

CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

Review of Lecture 1: range function

- Simplest version of range function is `range(n)`, where n is a positive integer. Function `range(n)` returns a list of integers ranging (see how the name is coming from?) from 0 to $n-1$, a total of n integer. For example, `range(5)` returns a list [0, 1, 2, 3, 4], while `range(-1)` returns an empty list.
- Usage: `range(n)` is part of for statement to repeat something for n times.
- Common mistake 1: use floating point number as parameters for range function. For example, `range(1.2)` is wrong.
- Common mistake 2: use square brackets [and] instead of parentheses (and) after range function. Note that range is a function name, it is followed by a pair of matched parentheses that enclose parameters for range function. For example, `range[2]` is wrong while `range(2)` is correct.

Review of Lecture 1: turtle graphics

- Imagine a turtle has a pen, when it moves some distance, say, 100 pixels, a distance unit in digital world, a line is drawn on the screen.
- Then the turtle turn left 120 degrees.
- Repeat the above two steps for 3 times.

```
1 import turtle  
2  
3 t = turtle.Turtle()  
4  
5 t.fd(100) # same as t.forward(100), ie, t moves forward 100 pixels  
6 t.left(120) # t turns left 90 degrees  
7  
8 t.fd(100)  
9 t.left(120)  
10  
11 t.fd(100)  
12 t.left(120)
```

Review of Lecture 1: use turtle to draw a triangle

- After turning 120 degrees for three times, the turtle turns around in a circle and comes back to its start point.
- Use for-statement and range function to rewrite the above program.

```
1 import turtle  
2  
3 t = turtle.Turtle()  
4  
5 for i in range(3):  
6     t.fd(100) #same as t.forward(100)  
7     t.left(120)
```

For more commands, read [turtle documentation](#)

Draw a polygon with $n \geq 3$ sides using turtle

Pseudocode (not actual code) work with any language with turtle library.
Also called algorithm, consisting of step to step instructions.

import turtle library

instantiate a turtle object called t

initialize n to be an integer at least 3

Repeat the following for n times

- (1) t moves forward t fixed number of distance
- (2) t turns left $360 / n$ degrees

Explanation: every time a turtle turns $360 / n$, after n rounds, move back to the start point.

Programming Environment

- Best choice: use command line and vi / Emac editors. Avoid using IDE (Integrated development environment).
- Second choice: Install Python and IDLE, an IDE for Python.
- Last choice (not encouraged): use online editor.
 - ▶ [onlinegdb](#) for non-turtle-graphics programs
 - ▶ [trinket](#) for turtle graphics programs.
- In windows, install WSL (Windows Subsystem for Linux) for command line testing.
- Lab 1001 G has Linux laptops, borrow one from TAs. Need department's permission to borrow overnight.
- You can borrow a Lenovo laptop free for a semester from the 2nd floor of library (at Audio/Video department, need an additional photo ID besides school's ID).

Today's Topics



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

Today's Topics



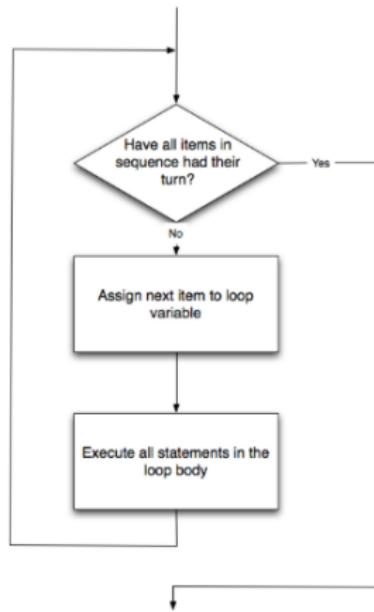
- **For-loops**
- `range()`
- Variables
- Characters
- Strings

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](#)

