

CSci 127: Introduction to Computer Science



hunter.cuny.edu/csci

Today's Topics



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

In Pairs or Triples...

Some review and some novel challenges:

```
1 #Predict what will be printed:
2 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)

Variables



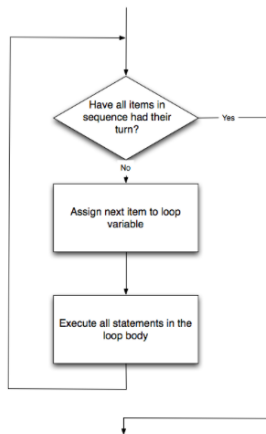
- A **variable** is a reserved memory location for storing a value.
- Different kinds, or **types**, of values need different amounts of space:
 - ▶ **int**: integer or whole numbers
 - ▶ **float**: floating point or real numbers
 - ▶ **string**: sequence of characters
 - ▶ **list**: a sequence of items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

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 (like `for`).
(List of reserved words in *Think CS*, §2.5.)

for-loop



How to Think Like CS, §4.5

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

where `list` is a list of items:

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

In Pairs or Triples...

Some review and some novel challenges:

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


Python Tutor

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

(Demo with pythonTutor)

range()



Simplest version:

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

```
range(101)
```

range()

What if you wanted to start somewhere else:



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

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

range()

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

- `range(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)
- For example, if you want the the list `[5, 10, ..., 50]` you would write:

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



In summary: `range()`



The three versions:

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

Standardized Code for Characters

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


ASCII TABLE

| Decimal | Hex | Char | Decimal | Hex | Char | Decimal | Hex | Char | Decimal | Hex | Char |
|---------|-----|------------------------|---------|-----|---------|---------|-----|------|---------|-----|-------|
| 0 | 0 | [NULL] | 32 | 20 | [SPACE] | 64 | 40 | @ | 96 | 60 | ` |
| 1 | 1 | [START OF HEADING] | 33 | 21 | ! | 65 | 41 | A | 97 | 61 | a |
| 2 | 2 | [START OF TEXT] | 34 | 22 | " | 66 | 42 | B | 98 | 62 | b |
| 3 | 3 | [END OF TEXT] | 35 | 23 | # | 67 | 43 | C | 99 | 63 | c |
| 4 | 4 | [END OF TRANSMISSION] | 36 | 24 | \$ | 68 | 44 | D | 100 | 64 | d |
| 5 | 5 | [ENQUIRY] | 37 | 25 | % | 69 | 45 | E | 101 | 65 | e |
| 6 | 6 | [ACKNOWLEDGE] | 38 | 26 | & | 70 | 46 | F | 102 | 66 | f |
| 7 | 7 | [BELL] | 39 | 27 | ' | 71 | 47 | G | 103 | 67 | g |
| 8 | 8 | [BACKSPACE] | 40 | 28 | (| 72 | 48 | H | 104 | 68 | h |
| 9 | 9 | [HORIZONTAL TAB] | 41 | 29 |) | 73 | 49 | I | 105 | 69 | i |
| 10 | A | [LINE FEED] | 42 | 2A | * | 74 | 4A | J | 106 | 6A | j |
| 11 | B | [VERTICAL TAB] | 43 | 2B | + | 75 | 4B | K | 107 | 6B | k |
