'\0CE'	'0CE'	'090'	'\0CF'	'0CF'	'091'
'\0D0'	'0D0'	'092'	'\0D1'	'0D1'	'093'
'\0D2'	'0D2'	'094'	'\0D3'	'0D3'	'095'
'\0D4'	'0D4'	'096'	'\0D5'	'0D5'	'097'
'\0D6'	'0D6'	'098'	'\0D7'	'0D7'	'099'
'\0D8'	'0D8'	'09A'	'\0D9'	'0D9'	'09B'
'\0DA'	'0DA'	'09C'	'\0DB'	'0DB'	'09D'
'\0DC'	'0DC'	'09E'	'\0DD'	'0DD'	'09F'
'\0DE'	'0DE'	'0A0'	'\0EF'	'0EF'	'0A1'
'\0F0'	'0F0'	'0A2'	'\0F1'	'0F1'	'0A3'
'\0F2'	'0F2'	'0A4'	'\0F3'	'0F3'	'0A5'
'\0F4'	'0F4'	'0A6'	'\0F5'	'0F5'	'0A7'
'\0F6'	'0F6'	'0A8'	'\0F7'	'0F7'	'0A9'
'\0F8'	'0F8'	'0AA'	'\0F9'	'0F9'	'0AB'
'\0FA'	'0FA'	'0AC'	'\0FB'	'0FB'	'0AD'
'\0FC'	'0FC'	'0AE'	'\0FD'	'0FD'	'0BD'
'\0FE'	'0FE'	'0C0'	'\0FF'	'0FF'	'0C1'

- `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'.
- What is `chr(33)`?

## In Pairs or Triples...

```
1  for c in range(65,90):
2      print(chr(c))
3
4  message = "I love Python"
5  newMessage = ""
6  for c in message:
7      print(ord(c))    #Print the Unicode of each
                       number
8      print(chr(ord(c)+1))  #Print the next
                           character
9  newMessage = newMessage + chr(ord(c)+1) #
                           add to the new message
10 print("The coded message is", newMessage)
```

## Ceasar Ciper: hints for P9 of programming assignments

```
1 word = input("Enter a string: ")
2 codedWord = ""
3 shift = 2 #shift two letters
4 for ch in word:
5     offset = ord(ch) - ord('A') #relative
          distance to 'A'
6     wrap = (offset + shift) % ? #what is ?
7     #TODO: compute the new letter
8     #TODO: add the newChar to the coded word
9
10 print("After shifting", shift, "letters,", \
11      word, "becomes", codedWord)
```

# User Input

*Covered in detail in Lab 2:*

---

```
→ 1 mess = input('Please enter a message: ')
  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 "A" to the end of the strings `s`.

# Today's Topics



- For-loops
- `range()`
- Variables
- Characters
- **Strings**

## More on Strings: String Methods

```
s = "FridaysSaturdaysSundays"  
num = s.count("s")
```

- The first line creates a variable, called `s`, that stores the string:  
"FridaysSaturdaysSundays"

## More on Strings: String Methods

```
s = "FridaysSaturdaysSundays"  
num = s.count("s")
```

- The first line creates a variable, called `s`, that stores the string:  
"FridaysSaturdaysSundays"
- There are many useful functions for strings (more in Lab 2).

## More on Strings: String Methods

```
s = "FridaysSaturdaysSundays"  
num = s.count("s")
```

- The first line creates a variable, called `s`, that stores the string:  
"FridaysSaturdaysSundays"
- There are many useful functions for strings (more in Lab 2).
- `s.count(x)` will count the number of times the pattern, `x`, appears in `s`.

## More on Strings: String Methods

```
s = "FridaysSaturdaysSundays"  
num = s.count("s")
```

- The first line creates a variable, called `s`, that stores the string:  
"FridaysSaturdaysSundays"
- There are many useful functions for strings (more in Lab 2).
- `s.count(x)` will count the number of times the pattern, `x`, appears in `s`.
  - ▶ `s.count("s")` counts the number of lower case `s` that occurs.

## More on Strings: String Methods

```
s = "FridaysSaturdaysSundays"  
num = s.count("s")
```

- The first line creates a variable, called `s`, that stores the string:  
"FridaysSaturdaysSundays"
- There are many useful functions for strings (more in Lab 2).
- `s.count(x)` will count the number of times the pattern, `x`, appears in `s`.
  - ▶ `s.count("s")` counts the number of lower case `s` that occurs.
  - ▶ `num = s.count("s")` stores the result in the variable `num`, for later.

# More on Strings: String Methods