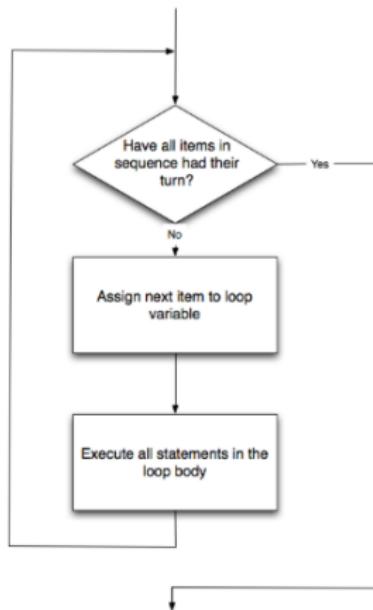
for-loop



```
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

Converting from Character to Code:

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

ASCII TABLE														
Decimal	Hex	Octal	Name	Char	Decimal	Hex	Octal	Name	Char	Decimal	Hex	Octal	Name	Char
0	00	0	NULL	\0	32	20	40	SIGKILL	\000	64	40	100	SIGPOLL	\001
1	01	1	SOH	\001	33	21	41	SIGALRM	\002	65	41	101	SIGSTOP	\003
2	02	2	STX	\002	34	22	42	SIGPOLL	\004	66	42	102	SIGCONT	\005
3	03	3	ETX	\003	35	23	43	SIGPOLL	\006	67	43	103	SIGKILL	\007
4	04	4	EOT	\004	36	24	44	SIGPOLL	\008	68	44	104	SIGPOLL	\009
5	05	5	ENQ	\005	37	25	45	SIGPOLL	\010	69	45	105	SIGPOLL	\011
6	06	6	ACK	\006	38	26	46	SIGPOLL	\012	70	46	106	SIGPOLL	\013
7	07	7	NAK	\007	39	27	47	SIGPOLL	\014	71	47	107	SIGPOLL	\015
8	08	10	SYN	\008	40	28	48	SIGPOLL	\016	72	48	108	SIGPOLL	\017
9	09	11	DC1	\009	41	29	49	SIGPOLL	\018	73	49	109	SIGPOLL	\019
10	0A	12	DC2	\00A	42	2A	4A	SIGPOLL	\01A	74	4A	110	SIGPOLL	\01B
11	0B	13	DC3	\00B	43	2B	4B	SIGPOLL	\01B	75	4B	111	SIGPOLL	\01C
12	0C	14	DC4	\00C	44	2C	4C	SIGPOLL	\01D	76	4C	112	SIGPOLL	\01E
13	0D	15	SUSP	\00D	45	2D	4D	SIGPOLL	\01F	77	4D	113	SIGPOLL	\01F
14	0E	16	EM	\00E	46	2E	4E	SIGPOLL	\020	78	4E	114	SIGPOLL	\021
15	0F	17	FS	\00F	47	2F	4F	SIGPOLL	\022	79	4F	115	SIGPOLL	\023
16	10	20	GS	\010	48	30	50	SIGPOLL	\024	80	50	116	SIGPOLL	\025
17	11	21	RS	\011	49	31	51	SIGPOLL	\026	81	51	117	SIGPOLL	\027
18	12	22	US	\012	4A	32	52	SIGPOLL	\028	82	52	118	SIGPOLL	\029
19	13	23	DEL	\013	4B	33	53	SIGPOLL	\02A	83	53	119	SIGPOLL	\02B
20	14	24			4C	34	54	SIGPOLL	\02C	84	54	120	SIGPOLL	\02D
21	15	25			4D	35	55	SIGPOLL	\02E	85	55	121	SIGPOLL	\02F
22	16	26			4E	36	56	SIGPOLL	\030	86	56	122	SIGPOLL	\031
23	17	27			4F	37	57	SIGPOLL	\032	87	57	123	SIGPOLL	\033
24	18	28			50	38	58	SIGPOLL	\034	88	58	124	SIGPOLL	\035
25	19	29			51	39	59	SIGPOLL	\036	89	59	125	SIGPOLL	\037
26	1A	2A			52	3A	5A	SIGPOLL	\038	90	5A	126	SIGPOLL	\039
27	1B	2B			53	3B	5B	SIGPOLL	\03A	91	5B	127	SIGPOLL	\03B
28	1C	2C			54	3C	5C	SIGPOLL	\03C	92	5C	128	SIGPOLL	\03D
29	1D	2D			55	3D	5D	SIGPOLL	\03E	93	5D	129	SIGPOLL	\03F
30	1E	2E			56	3E	5E	SIGPOLL	\040	94	5E	130	SIGPOLL	\041
31	1F	2F			57	3F	5F	SIGPOLL	\042	95	5F	131	SIGPOLL	\043
32	20	30			58	40	60	SIGPOLL	\044	96	60	132	SIGPOLL	\045
33	21	31			59	41	61	SIGPOLL	\046	97	61	133	SIGPOLL	\047
34	22	32			5A	42	62	SIGPOLL	\048	98	62	134	SIGPOLL	\049
35	23	33			5B	43	63	SIGPOLL	\04A	99	63	135	SIGPOLL	\04B
36	24	34			5C	44	64	SIGPOLL	\04C	100	64	136	SIGPOLL	\04D
37	25	35			5D	45	65	SIGPOLL	\04E	101	65	137	SIGPOLL	\04F
38	26	36			5E	46	66	SIGPOLL	\050	102	66	138	SIGPOLL	\051
39	27	37			5F	47	67	SIGPOLL	\052	103	67	139	SIGPOLL	\053
40	28	38			60	48	68	SIGPOLL	\054	104	68	140	SIGPOLL	\055
41	29	39			61	49	69	SIGPOLL	\056	105	69	141	SIGPOLL	\057
42	2A	3A			62	4A	6A	SIGPOLL	\058	106	6A	142	SIGPOLL	\059
43	2B	3B			63	4B	6B	SIGPOLL	\05A	107	6B	143	SIGPOLL	\05B
44	2C	3C			64	4C	6C	SIGPOLL	\05C	108	6C	144	SIGPOLL	\05D
45	2D	3D			65	4D	6D	SIGPOLL	\05E	109	6D	145	SIGPOLL	\05F
46	2E	3E			66	4E	6E	SIGPOLL	\060	110	6E	146	SIGPOLL	\061
47	2F	3F			67	4F	6F	SIGPOLL	\062	111	6F	147	SIGPOLL	\063
48	30	40			68	50	70	SIGPOLL	\064	112	70	148	SIGPOLL	\065
49	31	41			69	51	71	SIGPOLL	\066	113	71	149	SIGPOLL	\067
50	32	42			6A	52	72	SIGPOLL	\068	114	72	150	SIGPOLL	\069
51	33	43			6B	53	73	SIGPOLL	\06A	115	73	151	SIGPOLL	\06B
52	34	44			6C	54	74	SIGPOLL	\06C	116	74	152	SIGPOLL	\06D
53	35	45			6D	55	75	SIGPOLL	\06E	117	75	153	SIGPOLL	\06F
54	36	46			6E	56	76	SIGPOLL	\070	118	76	154	SIGPOLL	\071
55	37	47			6F	57	77	SIGPOLL	\072	119	77	155	SIGPOLL	\073
56	38	48			70	58	78	SIGPOLL	\074	120	78	156	SIGPOLL	\075
57	39	49			71	59	79	SIGPOLL	\076	121	79	157	SIGPOLL	\077
58	3A	4A			72	5A	7A	SIGPOLL	\078	122	7A	158	SIGPOLL	\079
59	3B	4B			73	5B	7B	SIGPOLL	\07A	123	7B	159	SIGPOLL	\07B
60	3C	4C			74	5C	7C	SIGPOLL	\07C	124	7C	160	SIGPOLL	\07D
61	3D	4D			75	5D	7D	SIGPOLL	\07E	125	7D	161	SIGPOLL	\07F
62	3E	4E			76	5E	7E	SIGPOLL	\080	126	7E	162	SIGPOLL	\081
63	3F	4F			77	5F	7F	SIGPOLL	\082	127	7F	163	SIGPOLL	\083
64	40	50			78	60	80	SIGPOLL	\084	128	80	164	SIGPOLL	\085
65	41	51			79	61	81	SIGPOLL	\086	129	81	165	SIGPOLL	\087
66	42	52			7A	62	82	SIGPOLL	\088	130	82	166	SIGPOLL	\089
67	43	53			7B	63	83	SIGPOLL	\08A	131	83	167	SIGPOLL	\08B
68	44	54			7C	64	84	SIGPOLL	\08C	132	84	168	SIGPOLL	\08D
69	45	55			7D	65	85	SIGPOLL	\08E	133	85	169	SIGPOLL	\08F
70	46	56			7E	66	86	SIGPOLL	\090	134	86	170	SIGPOLL	\091
71	47	57			7F	67	87	SIGPOLL	\092	135	87	171	SIGPOLL	\093
72	48	58			80	68	88	SIGPOLL	\094	136	88	172	SIGPOLL	\095
73	49	59			81	69	89	SIGPOLL	\096	137	89	173	SIGPOLL	\097
74	4A	5A			82	6A	8A	SIGPOLL	\098	138	8A	174	SIGPOLL	\099
75	4B	5B			83	6B	8B	SIGPOLL	\09A	139	8B	175	SIGPOLL	\09B
