

CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

Frequently Asked Questions

From email

Frequently Asked Questions

From email

- **I am not sure how to submit the Lab.**

Frequently Asked Questions

From email

- **I am not sure how to submit the Lab.**
You don't submit the lab, you read the lab.

Frequently Asked Questions

From email

- **I am not sure how to submit the Lab.**

You don't submit the lab, you read the lab.

When you are done, start working on this week's 5 programming assignments (this week we will be working on programs 6-10)

Frequently Asked Questions

From email

- **I am not sure how to submit the Lab.**

You don't submit the lab, you read the lab.

When you are done, start working on this week's 5 programming assignments (this week we will be working on programs 6-10)

- **Can I work ahead?**

Frequently Asked Questions

From email

- **I am not sure how to submit the Lab.**

You don't submit the lab, you read the lab.

When you are done, start working on this week's 5 programming assignments (this week we will be working on programs 6-10)

- **Can I work ahead?**

Absolutely! Submission is open on Gradescope, 3 weeks before the deadline.

Frequently Asked Questions

From email

- **I am not sure how to submit the Lab.**

You don't submit the lab, you read the lab.

When you are done, start working on this week's 5 programming assignments (this week we will be working on programs 6-10)

- **Can I work ahead?**

Absolutely! Submission is open on Gradescope, 3 weeks before the deadline.

IMPORTANT: Students who work on the due dates in this class tend to miss deadlines and fall behind. If, instead, you work on programs the week of the associated lecture, you will have time to ask for help if you get stuck and still make the deadline.

Frequently Asked Questions

From email

- **I am not sure how to submit the Lab.**

You don't submit the lab, you read the lab.

When you are done, start working on this week's 5 programming assignments (this week we will be working on programs 6-10)

- **Can I work ahead?**

Absolutely! Submission is open on Gradescope, 3 weeks before the deadline.

IMPORTANT: Students who work on the due dates in this class tend to miss deadlines and fall behind. If, instead, you work on programs the week of the associated lecture, you will have time to ask for help if you get stuck and still make the deadline.

- **When is the midterm?**

Frequently Asked Questions

From email

- **I am not sure how to submit the Lab.**

You don't submit the lab, you read the lab.

When you are done, start working on this week's 5 programming assignments (this week we will be working on programs 6-10)

- **Can I work ahead?**

Absolutely! Submission is open on Gradescope, 3 weeks before the deadline.

IMPORTANT: Students who work on the due dates in this class tend to miss deadlines and fall behind. If, instead, you work on programs the week of the associated lecture, you will have time to ask for help if you get stuck and still make the deadline.

- **When is the midterm?**

There is no midterm. Instead there's required weekly quizzes, code reviews and programming assignments.

Seat Number

- For contact tracing purposes, the College requests that you **remain in the same seat for the entire semester.**

Seat Number

- For contact tracing purposes, the College requests that you **remain in the same seat for the entire semester.**
- Please, **write down the row and seat number** you are seating in and continue to seat there for the rest of the semester.

Seat Number

- For contact tracing purposes, the College requests that you **remain in the same seat for the entire semester.**
- Please, **write down the row and seat number** you are seating in and continue to seat there for the rest of the semester.
- **Submit your row and seat number** using this form: bit.ly/127_seat_number

Seat Number

- For contact tracing purposes, the College requests that you **remain in the same seat for the entire semester.**
- Please, **write down the row and seat number** you are seating in and continue to seat there for the rest of the semester.
- **Submit your row and seat number** using this form: bit.ly/127_seat_number
- The link to the form can also be found on Blackboard under Announcements.