```
s = "FridaysSaturdaysSundays"  
num = s.count("s")
```

- The first line creates a variable, called `s`, that stores the string: "FridaysSaturdaysSundays"
- There are many useful functions for strings (more in Lab 2).
- `s.count(x)` will count the number of times the pattern, `x`, appears in `s`.
  - ▶ `s.count("s")` counts the number of lower case `s` that occurs.
  - ▶ `num = s.count("s")` stores the result in the variable `num`, for later.
  - ▶ What would `print(s.count("sS"))` output?

# More on Strings: String Methods

```
s = "FridaysSaturdaysSundays"  
num = s.count("s")
```

- The first line creates a variable, called `s`, that stores the string:  
`"FridaysSaturdaysSundays"`
- There are many useful functions for strings (more in Lab 2).
- `s.count(x)` will count the number of times the pattern, `x`, appears in `s`.
  - ▶ `s.count("s")` counts the number of lower case `s` that occurs.
  - ▶ `num = s.count("s")` stores the result in the variable `num`, for later.
  - ▶ What would `print(s.count("sS"))` output?
  - ▶ What about:  
`mess = "10 20 21 9 101 35"  
mults = mess.count("0 ")  
print(mults)`

## More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"  
days = s[7]  
days = s[7:15]  
days = s[:-1]
```

- Strings are made up of individual characters (letters, numbers, etc.)

## More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"  
days = s[7]  
days = s[7:15]  
days = s[:-1]
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

## More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"  
days = s[7]  
days = s[7:15]  
days = s[:-1]
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"  
days = s[7]  
days = s[7:15]  
days = s[:-1]
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- `s[0]` is

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- `s[0]` is 'F'.

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- `s[1]` is

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- `s[1]` is 'r'.

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- $s[-1]$  is

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- $s[-1]$  is ‘s’.

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- `s[3:6]` is

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- `s[3:6]` is 'day'.

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- `s[:3]` is

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- `s[:3]` is 'Fri'.

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- `s[:-1]` is

# More on Strings: Indexing & Substrings

```
s = "FridaysSaturdaysSundays"
```

- Strings are made up of individual characters (letters, numbers, etc.)
- Useful to be able to refer to pieces of a string, either an individual location or a “substring” of the string.

0	1	2	3	4	5	6	7	8	...	16	17	18	19	20	21	22	
F	r	i	d	a	y	s	S	a	...	S	u	n	d	a	y	s	
													...	-4	-3	-2	-1

- `s[:-1]` is 'FridaysSaturdaysSunday'.  
*(no trailing 's' at the end)*

# Today's Topics



- For-loops
- range()
- Variables
- Characters
- Strings

# Recap

- In Python, we introduced:

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

# Recap

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

- In Python, we introduced:

- ▶ For-loops

# Recap

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

- In Python, we introduced:

- ▶ For-loops
- ▶ range()

# Recap

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

- In Python, we introduced:

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

# Recap

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

- In Python, we introduced:

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

# Recap

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

- In Python, we introduced:

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

# Recap

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

- In Python, we introduced:

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

# Recap

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

- In Python, we introduced:

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

# Recap

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

- In Python, we introduced:

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

# Practice Quiz & Final Questions



- Since you must pass the final exam to pass the course, we end every lecture with final exam review.

# Practice Quiz & Final Questions



- Since you must pass the final exam to pass the course, we end every lecture with final exam review.
- Pull out something to write on (not to be turned in).
- Lightning rounds:
  - ▶ write as much you can for 60 seconds;
  - ▶ followed by answer; and
  - ▶ repeat.
- Past exams are on the webpage ([under Final Exam Information](#)).
- We're starting with Spring 2018, Mock Exam.

# Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab

# Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001G Hunter North

# Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001G Hunter North
- If you haven't already, schedule an appointment to take the Code Review (**one every week**) in lab 1001G Hunter North

# Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001G Hunter North
- If you haven't already, schedule an appointment to take the Code Review (**one every week**) in lab 1001G Hunter North
- Submit this week's 5 programming assignments (**programs 6-10**)

# Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001G Hunter North
- If you haven't already, schedule an appointment to take the Code Review (**one every week**) in lab 1001G Hunter North
- Submit this week's 5 programming assignments (**programs 6-10**)
- If you need help, schedule an appointment for Tutoring in lab 1001G 11:30am-5:30pm

# Weekly Reminders!



Before next lecture, don't forget to:

- Work on this week's Online Lab
- Schedule an appointment to take the Quiz in lab 1001G Hunter North
- If you haven't already, schedule an appointment to take the Code Review (**one every week**) in lab 1001G Hunter North
- Submit this week's 5 programming assignments (**programs 6-10**)
- If you need help, schedule an appointment for Tutoring in lab 1001G 11:30am-5:30pm
- Take the Lecture Preview on Blackboard on Monday (or no later than 10am on Tuesday)

# Lecture Slips & Writing Boards



- Hand your lecture slip to a UTA.
- Return writing boards as you leave.