76	4C	5C			84	6C	8C	SIGPOLL	\09C	140	8C	176	SIGPOLL	\09D
77	4D	5D			85	6D	8D	SIGPOLL	\09E	141	8D	177	SIGPOLL	\09F
78	4E	5E			86	6E	8E	SIGPOLL	\0A0	142	8E	178	SIGPOLL	\0A1
79	4F	5F			87	6F	8F	SIGPOLL	\0A2	143	8F	179	SIGPOLL	\0A3
80	50	60			88	70	90	SIGPOLL	\0A4	144	90	180	SIGPOLL	\0A5
81	51	61			89	71	91	SIGPOLL	\0A6	145	91	181	SIGPOLL	\0A7
82	52	62			8A	72	92	SIGPOLL	\0A8	146	92	182	SIGPOLL	\0A9
83	53	63			8B	73	93	SIGPOLL	\0AA	147	93	183	SIGPOLL	\0AB
84	54	64			8C	74	94	SIGPOLL	\0AC	148	94	184	SIGPOLL	\0AD
85	55	65			8D	75	95	SIGPOLL	\0AE	149	95	185	SIGPOLL	\0AF
86	56	66			8E	76	96	SIGPOLL	\0B0	150	96	186	SIGPOLL	\0B1
87	57	67			8F	77	97	SIGPOLL	\0B2	151	97	187	SIGPOLL	\0B3
88	58	68			90	78	98	SIGPOLL	\0B4	152	98	188	SIGPOLL	\0B5
89	59	69			91	79	99	SIGPOLL	\0B6	153	99	189	SIGPOLL	\0B7
90	5A	6A			92	7A	9A	SIGPOLL	\0B8	154	9A	190	SIGPOLL	\0B9
91	5B	6B			93	7B	9B	SIGPOLL	\0BA	155	9B	191	SIGPOLL	\0BB
92	5C	6C			94	7C	9C	SIGPOLL	\0BC	156	9C	192	SIGPOLL	\0BD
93	5D	6D			95	7D	9D	SIGPOLL	\0BE	157	9D	193	SIGPOLL	\0BF
94	5E	6E			96	7E	9E	SIGPOLL	\0C0	158	9E	194	SIGPOLL	\0C1
95	5F	6F			97	7F	9F	SIGPOLL	\0C2	159	9F	195	SIGPOLL	\0C3
96	60	70			98	80	A0	SIGPOLL	\0C4	160	A0	196	SIGPOLL	\0C5
97	61	71			99	81	A1	SIGPOLL	\0C6	161	A1	197	SIGPOLL	\0C7
98	62	72			9A	82	A2	SIGPOLL	\0C8	162	A2	198	SIGPOLL	\0C9
99	63	73			9B	83	A3	SIGPOLL	\0CA	163	A3	199	SIGPOLL	\0CB
100	64	74			9C	84	A4	SIGPOLL	\0CC	164	A4	200	SIGPOLL	\0CD
101	65	75			9D	85	A5	SIGPOLL	\0CE	165	A5	201	SIGPOLL	\0CF
102	66	76			9E	86	A6	SIGPOLL	\0D0	166	A6	202	SIGPOLL	\0D1
103	67	77			9F	87	A7	SIGPOLL	\0D2	167	A7	203	SIGPOLL	\0D3
104	68	78			90	88	A8	SIGPOLL	\0D4	168	A8	204	SIGPOLL	\0D5
105	69	79			91	89	A9	SIGPOLL	\0D6	169	A9	205		

- `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		128	80		255	F0	
	1	01		129	81		256	F1	
	2	02		130	82		257	F2	
	3	03		131	83		258	F3	
	4	04		132	84		259	F4	
	5	05		133	85		260	F5	
	6	06		134	86		261	F6	
	7	07		135	87		262	F7	
	8	08		136	88		263	F8	
	9	09		137	89		264	F9	
	A	0A		138	8A		265	FA	
	B	0B		139	8B		266	FB	
	C	0C		140	8C		267	FC	
	D	0D		141	8D		268	FD	
	E	0E		142	8E		269	FE	
	F	0F		143	8F		270	FF	
	G	10		144	90		271	00	
	H	11		145	91		272	01	
	I	12		146	92		273	02	
	J	13		147	93		274	03	
	K	14		148	94		275	04	
	L	15		149	95		276	05	
	M	16		150	96		277	06	
	N	17		151	97		278	07	
	O	18		152	98		279	08	
	P	19		153	99		280	09	
	Q	1A		154	9A		281	0A	
	R	1B		155	9B		282	0B	
	S	1C		156	9C		283	0C	
	T	1D		157	9D		284	0D	
	U	1E		158	9E		285	0E	
	V	1F		159	9F		286	0F	
	W	20		160	A0		287	10	
	X	21		161	A1		288	11	
	Y	22		162	A2		289	12	
	Z	23		163	A3		290	13	
		24		164	A4		291	14	
		25		165	A5		292	15	
		26		166	A6		293	16	
		27		167	A7		294	17	
		28		168	A8		295	18	
		29		169	A9		296	19	
		2A		170	AA		297	1A	
		2B		171	AB		298	1B	
		2C		172	AC		299	1C	
		2D		173	AD		300	1D	
		2E		174	AE		301	1E	
		2F		175	AF		302	1F	
		30		176	BF		303	20	
		31		177	CF		304	21	
		32		178	DF		305	22	
		33		179	EF		306	23	
		34		180	FF		307	24	
		35		181	00		308	25	
		36		182	01		309	26	
		37		183	02		310	27	
		38		184	03		311	28	
		39		185	04		312	29	
		3A		186	05		313	2A	
		3B		187	06		314	2B	
		3C		188	07		315	2C	
		3D		189	08		316	2D	
		3E		190	09		317	2E	
		3F		191	0A		318	2F	
		40		192	0B		319	30	
		41		193	0C		320	31	
		42		194	0D		321	32	
		43		195	0E		322	33	
		44		196	0F		323	34	
		45		197	10		324	35	
		46		198	11		325	36	
		47		199	12		326	37	
		48		200	13		327	38	
		49		201	14		328	39	
		4A		202	15		329	3A	
		4B		203	16		330	3B	
		4C		204	17		331	3C	
		4D		205	18		332	3D	
		4E		206	19		333	3E	
		4F		207	1A		334	3F	
		50		208	1B		335	40	
		51		209	1C		336	41	
		52		210	1D		337	42	
		53		211	1E		338	43	
		54		212	1F		339	44	
		55		213	20		340	45	
		56		214	21		341	46	
		57		215	22		342	47	
		58		216	23		343	48	
		59		217	24		344	49	
		5A		218	25		345	4A	
		5B		219	26		346	4B	
		5C		220	27		347	4C	
		5D		221	28		348	4D	
		5E		222	29		349	4E	
		5F		223	2A		350	4F	
		60		224	2B		351	50	
		61		225	2C		352	51	
		62		226	2D		353	52	
		63		227	2E		354	53	
		64		228	2F		355	54	
		65		229	30		356	55	
		66		230	31		357	56	
		67		231	32		358	57	
		68		232	33		359	58	
		69		233	34		360	59	
		6A		234	35		361	5A	
		6B		235	36		362	5B	
		6C		236	37		363	5C	
		6D		237	38		364	5D	
		6E		238	39		365	5E	
		6F		239	3A		366	5F	
		70		240	3B		367	60	
		71		241	3C		368	61	
		72		242	3D		369	62	
		73		243	3E		370	63	
		74		244	3F		371	64	
		75		245	40		372	65	
		76		246	41		373	66	
		77		247	42		374	67	
		78		248	43		375	68	
		79		249	44		376	69	
		7A		250	45		377	6A	
		7B		251	46		378	6B	
		7C		252	47		379	6C	
		7D		253	48		380	6D	
		7E		254	49		381	6E	
		7F		255	4A		382	6F	
		80		256	4B		383	70	
		81		257	4C		384	71	
		82		258	4D		385	72	
		83		259	4E		386	73	
		84		260	4F		387	74	
		85		261	50		388	75	
		86		262	51		389	76	
		87		263	52		390	77	