Today's Topics



- For-loops
- `range()`
- Variables
- Characters
- Strings
- Guests: Internships & Clubs

Today's Topics



- **For-loops**
- `range()`
- Variables
- Characters
- Strings
- Guests: Internships & Clubs

Group Work

Some review and some novel challenges:

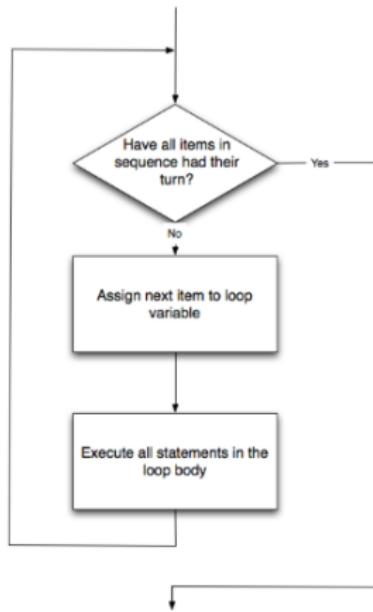
```
1 #Predict what will be printed:  
2 for i in range(4):  
3     print('The world turned upside down')  
4 for j in [0,1,2,3,4,5]:  
5     print(j)  
6 for count in range(6):  
7     print(count)  
8 for color in ['red', 'green', 'blue']:  
9     print(color)  
10    for i in range(2):  
11      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

Python Tutor

```
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(j)  
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!')
```

(Demo with pythonTutor)

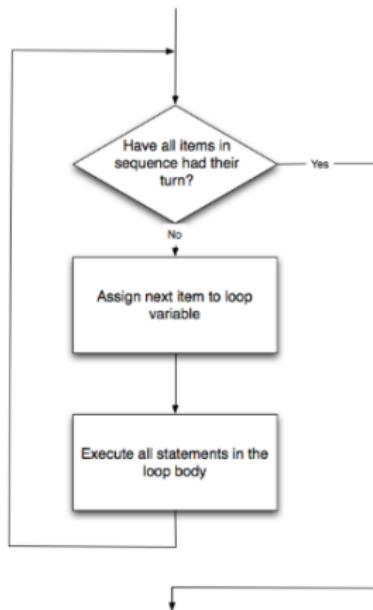
for-loop



```
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
- Guests: Internships & Clubs

More on range():

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

Python Tutor

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

(Demo with pythonTutor)

range()

Simplest version:

- `range(stop)`



range()



Simplest version:

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

range()



Simplest version:

- `range(stop)`
- Produces a list: `[0,1,2,3,...,stop-1]`
- For example, if you want the list `[0,1,2,3,...,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
 $[5, 10, \dots, 50]$
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
- Guests: Internships & Clubs

Variables

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



Variables

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



Variables

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



Variables



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

Variables



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

Variables



- A **variable** is a reserved memory location for storing a value.
- Different kinds, or **types**, of values need different amounts of space:
 - ▶ **int**: integer or whole numbers
 - ▶ **float**: floating point or real numbers
 - ▶ **string**: sequence of characters
 - ▶ **list**: a sequence of items
 - e.g. [3, 1, 4, 5, 9] or
 - [‘violet’, ‘purple’, ‘indigo’]

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.



Variable Names

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



Variable Names



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

Variable Names



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

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
- Guests: Internships & Clubs