		88		264	53		391	78	
		89		265	54		392	79	
		8A		266	55		393	7A	
		8B		267	56		394	7B	
		8C		268	57		395	7C	
		8D		269	58		396	7D	
		8E		270	59		397	7E	
		8F		271	5A		398	7F	
		90		272	5B		399	80	
		91		273	5C		400	81	
		92		274	5D		401	82	
		93		275	5E		402	83	
		94		276	5F		403	84	
		95		277	60		404	85	
		96		278	61		405	86	
		97		279	62		406	87	
		98		280	63		407	88	
		99		281	64		408	89	
		9A		282	65		409	8A	
		9B		283	66		410	8B	
		9C		284	67		411	8C	
		9D		285	68		412	8D	
		9E		286	69		413	8E	
		9F		287	6A		414	8F	
		9A0		288	6B		415	90	
		9A1		289	6C		416	91	
		9A2		290	6D		417	92	
		9A3		291	6E		418	93	
		9A4		292	6F		419	94	
		9A5		293	70		420	95	
		9A6		294	71		421	96	
		9A7		295	72		422	97	
		9A8		296	73		423		

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	\c	67	43	C
4	04	\4	36	24	\d	68	44	D
5	05	\5	37	25	\e	69	45	E
6	06	\6	38	26	\f	70	46	F
7	07	\7	39	27	\g	71	47	G
8	08	\8	40	28	\h	72	48	H
9	09	\9	41	29	\i	73	49	I
10	0A	\10	42	2A	\j	74	4A	J
11	0B	\11	43	2B	\k	75	4B	K
12	0C	\12	44	2C	\l	76	4C	L
13	0D	\13	45	2D	\m	77	4D	M
14	0E	\14	46	2E	\n	78	4E	N
15	0F	\15	47	2F	\r	79	4F	R
16	10	\16	48	30	\n	80	50	\n
17	11	\17	49	31	\r	81	51	\r
18	12	\18	4A	32	\n	82	52	\n
19	13	\19	4B	33	\r	83	53	\r
20	14	\20	4C	34	\n	84	54	\n
21	15	\21	4D	35	\r	85	55	\r
22	16	\22	4E	36	\n	86	56	\n
23	17	\23	4F	37	\r	87	57	\r
24	18	\24	50	38	\n	88	58	\n
25	19	\25	51	39	\r	89	59	\r
26	1A	\26	52	3A	\n	90	5A	\n
27	1B	\27	53	3B	\r	91	5B	\r
28	1C	\28	54	3C	\n	92	5C	\n
29	1D	\29	55	3D	\r	93	5D	\r
30	1E	\20	56	3E	\n	94	5E	\n
31	1F	\21	57	3F	\r	95	5F	\r
32	20	\22	58	40	\n	96	60	\n
33	21	\23	59	41	\r	97	61	\r
34	22	\24	5A	42	\n	98	62	\n
35	23	\25	5B	43	\r	99	63	\r
36	24	\26	5C	44	\n	100	64	\n
37	25	\27	5D	45	\r	101	65	\r
38	26	\28	5E	46	\n	102	66	\n
39	27	\29	5F	47	\r	103	67	\r
40	28	\20	60	48	\n	104	68	\n
41	29	\21	61	49	\r	105	69	\r
42	2A	\22	62	4A	\n	106	6A	\n
43	2B	\23	63	4B	\r	107	6B	\r
44	2C	\24	64	4C	\n	108	6C	\n
45	2D	\25	65	4D	\r	109	6D	\r
46	2E	\26	66	4E	\n	110	6E	\n
47	2F	\27	67	4F	\r	111	6F	\r
48	30	\28	68	50	\n	112	70	\n
49	31	\29	69	51	\r	113	71	\r
50	32	\20	6A	52	\n	114	72	\n
51	33	\21	6B	53	\r	115	73	\r
52	34	\22	6C	54	\n	116	74	\n
53	35	\23	6D	55	\r	117	75	\r
54	36	\24	6E	56	\n	118	76	\n
55	37	\25	6F	57	\r	119	77	\r
56	38	\26	70	58	\n	120	78	\n
57	39	\27	71	59	\r	121	79	\r
58	3A	\28	72	5A	\n	122	7A	\n
59	3B	\29	73	5B	\r	123	7B	\r
60	3C	\20	74	5C	\n	124	7C	\n
61	3D	\21	75	5D	\r	125	7D	\r
62	3E	\22	76	5E	\n	126	7E	\n
63	3F	\23	77	5F	\r	127	7F	\r
64	40	\24	78	60	\n	128	80	\n
65	41	\25	79	61	\r	129	81	\r
66	42	\26	7A	62	\n	130	82	\n
67	43	\27	7B	63	\r	131	83	\r
68	44	\28	7C	64	\n	132	84	\n
69	45	\29	7D	65	\r	133	85	\r
70	46	\20	7E	66	\n	134	86	\n
71	47	\21	7F	67	\r	135	87	\r
72	48	\22	80	68	\n	136	88	\n
73	49	\23	81	69	\r	137	89	\r
74	4A	\24	82	6A	\n	138	8A	\n
75	4B	\25	83	6B	\r	139	8B	\r
76	4C	\26	84	6C	\n	140	8C	\n
77	4D	\27	85	6D	\r	141	8D	\r
78	4E	\28	86	6E	\n	142	8E	\n
79	4F	\29	87	6F	\r	143	8F	\r
80	50	\20	88	70	\n	144	90	\n
81	51	\21	89	71	\r	145	91	\r
82	52	\22	8A	72	\n	146	92	\n
83	53	\23	8B	73	\r	147	93	\r
84	54	\24	8C	74	\n	148	94	\n
85	55	\25	8D	75	\r	149	95	\r
86	56	\26	8E	76	\n	150	96	\n
87	57	\27	8F	77	\r	151	97	\r
88	58	\28	90	78	\n	152	98	\n
89	59	\29	91	79	\r	153	99	\r
90	5A	\20	92	7A	\n	154	9A	\n
91	5B	\21	93	7B	\r	155	9B	\r
92	5C	\22	94	7C	\n	156	9C	\n
93	5D	\23	95	7D	\r	157	9D	\r
94	5E	\24	96	7E	\n	158	9E	\n
95	5F	\25	97	7F	\r	159	9F	\r
96	60	\26	98	80	\n	160	100	\n
97	61	\27	99	81	\r	161	101	\r
98	62	\28	9A	82	\n	162	102	\n
99	63	\29	9B	83	\r	163	103	\r
100	64	\20	9C	84	\n	164	104	\n
101	65	\21	9D	85	\r	165	105	\r
102	66	\22	9E	86	\n	166	106	\n
103	67	\23	9F	87	\r	167	107	\r
104	68	\24	90	88	\n	168	108	\n
105	69	\25	91	89	\r	169	109	\r
106	6A	\26	92	8A	\n	170	10A	\n
107	6B	\27	93	8B	\r	171	10B	\r
108	6C	\28	94	8C	\n	172	10C	\n
109	6D	\29	95	8D	\r	173	10D	\r
110	6E	\20	96	8E	\n	174	10E	\n
111	6F	\21	97	8F	\r	175	10F	\r
112	70	\22	98	90	\n	176	110	\n
113	71	\23	99	91	\r	177	111	\r
114	72	\24	9A	92	\n	178	112	\n
115	73	\25	9B	93	\r	179	113	\r
116	74	\26	9C	94	\n	180	114	\n
117	75	\27	9D	95	\r	181	115	\r
118	76	\28	9E	96	\n	182	116	\n
119	77	\29	9F	97	\r	183	117	\r
120	78	\20	90	98	\n	184	118	\n
121	79	\21	91	99	\r	185	119	\r
122	7A	\22	92	9A	\n	186	11A	\n
123	7B	\23	93	9B	\r	187	11B	\r
124	7C	\24	94	9C	\n	188	11C	\n
125	7D	\25	95	9D	\r	189	11D	\r
126	7E	\26	96	9E	\n	190	11E	\n
127	7F	\27	97	9F	\r	191	11F	\r
128	80	\28	98	90	\n	192	120	\n
129	81	\29	99	91	\r	193	121	\r
130	82	\20	9A	92	\n	194	122	\n
131	83	\21	9B	93	\r	195	123	\r
132	84	\22	9C	94	\n	196	124	\n
133	85	\23	9D	95	\r	197	125	\r
134	86	\24	9E	96	\n	198	126	\n
135	87	\25	9F	97	\r	199	127	\r
136	88	\26	90	98	\n	200	128	\n
137	89	\27	91	99	\r	201	129	\r
138	8A	\28	92	9A	\n	202	12A	\n
139	8B	\29	93	9B	\r	203	12B	\r
140	8C	\20	94	9C	\n	204	12C	\n
141	8D	\21	95	9D	\r	205	12D	\r
142	8E	\22	96	9E	\n	206	12E	\n
143	8F	\23	97	9F	\r	207	12F	\r
144	90	\24	98	90	\n	208	130	\n
145	91	\25	99	91	\r	209	131	\r
146	92	\26	9A	92	\n	210	132	\n
147	93	\27	9B	93	\r	211	133	\r
148	94	\28	9C	94	\n	212	134	\n
149	95	\29	9D	95	\r	213	135	\r
150	96	\20	9E	96	\n	214	136	\n
151	97	\21	9F	97	\r	215	137	\r
152	98	\22	90	98	\n	216	138	\n
153	99	\23	91	99	\r	217	139	\r
154	9A	\24	92	9A	\n	218	13A	\n
155	9B	\25	93	9B	\r	219	13B	\r
156	9C	\26	94	9C	\n	220	13C	\n
157	9D	\27	95	9D	\r	221	13D	\r
158	9E	\28	96	9E	\n	222	13E	\n
159	9F	\29	97	9F	\r	223	13F	\r
160	90	\20	98	90	\n	224	140	\n
161	91	\21	99	91	\r	225	141	\r
162	92	\22	9A	92	\n	226	142	\n
163	93	\23	9B	93	\r	227	143	\r
164	94	\24	9C	94	\n	228	144	\n
165	95	\25	9D	95	\r	229	145	\r
166	96	\26	9E	96	\n	230	146	\n
167	97	\27	9F	97	\r	231	147	\r
168	98	\28	90	98	\n	232	148	\n
169	99	\29	91	99	\r	233	149	\r
170	9A	\20	92	9A	\n	234	14A	\n
171	9B	\21	93	9B	\r	235	14B	\r
172	9C	\22	94	9C	\n	236	14C	\n
173	9D	\23	95	9D	\r	237	14D	\r
174	9E	\24	96	9E	\n	238	14E	\n
175	9F	\25	97	9F	\r	239	14F	\r
176	90	\26	98	90	\n	240	150	\n
177	91	\27	99	91	\r	241	151	\r
178	92	\28	9A	92	\n	242	152	\n
179	93	\29	9B	93	\r			

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	Hex	Char	Octal	Hex	Char
0	00	\0	000	00	\0	000	00	\0
1	01	\1	001	01	\1	001	01	\1
2	02	\2	002	02	\2	002	02	\2
3	03	\3	003	03	\3	003	03	\3
4	04	\4	004	04	\4	004	04	\4
5	05	\5	005	05	\5	005	05	\5
6	06	\6	006	06	\6	006	06	\6
7	07	\7	007	07	\7	007	07	\7
8	08	\8	010	08	\8	010	08	\8
9	09	\9	011	09	\9	011	09	\9
10	0A	\n	012	0A	\n	012	0A	\n
11	0B	\v	013	0B	\v	013	0B	\v
12	0C	\f	014	0C	\f	014	0C	\f
13	0D	\r	015	0D	\r	015	0D	\r
14	0E	\t	016	0E	\t	016	0E	\t
15	0F	\b	017	0F	\b	017	0F	\b
16	10	\012	020	10	\012	020	10	\012
17	11	\013	021	11	\013	021	11	\013
18	12	\014	022	12	\014	022	12	\014
19	13	\015	023	13	\015	023	13	\015
20	14	\016	024	14	\016	024	14	\016
21	15	\017	025	15	\017	025	15	\017
22	16	\020	026	16	\020	026	16	\020
23	17	\021	027	17	\021	027	17	\021
24	18	\022	028	18	\022	028	18	\022
25	19	\023	029	19	\023	029	19	\023
26	1A	\024	02A	1A	\024	02A	1A	\024
27	1B	\025	02B	1B	\025	02B	1B	\025
28	1C	\026	02C	1C	\026	02C	1C	\026
29	1D	\027	02D	1D	\027	02D	1D	\027
30	1E	\030	02E	1E	\030	02E	1E	\030
31	1F	\031	02F	1F	\031	02F	1F	\031
32	20	\040	030	20	\040	030	20	\040
33	21	\041	031	21	\041	031	21	\041
34	22	\042	032	22	\042	032	22	\042
35	23	\043	033	23	\043	033	23	\043
36	24	\044	034	24	\044	034	24	\044
37	25	\045	035	25	\045	035	25	\045
38	26	\046	036	26	\046	036	26	\046
39	27	\047	037	27	\047	037	27	\047
40	28	\050	038	28	\050	038	28	\050
41	29	\051	039	29	\051	039	29	\051
42	2A	\052	03A	2A	\052	03A	2A	\052
43	2B	\053	03B	2B	\053	03B	2B	\053
44	2C	\054	03C	2C	\054	03C	2C	\054
45	2D	\055	03D	2D	\055	03D	2D	\055
46	2E	\056	03E	2E	\056	03E	2E	\056
47	2F	\057	03F	2F	\057	03F	2F	\057
48	30	\060	040	30	\060	040	30	\060
49	31	\061	041	31	\061	041	31	\061
50	32	\062	042	32	\062	042	32	\062
51	33	\063	043	33	\063	043	33	\063
52	34	\064	044	34	\064	044	34	\064
53	35	\065	045	35	\065	045	35	\065
54	36	\066	046	36	\066	046	36	\066
55	37	\067	047	37	\067	047	37	\067
56	38	\070	048	38	\070	048	38	\070
57	39	\071	049	39	\071	049	39	\071
58	3A	\072	04A	3A	\072	04A	3A	\072
59	3B	\073	04B	3B	\073	04B	3B	\073
60	3C	\074	04C	3C	\074	04C	3C	\074
61	3D	\075	04D	3D	\075	04D	3D	\075
62	3E	\076	04E	3E	\076	04E	3E	\076
63	3F	\077	04F	3F	\077	04F	3F	\077
64	40	\080	050	40	\080	050	40	\080
65	41	\081	051	41	\081	051	41	\081
66	42	\082	052	42	\082	052	42	\082
67	43	\083	053	43	\083	053	43	\083
68	44	\084	054	44	\084	054	44	\084
69	45	\085	055	45	\085	055	45	\085
70	46	\086	056	46	\086	056	46	\086
71	47	\087	057	47	\087	057	47	\087
72	48	\090	058	48	\090	058	48	\090
73	49	\091	059	49	\091	059	49	\091
74	4A	\092	05A	4A	\092	05A	4A	\092
75	4B	\093	05B	4B	\093	05B	4B	\093
76	4C	\094	05C	4C	\094	05C	4C	\094
77	4D	\095	05D	4D	\095	05D	4D	\095
78	4E	\096	05E	4E	\096	05E	4E	\096
79	4F	\097	05F	4F	\097	05F	4F	\097
80	50	\0A0	060	50	\0A0	060	50	\0A0
81	51	\0A1	061	51	\0A1	061	51	\0A1
82	52	\0A2	062	52	\0A2	062	52	\0A2
83	53	\0A3	063	53	\0A3	063	53	\0A3
84	54	\0A4	064	54	\0A4	064	54	\0A4
85	55	\0A5	065	55	\0A5	065	55	\0A5
86	56	\0A6	066	56	\0A6	066	56	\0A6
87	57	\0A7	067	57	\0A7	067	57	\0A7
88	58	\0A8	068	58	\0A8	068	58	\0A8
89	59	\0A9	069	59	\0A9	069	59	\0A9
90	5A	\0A0	06A	5A	\0A0	06A	5A	\0A0
91	5B	\0A1	06B	5B	\0A1	06B	5B	\0A1
92	5C	\0A2	06C	5C	\0A2	06C	5C	\0A2
93	5D	\0A3	06D	5D	\0A3	06D	5D	\0A3
94	5E	\0A4	06E	5E	\0A4	06E	5E	\0A4
95	5F	\0A5	06F	5F	\0A5	06F	5F	\0A5
96	60	\0A6	070	60	\0A6	070	60	\0A6
97	61	\0A7	071	61	\0A7	071	61	\0A7
98	62	\0A8	072	62	\0A8	072	62	\0A8
99	63	\0A9	073	63	\0A9	073	63	\0A9
100	64	\0A0	074	64	\0A0	074	64	\0A0