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	Decimal	Hex	Char	Octal	Decimal	Hex	Char	Octal
0	00	\0	000	1	01	\1	001	2	02	\2	002
3	03	\3	003	4	04	\4	004	5	05	\5	005
6	06	\6	006	7	07	\7	007	8	08	\8	008
9	09	\9	009	10	0A	\A	010	11	0B	\B	011
12	0C	\C	014	13	0D	\D	015	14	0E	\E	016
15	0F	\F	017	16	10	\10	020	17	11	\11	021
18	12	\12	022	19	13	\13	023	20	14	\14	024
21	16	\16	026	22	17	\17	027	23	18	\18	028
24	1C	\1C	032	25	1D	\1D	033	26	1E	\1E	034
27	1F	\1F	037	28	20	\20	040	29	21	\21	041
30	22	\22	042	31	23	\23	043	32	24	\24	044
33	26	\26	046	34	27	\27	047	35	28	\28	048
36	2C	\2C	052	37	2D	\2D	053	38	2E	\2E	054
39	2F	\2F	057	40	30	\30	060	41	31	\31	061
42	32	\32	062	43	33	\33	063	44	34	\34	064
46	36	\36	066	47	37	\37	067	48	38	\38	070
49	39	\39	071	50	3A	\3A	072	51	3B	\3B	073
52	3C	\3C	074	53	3D	\3D	075	54	3E	\3E	076
56	3F	\3F	077	57	40	\40	080	58	41	\41	081
59	42	\42	082	60	43	\43	083	61	44	\44	084
62	46	\46	086	63	47	\47	087	64	48	\48	088
65	49	\49	089	66	4A	\4A	090	67	4B	\4B	091
68	4C	\4C	092	69	4D	\4D	093	70	4E	\4E	094
71	4F	\4F	095	72	50	\50	096	73	51	\51	097
74	52	\52	098	75	53	\53	099	76	54	\54	100
77	56	\56	101	78	57	\57	102	79	58	\58	103
80	59	\59	104	81	5A	\5A	105	82	5B	\5B	106
83	5C	\5C	108	84	5D	\5D	109	85	5E	\5E	110
86	5F	\5F	111	87	60	\60	112	88	61	\61	113
89	62	\62	114	90	63	\63	115	91	64	\64	116
92	66	\66	118	93	67	\67	119	94	68	\68	120
95	69	\69	121	96	6A	\6A	122	97	6B	\6B	123
98	6C	\6C	125	99	6D	\6D	126	100	6E	\6E	127