101	65	\0A1	075	65	\0A1	075	65	\0A1
102	66	\0A2	076	66	\0A2	076	66	\0A2
103	67	\0A3	077	67	\0A3	077	67	\0A3
104	68	\0A4	078	68	\0A4	078	68	\0A4
105	69	\0A5	079	69	\0A5	079	69	\0A5
106	6A	\0A6	07A	6A	\0A6	07A	6A	\0A6
107	6B	\0A7	07B	6B	\0A7	07B	6B	\0A7
108	6C	\0A8	07C	6C	\0A8	07C	6C	\0A8
109	6D	\0A9	07D	6D	\0A9	07D	6D	\0A9
110	6E	\0A0	07E	6E	\0A0	07E	6E	\0A0
111	6F	\0A1	07F	6F	\0A1	07F	6F	\0A1
112	70	\0A2	080	70	\0A2	080	70	\0A2
113	71	\0A3	081	71	\0A3	081	71	\0A3
114	72	\0A4	082	72	\0A4	082	72	\0A4
115	73	\0A5	083	73	\0A5	083	73	\0A5
116	74	\0A6	084	74	\0A6	084	74	\0A6
117	75	\0A7	085	75	\0A7	085	75	\0A7
118	76	\0A8	086	76	\0A8	086	76	\0A8
119	77	\0A9	087	77	\0A9	087	77	\0A9
120	78	\0A0	088	78	\0A0	088	78	\0A0
121	79	\0A1	089	79	\0A1	089	79	\0A1
122	7A	\0A2	08A	7A	\0A2	08A	7A	\0A2
123	7B	\0A3	08B	7B	\0A3	08B	7B	\0A3
124	7C	\0A4	08C	7C	\0A4	08C	7C	\0A4
125	7D	\0A5	08D	7D	\0A5	08D	7D	\0A5
126	7E	\0A6	08E	7E	\0A6	08E	7E	\0A6
127	7F	\0A7	08F	7F	\0A7	08F	7F	\0A7
128	80	\0A8	090	80	\0A8	090	80	\0A8
129	81	\0A9	091	81	\0A9	091	81	\0A9
130	82	\0A0	092	82	\0A0	092	82	\0A0
131	83	\0A1	093	83	\0A1	093	83	\0A1
132	84	\0A2	094	84	\0A2	094	84	\0A2
133	85	\0A3	095	85	\0A3	095	85	\0A3
134	86	\0A4	096	86	\0A4	096	86	\0A4
135	87	\0A5	097	87	\0A5	097	87	\0A5
136	88	\0A6	098	88	\0A6	098	88	\0A6
137	89	\0A7	099	89	\0A7	099	89	\0A7
138	8A	\0A8	09A	8A	\0A8	09A	8A	\0A8
139	8B	\0A9	09B	8B	\0A9	09B	8B	\0A9
140	8C	\0A0	09C	8C	\0A0	09C	8C	\0A0
141	8D	\0A1	09D	8D	\0A1	09D	8D	\0A1
142	8E	\0A2	09E	8E	\0A2	09E	8E	\0A2
143	8F	\0A3	09F	8F	\0A3	09F	8F	\0A3
144	90	\0A4	0A0	90	\0A4	0A0	90	\0A4
145	91	\0A5	0A1	91	\0A5	0A1	91	\0A5
146	92	\0A6	0A2	92	\0A6	0A2	92	\0A6
147	93	\0A7	0A3	93	\0A7	0A3	93	\0A7
148	94	\0A8	0A4	94	\0A8	0A4	94	\0A8
149	95	\0A9	0A5	95	\0A9	0A5	95	\0A9
150	96	\0A0	0A6	96	\0A0	0A6	96	\0A0
151	97	\0A1	0A7	97	\0A1	0A7	97	\0A1
152	98	\0A2	0A8	98	\0A2	0A8	98	\0A2
153	99	\0A3	0A9	99	\0A3	0A9	99	\0A3
154	9A	\0A4	0AA	9A	\0A4	0AA	9A	\0A4
155	9B	\0A5	0AB	9B	\0A5	0AB	9B	\0A5
156	9C	\0A6	0AC	9C	\0A6	0AC	9C	\0A6
157	9D	\0A7	0AD	9D	\0A7	0AD	9D	\0A7
158	9E	\0A8	0AE	9E	\0A8	0AE	9E	\0A8
159	9F	\0A9	0AF	9F	\0A9	0AF	9F	\0A9
160	9A0	\0A0	0A0	9A0	\0A0	0A0	9A0	\0A0
161	9A1	\0A1	0A1	9A1	\0A1	0A1	9A1	\0A1
162	9A2	\0A2	0A2	9A2	\0A2	0A2	9A2	\0A2
163	9A3	\0A3	0A3	9A3	\0A3	0A3	9A3	\0A3
164	9A4	\0A4	0A4	9A4	\0A4	0A4	9A4	\0A4
165	9A5	\0A5	0A5	9A5	\0A5	0A5	9A5	\0A5
166	9A6	\0A6	0A6	9A6	\0A6	0A6	9A6	\0A6
16								

- `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:

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

ASCII TABLE														
Decimal	Hex	Octal	Name	Char	Decimal	Hex	Octal	Name	Char	Decimal	Hex	Octal	Name	Char
0	00	0	NULL	\0	32	20	40	SIGKILL	\000	64	40	100	SIGPOLL	\001
1	01	1	SOH	\001	33	21	41	SIGALRM	\002	65	41	101	SIGSTOP	\003
2	02	2	STX	\002	34	22	42	SIGPOLL	\004	66	42	102	SIGCONT	\005
3	03	3	ETX	\003	35	23	43	SIGPOLL	\006	67	43	103	SIGKILL	\007
4	04	4	ENQ	\004	36	24	44	SIGPOLL	\008	68	44	104	SIGPOLL	\009
5	05	5	KSYN	\005	37	25	45	SIGPOLL	\010	69	45	105	SIGPOLL	\011
6	06	6	ACK	\006	38	26	46	SIGPOLL	\012	70	46	106	SIGPOLL	\013
7	07	7	NAK	\007	39	27	47	SIGPOLL	\014	71	47	107	SIGPOLL	\015
8	08	10	SYN	\008	40	28	48	SIGPOLL	\016	72	48	108	SIGPOLL	\017
9	09	11	EOT	\009	41	29	49	SIGPOLL	\018	73	49	109	SIGPOLL	\019
10	0A	12	EM	\00A	42	2A	4A	SIGPOLL	\01A	74	4A	110	SIGPOLL	\01B
11	0B	13	END	\00B	43	2B	4B	SIGPOLL	\01B	75	4B	111	SIGPOLL	\01C
12	0C	14	ESC	\00C	44	2C	4C	SIGPOLL	\01C	76	4C	112	SIGPOLL	\01D
13	0D	15	SUSP	\00D	45	2D	4D	SIGPOLL	\01D	77	4D	113	SIGPOLL	\01E
14	0E	16	DC1	\00E	46	2E	4E	SIGPOLL	\01E	78	4E	114	SIGPOLL	\01F
15	0F	17	DC2	\00F	47	2F	4F	SIGPOLL	\01F	79	4F	115	SIGPOLL	\020
16	10	20	DC3	\010	48	30	50	SIGPOLL	\01F	80	50	116	SIGPOLL	\021
17	11	21	DC4	\011	49	31	51	SIGPOLL	\01F	81	51	117	SIGPOLL	\022
18	12	22	NAK	\012	50	32	52	SIGPOLL	\01F	82	52	118	SIGPOLL	\023
19	13	23	SYN	\013	51	33	53	SIGPOLL	\01F	83	53	119	SIGPOLL	\024
20	14	24	ENQ	\014	52	34	54	SIGPOLL	\01F	84	54	120	SIGPOLL	\025
21	15	25	ACK	\015	53	35	55	SIGPOLL	\01F	85	55	121	SIGPOLL	\026
22	16	26	DC1	\016	54	36	56	SIGPOLL	\01F	86	56	122	SIGPOLL	\027
23	17	27	DC2	\017	55	37	57	SIGPOLL	\01F	87	57	123	SIGPOLL	\028
24	18	28	DC3	\018	56	38	58	SIGPOLL	\01F	88	58	124	SIGPOLL	\029
25	19	29	DC4	\019	57	39	59	SIGPOLL	\01F	89	59	125	SIGPOLL	\02A
26	1A	2A	NAK	\01A	58	3A	5A	SIGPOLL	\01F	90	5A	126	SIGPOLL	\02B
27	1B	2B	SYN	\01B	59	3B	5B	SIGPOLL	\01F	91	5B	127	SIGPOLL	\02C
28	1C	2C	ENQ	\01C	60	3C	5C	SIGPOLL	\01F	92	5C	128	SIGPOLL	\02D
29	1D	2D	ACK	\01D	61	3D	5D	SIGPOLL	\01F	93	5D	129	SIGPOLL	\02E
30	1E	2E	DC1	\01E	62	3E	5E	SIGPOLL	\01F	94	5E	130	SIGPOLL	\02F
31	1F	2F	DC2	\01F	63	3F	5F	SIGPOLL	\01F	95	5F	131	SIGPOLL	\030
32	20	30	DC3	\020	64	40	60	SIGPOLL	\01F	96	60	132	SIGPOLL	\031
33	21	31	DC4	\021	65	41	61	SIGPOLL	\01F	97	61	133	SIGPOLL	\032