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
' '	40	20	'A'	41	25
'A'	65	41	'B'	66	42
'B'	66	42	'C'	67	43
'C'	67	43	'D'	68	44
'D'	68	44	'E'	69	45
'E'	69	45	'F'	70	46
'F'	70	46	'G'	71	47
'G'	71	47	'H'	72	48
'H'	72	48	'I'	73	49
'I'	73	49	'J'	74	4A
'J'	74	4A	'K'	75	4B
'K'	75	4B	'L'	76	4C
'L'	76	4C	'M'	77	4D
'M'	77	4D	'N'	78	4E
'N'	78	4E	'O'	79	4F
'O'	79	4F	'P'	80	50
'P'	80	50	'Q'	81	51
'Q'	81	51	'R'	82	52
'R'	82	52	'S'	83	53
'S'	83	53	'T'	84	54
'T'	84	54	'U'	85	55
'U'	85	55	'V'	86	56
'V'	86	56	'W'	87	57
'W'	87	57	'X'	88	58
'X'	88	58	'Y'	89	59
'Y'	89	59	'Z'	90	5A
'Z'	90	5A	'[space]'	91	5B
'[space]'	91	5B	'`'	92	5C
'`'	92	5C	'`'	93	5D
'`'	93	5D	'`'	94	5E
'`'	94	5E	'`'	95	5F
'`'	95	5F	'`'	96	60
'`'	96	60	'`'	97	61
'`'	97	61	'`'	98	62
'`'	98	62	'`'	99	63
'`'	99	63	'`'	100	64
'`'	100	64	'`'	101	65
'`'	101	65	'`'	102	66
'`'	102	66	'`'	103	67
'`'	103	67	'`'	104	68
'`'	104	68	'`'	105	69
'`'	105	69	'`'	106	6A
'`'	106	6A	'`'	107	6B
'`'	107	6B	'`'	108	6C
'`'	108	6C	'`'	109	6D
'`'	109	6D	'`'	110	6E
'`'	110	6E	'`'	111	6F
'`'	111	6F	'`'	112	70
'`'	112	70	'`'	113	71
'`'	113	71	'`'	114	72
'`'	114	72	'`'	115	73
'`'	115	73	'`'	116	74
'`'	116	74	'`'	117	75
'`'	117	75	'`'	118	76
'`'	118	76	'`'	119	77
'`'	119	77	'`'	120	78
'`'	120	78	'`'	121	79
'`'	121	79	'`'	122	7A
'`'	122	7A	'`'	123	7B
'`'	123	7B	'`'	124	7C
'`'	124	7C	'`'	125	7D
'`'	125	7D	'`'	126	7E
'`'	126	7E	'`'	127	7F
'`'	127	7F	'`'	128	80
'`'	128	80	'`'	129	81
'`'	129	81	'`'	130	82
'`'	130	82	'`'	131	83
'`'	131	83	'`'	132	84
'`'	132	84	'`'	133	85
'`'	133	85	'`'	134	86
'`'	134	86	'`'	135	87
'`'	135	87	'`'	136	88
'`'	136	88	'`'	137	89
'`'	137	89	'`'	138	8A
'`'	138	8A	'`'	139	8B
'`'	139	8B	'`'	140	8C
'`'	140	8C	'`'	141	8D
'`'	141	8D	'`'	142	8E
'`'	142	8E	'`'	143	8F
'`'	143	8F	'`'	144	90
'`'	144	90	'`'	145	91
'`'	145	91	'`'	146	92
'`'	146	92	'`'	147	93
'`'	147	93	'`'	148	94
'`'	148	94	'`'	149	95
'`'	149	95	'`'	150	96
'`'	150	96	'`'	151	97
'`'	151	97	'`'	152	98
'`'	152	98	'`'	153	99
'`'	153	99	'`'	154	9A
'`'	154	9A	'`'	155	9B
'`'	155	9B	'`'	156	9C
'`'	156	9C	'`'	157	9D
'`'	157	9D	'`'	158	9E
'`'	158	9E	'`'	159	9F
'`'	159	9F	'`'	160	A0
'`'	160	A0	'`'	161	A1
'`'	161	A1	'`'	162	A2
'`'	162	A2	'`'	163	A3
'`'	163	A3	'`'	164	A4
'`'	164	A4	'`'	165	A5
'`'	165	A5	'`'	166	A6
'`'	166	A6	'`'	167	A7
'`'	167	A7	'`'	168	A8
'`'	168	A8	'`'	169	A9
'`'	169	A9	'`'	170	A0
'`'	170	A0	'`'	171	A1
'`'	171	A1	'`'	172	A2
'`'	172	A2	'`'	173	A3
'`'	173	A3	'`'	174	A4
'`'	174	A4	'`'	175	A5
'`'	175	A5	'`'	176	A6
'`'	176	A6	'`'	177	A7
'`'	177	A7	'`'	178	A8
'`'	178	A8	'`'	179	A9
'`'	179	A9	'`'	180	A0
'`'	180	A0	'`'	181	A1
'`'	181	A1	'`'	182	A2
'`'	182	A2	'`'	183	A3
'`'	183	A3	'`'	184	A4
'`'	184	A4	'`'	185	A5
'`'	185	A5	'`'	186	A6
'`'	186	A6	'`'	187	A7
'`'	187	A7	'`'	188	A8
'`'	188	A8	'`'	189	A9
'`'	189	A9	'`'	190	A0
'`'	190	A0	'`'	191	A1
'`'	191	A1	'`'	192	A2
'`'	192	A2	'`'	193	A3
'`'	193	A3	'`'	194	A4
'`'	194	A4	'`'	195	A5
'`'	195	A5	'`'	196	A6
'`'	196	A6	'`'	197	A7
'`'	197	A7	'`'	198	A8
'`'	198	A8	'`'	199	A9
'`'	199	A9	'`'	200	A0

- `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	Binary	Hex	Char	Binary	Hex	Char	Binary	Hex	Char
	00000000	00		00000001	01		00000010	02	
	00000002	03		00000003	04		00000011	0B	
	00000004	05		00000005	06		00000012	0A	
	00000006	07		00000007	08		00000013	0B	
	00000008	09		00000009	0A		00000014	0C	
	0000000A	0B		0000000B	0C		00000015	0D	
	0000000C	0E		0000000D	0F		00000016	0E	
	0000000E	0F		0000000F	10		00000017	0F	
	00000010	11		00000011	12		00000018	10	
	00000012	13		00000013	14		00000019	11	
	00000014	15		00000015	16		0000001A	1B	
	00000016	17		00000017	18		0000001B	1D	
	00000018	19		00000019	1A		0000001C	1E	
	0000001A	1B		0000001B	1C		0000001D	1F	
	0000001C	1D		0000001D	1E		0000001E	1E	
	0000001E	1F		0000001F	1F		0000001F	1F	
	00000020	20		00000021	21		00000022	22	
	00000022	23		00000023	24		00000024	25	
	00000024	26		00000025	27		00000025	25	
	00000026	27		00000027	28		00000026	26	
	00000028	29		00000029	2A		00000027	25	
	0000002A	2B		0000002B	2C		00000028	24	
	0000002C	2D		0000002D	2E		00000029	25	
	0000002E	2F		0000002F	2F		0000002A	26	
	00000030	30		00000031	31		00000032	32	
	00000032	33		00000033	34		00000034	35	
	00000034	36		00000035	37		00000035	35	
	00000036	38		00000037	39		00000036	36	
	00000038	3A		00000039	3B		00000037	35	
	0000003A	3C		0000003B	3D		00000038	36	
	0000003C	3E		0000003D	3F		00000039	35	
	0000003E	3F		0000003F	3F		0000003A	36	
	00000040	40		00000041	41		00000042	42	
	00000042	43		00000043	44		00000044	45	
	00000044	46		00000045	47		00000045	45	
	00000046	48		00000047	49		00000046	46	
	00000048	4A		00000049	4B		00000047	45	
	0000004A	4C		0000004B	4D		00000048	46	
	0000004C	4E		0000004D	4F		00000049	45	
	0000004E	4F		0000004F	4F		0000004A	46	
	00000050	50		00000051	51		00000052	52	
	00000052	53		00000053	54		00000054	55	
	00000054	56		00000055	57		00000055	55	
	00000056	58		00000057	59		00000056	56	
	00000058	5A		00000059	5B		00000057	55	
	0000005A	5C		0000005B	5D		00000058	56	
	0000005C	5E		0000005D	5F		00000059	55	
	0000005E	5F		0000005F	5F		0000005A	56	
	00000060	60		00000061	61		00000062	62	
	00000062	63		00000063	64		00000064	65	
	00000064	66		00000065	67		00000065	65	
	00000066	68		00000067	69		00000066	66	
	00000068	6A		00000069	6B		00000067	65	
	0000006A	6C		0000006B	6D		00000068	66	
	0000006C	6E		0000006D	6F		00000069	65	
	0000006E	6F		0000006F	6F		0000006A	66	
	00000070	70		00000071	71		00000072	72	
	00000072	73		00000073	74		00000074	75	
	00000074	76		00000075	77		00000075	75	
	00000076	78		00000077	79		00000076	76	
	00000078	7A		00000079	7B		00000077	75	
	0000007A	7C		0000007B	7D		00000078	76	
	0000007C	7E		0000007D	7F		00000079	75	
	0000007E	7F		0000007F	7F		0000007A	76	
	00000080	80		00000081	81		00000082	82	
	00000082	83		00000083	84		00000084	85	
	00000084	86		00000085	87		00000085	85	
	00000086	88		00000087	89		00000086	86	
	00000088	8A		00000089	8B		00000087	85	
	0000008A	8C		0000008B	8D		00000088	86	
	0000008C	8E		0000008D	8F		00000089	85	
	0000008E	8F		0000008F	8F		0000008A	86	
	00000090	90		00000091	91		00000092	92	
	00000092	93		00000093	94		00000094	95	
	00000094	96		00000095	97		00000095	95	
	00000096	98		00000097	99		00000096	96	
	00000098	9A		00000099	9B		00000097	95	
	0000009A	9C		0000009B	9D		00000098	96	
	0000009C	9E		0000009D	9F		00000099	95	
	0000009E	9F		0000009F	9F		0000009A	96	
	000000A0	A0		000000A1	A1		000000A2	A2	
	000000A2	A3		000000A3	A4		000000A4	A5	
	000000A4	A6		000000A5	A7		000000A5	A5	
	000000A6	A8		000000A7	A9		000000A6	A6	
	000000A8	AA		000000A9	AB		000000A7	A5	
	000000AA	AC		000000AB	AD		000000A8	A6	
	000000AC	AE		000000AD	AF		000000AB	AD	
	000000AE	AF		000000AF	AF		000000AC	AC	
	000000B0	B0		000000B1	B1		000000B2	B2	
	000000B2	B3		000000B3	B4		000000B4	B5	
	000000B4	B6		000000B5	B7		000000B5	B5	
	000000B6	B8		000000B7	B9		000000B6	B6	
	000000B8	BA		000000B9	BB		000000B7	B5	
	000000BA	BC		000000BB	BD		000000B8	B6	
	000000BC	BE		000000BD	BF		000000BB	BB	
	000000BE	BF		000000BF	BF		000000BC	BC	
	000000C0	C0		000000C1	C1		000000C2	C2	
	000000C2	C3		000000C3	C4		000000C4	C5	
	000000C4	C6		000000C5	C7		000000C5	C5	
	000000C6	C8		000000C7	C9		000000C6	C6	
	000000C8	CA		000000C9	CB		000000C7	C5	
	000000CA	CC		000000CB	CD		000000C8	C6	
	000000CC	CE		000000CD	CF		000000CB	CB	
	000000CE	CF		000000CF	CF		000000CC	CC	
	000000D0	D0		000000D1	D1		000000D2	D2	
	000000D2	D3		000000D3	D4		000000D4	D5	
	000000D4	D6		000000D5	D7		000000D5	D5	
	000000D6	D8		000000D7	D9		000000D6	D6	
	000000D8	DA		000000D9	DB		000000D7	D5	
	000000DA	DC		000000DB	DD		000000D8	D6	
	000000DC	DE		000000DD	DF		000000DB	DB	
	000000DE	DF		000000DF	DF		000000DC	DC	
	000000E0	E0		000000E1	E1		000000E2	E2	
	000000E2	E3		000000E3	E4		000000E4	E5	
	000000E4	E6		000000E5	E7		000000E5	E5	
	000000E6	E8		000000E7	E9		000000E6	E6	
	000000E8	EA		000000E9	EB		000000E7	E5	
	000000EA	EC		000000EB	ED		000000E8	E6	
	000000EC	EE		000000ED	FF		000000EB	EB	
	000000EE	FF		000000FF	FF		000000EC	EC	