34	22	32	NAK	\022	66	42	62	SIGPOLL	\01F	98	62	134	SIGPOLL	\033
35	23	33	SYN	\023	67	43	63	SIGPOLL	\01F	99	63	135	SIGPOLL	\034
36	24	34	ENQ	\024	68	44	64	SIGPOLL	\01F	100	64	136	SIGPOLL	\035
37	25	35	ACK	\025	69	45	65	SIGPOLL	\01F	101	65	137	SIGPOLL	\036
38	26	36	DC1	\026	70	46	66	SIGPOLL	\01F	102	66	138	SIGPOLL	\037
39	27	37	DC2	\027	71	47	67	SIGPOLL	\01F	103	67	139	SIGPOLL	\038
40	28	38	DC3	\028	72	48	68	SIGPOLL	\01F	104	68	140	SIGPOLL	\039
41	29	39	DC4	\029	73	49	69	SIGPOLL	\01F	105	69	141	SIGPOLL	\03A
42	2A	3A	NAK	\02A	74	4A	6A	SIGPOLL	\01F	106	6A	142	SIGPOLL	\03B
43	2B	3B	SYN	\02B	75	4B	6B	SIGPOLL	\01F	107	6B	143	SIGPOLL	\03C
44	2C	3C	ENQ	\02C	76	4C	6C	SIGPOLL	\01F	108	6C	144	SIGPOLL	\03D
45	2D	3D	ACK	\02D	77	4D	6D	SIGPOLL	\01F	109	6D	145	SIGPOLL	\03E
46	2E	3E	DC1	\02E	78	4E	6E	SIGPOLL	\01F	110	6E	146	SIGPOLL	\03F
47	2F	3F	DC2	\02F	79	4F	6F	SIGPOLL	\01F	111	6F	147	SIGPOLL	\040
48	30	40	DC3	\030	80	50	70	SIGPOLL	\01F	112	70	148	SIGPOLL	\041
49	31	41	DC4	\031	81	51	71	SIGPOLL	\01F	113	71	149	SIGPOLL	\042
50	32	42	NAK	\032	82	52	72	SIGPOLL	\01F	114	72	150	SIGPOLL	\043
51	33	43	SYN	\033	83	53	73	SIGPOLL	\01F	115	73	151	SIGPOLL	\044
52	34	44	ENQ	\034	84	54	74	SIGPOLL	\01F	116	74	152	SIGPOLL	\045
53	35	45	ACK	\035	85	55	75	SIGPOLL	\01F	117	75	153	SIGPOLL	\046
54	36	46	DC1	\036	86	56	76	SIGPOLL	\01F	118	76	154	SIGPOLL	\047
55	37	47	DC2	\037	87	57	77	SIGPOLL	\01F	119	77	155	SIGPOLL	\048
56	38	48	DC3	\038	88	58	78	SIGPOLL	\01F	120	78	156	SIGPOLL	\049
57	39	49	DC4	\039	89	59	79	SIGPOLL	\01F	121	79	157	SIGPOLL	\04A
58	3A	4A	NAK	\03A	90	5A	7A	SIGPOLL	\01F	122	7A	158	SIGPOLL	\04B
59	3B	4B	SYN	\03B	91	5B	7B	SIGPOLL	\01F	123	7B	159	SIGPOLL	\04C
60	3C	4C	ENQ	\03C	92	5C	7C	SIGPOLL	\01F	124	7C	160	SIGPOLL	\04D
61	3D	4D	ACK	\03D	93	5D	7D	SIGPOLL	\01F	125	7D	161	SIGPOLL	\04E
62	3E	4E	DC1	\03E	94	5E	7E	SIGPOLL	\01F	126	7E	162	SIGPOLL	\04F
63	3F	4F	DC2	\03F	95	5F	7F	SIGPOLL	\01F	127	7F	163	SIGPOLL	\050
64	40	50	DC3	\040	96	60	80	SIGPOLL	\01F	128	80	164	SIGPOLL	\051
65	41	51	DC4	\041	97	61	81	SIGPOLL	\01F	129	81	165	SIGPOLL	\052
66	42	52	NAK	\042	98	62	82	SIGPOLL	\01F	130	82	166	SIGPOLL	\053
67	43	53	SYN	\043	99	63	83	SIGPOLL	\01F	131	83	167	SIGPOLL	\054
68	44	54	ENQ	\044	100	64	84	SIGPOLL	\01F	132	84	168	SIGPOLL	\055
69	45	55	ACK	\045	101	65	85	SIGPOLL	\01F	133	85	169	SIGPOLL	\056
70	46	56	DC1	\046	102	66	86	SIGPOLL	\01F	134	86	170	SIGPOLL	\057
71	47	57	DC2	\047	103	67	87	SIGPOLL	\01F	135	87	171	SIGPOLL	\058
72	48	58	DC3	\048	104	68	88	SIGPOLL	\01F	136	88	172	SIGPOLL	\059
73	49	59	DC4	\049	105	69	89	SIGPOLL	\01F	137	89	173	SIGPOLL	\05A
74	4A	5A	NAK	\04A	106	6A	8A	SIGPOLL	\01F	138	8A	174	SIGPOLL	\05B
75	4B	5B	SYN	\04B	107	6B	8B	SIGPOLL	\01F	139	8B	175	SIGPOLL	\05C
76	4C	5C	ENQ	\04C	108	6C	8C	SIGPOLL	\01F	140	8C	176	SIGPOLL	\05D
77	4D	5D	ACK	\04D	109	6D	8D	SIGPOLL	\01F	141	8D	177	SIGPOLL	\05E
78	4E	5E	DC1	\04E	110	6E	8E	SIGPOLL	\01F	142	8E	178	SIGPOLL	\05F
79	4F	5F	DC2	\04F	111	6F	8F	SIGPOLL	\01F	143	8F	179	SIGPOLL	\060
80	50	60	DC3	\050	112	70	90	SIGPOLL	\01F	144	90	180	SIGPOLL	\061
81	51	61	DC4	\051	113	71	91	SIGPOLL	\01F	145	91	181	SIGPOLL	\062
82	52	62	NAK	\052	114	72	92	SIGPOLL	\01F	146	92	182	SIGPOLL	\063
83	53	63	SYN	\053	115	73	93	SIGPOLL	\01F	147	93	183	SIGPOLL	\064
84	54	64	ENQ	\054	116	74	94	SIGPOLL	\01F	148	94	184	SIGPOLL	\065
85	55	65	ACK	\055	117	75	95	SIGPOLL	\01F	149	95	185	SIGPOLL	\066
86	56	66	DC1	\056	118	76	96	SIGPOLL	\01F	150	96	186	SIGPOLL	\067
87	57	67	DC2	\057	119	77	97	SIGPOLL	\01F	151	97	187	SIGPOLL	\068
88	58	68	DC3	\058	120	78	98	SIGPOLL	\01F	152	98	188	SIGPOLL	\069
89	59	69	DC4	\059	121	79	99	SIGPOLL	\01F	153	99	189	SIGPOLL	\06A
90	5A	6A	NAK	\05A	122	7A	9A	SIGPOLL	\01F	154	9A	190	SIGPOLL	\06B
91	5B	6B	SYN	\05B	123	7B	9B	SIGPOLL	\01F	155	9B	191	SIGPOLL	\06C
92	5C	6C	ENQ	\05C	124	7C	9C	SIGPOLL	\01F	156	9C	192	SIGPOLL	\06D
93	5D	6D	ACK	\05D	125	7D	9D	SIGPOLL	\01F	157	9D	193	SIGPOLL	\06E
94	5E	6E	DC1	\05E	126	7E	9E	SIGPOLL	\01F	158	9E	194	SIGPOLL	\06F
95	5F	6F	DC2	\05F	127	7F	9F	SIGPOLL	\01F	159	9F	195	SIGPOLL	\070
96	60	70	DC3	\060	128	80	A0	SIGPOLL	\01F	160	A0	196	SIGPOLL	\071
97	61	71	DC4	\061	129	81	A1	SIGPOLL	\01F	161	A1	197	SIGPOLL	\072
98	62	72	NAK	\062	130	82	A2	SIGPOLL	\01F	162	A2	198	SIGPOLL	\073
99	63	73	SYN	\063	131	83	A3	SIGPOLL	\01F	163	A3	199	SIGPOLL	\074
100	64	74	ENQ	\064	132	84	A4	SIGPOLL	\01F	164	A4	200	SIGPOLL	\075
101	65	75	ACK	\065	133	85	A5	SIGPOLL	\01F	165	A5	201	SIGPOLL	\076
102	66	76	DC1	\066	134	86	A6	SIGPOLL	\01F	166	A6	202	SIGPOLL	\077
103	67	77												

- `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 c
8      print(chr(ord(c)+1)) #Print the next
9          character
10     newMessage = newMessage + chr(ord(c)+1) #
11         add to the new message
12
13     print("The coded message is", newMessage)
```