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	\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'.)

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	\0B0	068	58	\0B0	068	58	\0B0
89	59	\0B1	069	59	\0B1	069	59	\0B1
90	5A	\0B2	06A	5A	\0B2	06A	5A	\0B2
91	5B	\0B3	06B	5B	\0B3	06B	5B	\0B3
92	5C	\0B4	06C	5C	\0B4	06C	5C	\0B4
93	5D	\0B5	06D	5D	\0B5	06D	5D	\0B5
94	5E	\0B6	06E	5E	\0B6	06E	5E	\0B6
95	5F	\0B7	06F	5F	\0B7	06F	5F	\0B7
96	60	\0C0	070	60	\0C0	070	60	\0C0
97	61	\0C1	071	61	\0C1	071	61	\0C1
98	62	\0C2	072	62	\0C2	072	62	\0C2
99	63	\0C3	073	63	\0C3	073	63	\0C3
100	64	\0C4	074	64	\0C4	074	64	\0C4
101	65	\0C5	075	65	\0C5	075	65	\0C5
102	66	\0C6	076	66	\0C6	076	66	\0C6
103	67	\0C7	077	67	\0C7	077	67	\0C7
104	68	\0D0	078	68	\0D0	078	68	\0D0
105	69	\0D1	079	69	\0D1	079	69	\0D1
106	6A	\0D2	07A	6A	\0D2	07A	6A	\0D2
107	6B	\0D3	07B	6B	\0D3	07B	6B	\0D3
108	6C	\0D4	07C	6C	\0D4	07C	6C	\0D4
109	6D	\0D5	07D	6D	\0D5	07D	6D	\0D5
110	6E	\0D6	07E	6E	\0D6	07E	6E	\0D6
111	6F	\0D7	07F	6F	\0D7	07F	6F	\0D7
112	70	\0E0	080	70	\0E0	080	70	\0E0
113	71	\0E1	081	71	\0E1	081	71	\0E1
114	72	\0E2	082	72	\0E2	082	72	\0E2
115	73	\0E3	083	73	\0E3	083	73	\0E3
116	74	\0E4	084	74	\0E4	084	74	\0E4
117	75	\0E5	085	75	\0E5	085	75	\0E5
118	76	\0E6	086	76	\0E6	086	76	\0E6
119	77	\0E7	087	77	\0E7	087	77	\0E7
120	78	\0F0	088	78	\0F0	088	78	\0F0
121	79	\0F1	089	79	\0F1	089	79	\0F1
122	7A	\0F2	08A	7A	\0F2	08A	7A	\0F2
123	7B	\0F3	08B	7B	\0F3	08B	7B	\0F3
124	7C	\0F4	08C	7C	\0F4	08C	7C	\0F4
125	7D	\0F5	08D	7D	\0F5	08D	7D	\0F5
126	7E	\0F6	08E	7E	\0F6	08E	7E	\0F6
127	7F	\0F7	08F	7F	\0F7	08F	7F	\0F7

- `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...

Some review and some novel challenges:

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



Python Tutor

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

(Demo with pythonTutor)

Wrap



chr()	a	b	c		...		x	y	z
ord()	97	98	99		...		120	121	122



wrap: if offset > 26 then wrap around
% is the remainder
 $27 \% 26 = 1$

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**
- Guests: Internships & Clubs

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)

Lecture Slip

LECTURE 2, CSci 127
SPRING 2022

Name:								
EmpID:								

- **Introducing Design Challenges:** these are "think up an Algorithm"-type exercises. We introduce a topic in lecture, and then we ask you to apply it to solve a problem. Here we are asking you to come up with a sequence of steps (short English sentences) that describe the **process** – i.e. your Algorithm.
You should also identify the **input** and **output**.

1. Design a program that **counts** the number of plural nouns provided as a string containing only the nouns separated by spaces. Think about what the input is, what the output is, and how you can determine if a noun is plural.
Note: To simplify the problem, assume all plural nouns end in "s".

Input:

Output:

Process:

Today's Topics



- For-loops
- `range()`
- Variables
- Characters
- Strings
- **Guests: Internships & Clubs**

Guest Speakers

- Hunter staff
 - ▶ Elise Harris, Internship Manager, Cooperman Business Center and Computer Science
- Club officers
 - ▶ Asad Rafique, Hunter Association of Computing Machinery (ACM)
 - ▶ David Arcos Mawyn, Esports and Game Design Collective (EGD)
 - ▶ Kelly Camacho, Women in Computer Science (WiCS)
 - ▶ Isabel Abonitalla, Google Developers Student Club (DSC)
- See Announcement on Blackboard for links to important resources.

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(j)  
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(j)  
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(j)  
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(j)  
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(j)  
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(j)  
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(j)  
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(j)  
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(j)  
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 1001E 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 1001E Hunter North
- If you haven't already, schedule an appointment to take the Code Review (**one every two weeks**) in lab 1001E 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 1001E Hunter North
- If you haven't already, schedule an appointment to take the Code Review (**one every two weeks**) in lab 1001E 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 1001E Hunter North
- If you haven't already, schedule an appointment to take the Code Review (**one every two weeks**) in lab 1001E Hunter North
- Submit this week's 5 programming assignments (**programs 6-10**)
- If you need help, schedule an appointment for Tutoring in lab 1001E 11am-5pm

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 1001E Hunter North
- If you haven't already, schedule an appointment to take the Code Review (**one every two weeks**) in lab 1001E Hunter North
- Submit this week's 5 programming assignments (**programs 6-10**)
- If you need help, schedule an appointment for Tutoring in lab 1001E 11am-5pm
- 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.