[link to python turtor demo](#)

In Pairs or Triples...

Predict what will be printed.

```
1 for c in "World":  
2     print(c, c, '#')
```

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

Enter name and the year you are a freshman, print and calculate graduate year. *Covered in detail in Lab 2:*

```
1 name = input("Enter name: ")
2 year = int(input("Enter freshman year: "))
3
4 print("Hello, " + name) #same as print(
    Hello,", name)
5 graduateYear = year + 4
6 print("Will graduate in "+str(graduateYear))
7 #same:print("Will graduate in",graduateYear)
8 #str(graduateYear) converts int graduateYear
    to string. Suppose graduateYear is 2023,
    then str(graduateYear) is "2023".
```

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

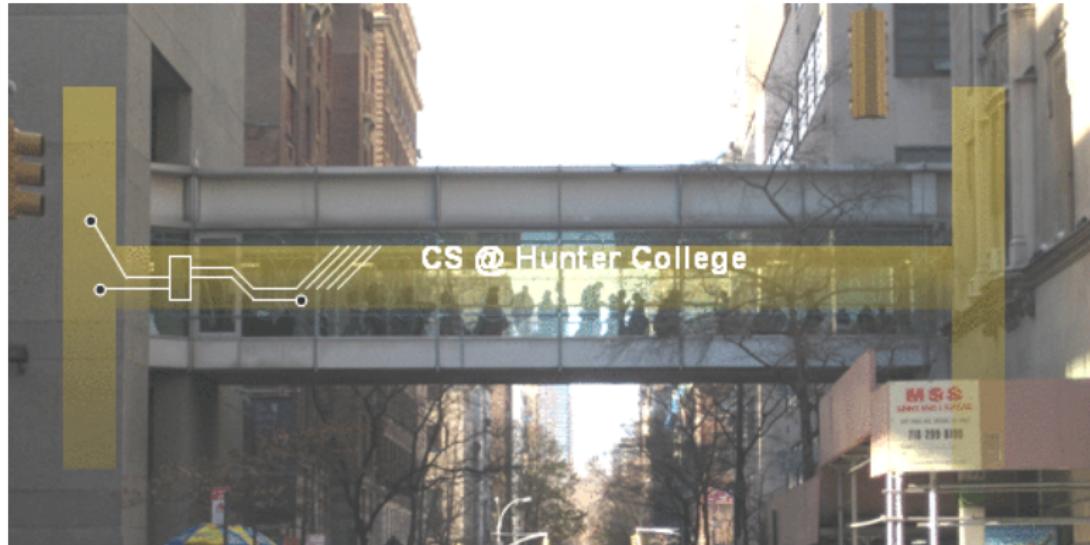
Weekly Reminders!



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.