| 12 | C | [FORM FEED] | 44 | 2C | , | 76 | 4C | L | 108 | 6C | l |
| 13 | D | [CARRIAGE RETURN] | 45 | 2D | - | 77 | 4D | M | 109 | 6D | m |
| 14 | E | [SHIFT OUT] | 46 | 2E | . | 78 | 4E | N | 110 | 6E | n |
| 15 | F | [SHIFT IN] | 47 | 2F | / | 79 | 4F | O | 111 | 6F | o |
| 16 | 10 | [DATA LINK ESCAPE] | 48 | 30 | 0 | 80 | 50 | P | 112 | 70 | p |
| 17 | 11 | [DEVICE CONTROL 1] | 49 | 31 | 1 | 81 | 51 | Q | 113 | 71 | q |
| 18 | 12 | [DEVICE CONTROL 2] | 50 | 32 | 2 | 82 | 52 | R | 114 | 72 | r |
| 19 | 13 | [DEVICE CONTROL 3] | 51 | 33 | 3 | 83 | 53 | S | 115 | 73 | s |
| 20 | 14 | [DEVICE CONTROL 4] | 52 | 34 | 4 | 84 | 54 | T | 116 | 74 | t |
| 21 | 15 | [NEGATIVE ACKNOWLEDGE] | 53 | 35 | 5 | 85 | 55 | U | 117 | 75 | u |
| 22 | 16 | [SYNCHRONOUS IDLE] | 54 | 36 | 6 | 86 | 56 | V | 118 | 76 | v |
| 23 | 17 | [ENG OF TRANS. BLOCK] | 55 | 37 | 7 | 87 | 57 | W | 119 | 77 | w |
| 24 | 18 | [CANCEL] | 56 | 38 | 8 | 88 | 58 | X | 120 | 78 | x |
| 25 | 19 | [END OF MEDIUM] | 57 | 39 | 9 | 89 | 59 | Y | 121 | 79 | y |
| 26 | 1A | [SUBSTITUTE] | 58 | 3A | : | 90 | 5A | Z | 122 | 7A | z |
| 27 | 1B | [ESCAPE] | 59 | 3B | ; | 91 | 5B | [| 123 | 7B | { |
| 28 | 1C | [FILE SEPARATOR] | 60 | 3C | < | 92 | 5C | \ | 124 | 7C | |
| 29 | 1D | [GROUP SEPARATOR] | 61 | 3D | = | 93 | 5D |] | 125 | 7D | } |
| 30 | 1E | [RECORD SEPARATOR] | 62 | 3E | > | 94 | 5E | ^ | 126 | 7E | ~ |
| 31 | 1F | [UNIT SEPARATOR] | 63 | 3F | ? | 95 | 5F | _ | 127 | 7F | [DEL] |

Converting from Character to Code:

(There is an ASCII table on the back of today's lecture slip.)

ASCII TABLE



| Decimal | Hex | Char | Decimal | Hex | Char | Decimal | Hex | Char | Decimal | Hex | Char |
|---------|-----|-------|---------|-----|-------|---------|-----|------|---------|-----|------|
| 0 | 00 | | 32 | 20 | Space | 64 | 40 | (| 96 | 60 | < |
| 1 | 01 | | 33 | 21 | ! | 65 | 41 |) | 97 | 61 | > |
| 2 | 02 | | 34 | 22 | " | 66 | 42 | * | 98 | 62 | ? |
| 3 | 03 | | 35 | 23 | \$ | 67 | 43 | + | 99 | 63 | @ |
| 4 | 04 | | 36 | 24 | % | 68 | 44 | , | 100 | 64 | A |
| 5 | 05 | | 37 | 25 | & | 69 | 45 | - | 101 | 65 | B |
| 6 | 06 | | 38 | 26 | ' | 70 | 46 | . | 102 | 66 | C |
| 7 | 07 | | 39 | 27 | (| 71 | 47 | / | 103 | 67 | D |
| 8 | 08 | | 40 | 28 |) | 72 | 48 | 0 | 104 | 68 | E |
| 9 | 09 | | 41 | 29 | ! | 73 | 49 | 1 | 105 | 69 | F |
| 10 | 0A | | 42 | 2A | " | 74 | 4A | 2 | 106 | 70 | G |
| 11 | 0B | | 43 | 2B | \$ | 75 | 4B | 3 | 107 | 71 | H |
| 12 | 0C | | 44 | 2C | % | 76 | 4C | 4 | 108 | 72 | I |
| 13 | 0D | | 45 | 2D | & | 77 | 4D | 5 | 109 | 73 | J |
| 14 | 0E | | 46 | 2E | ' | 78 | 4E | 6 | 110 | 74 | K |
| 15 | 0F | | 47 | 2F | (| 79 | 4F | 7 | 111 | 75 | L |
| 16 | 10 | | 48 | 30 |) | 80 | 50 | 8 | 112 | 76 | M |
| 17 | 11 | | 49 | 31 | ! | 81 | 51 | 9 | 113 | 77 | N |
| 18 | 12 | | 50 | 32 | " | 82 | 52 | A | 114 | 78 | O |
| 19 | 13 | | 51 | 33 | \$ | 83 | 53 | B | 115 | 79 | P |
| 20 | 14 | | 52 | 34 | % | 84 | 54 | C | 116 | 80 | Q |
| 21 | 15 | | 53 | 35 | & | 85 | 55 | D | 117 | 81 | R |
| 22 | 16 | | 54 | 36 | ' | 86 | 56 | E | 118 | 82 | S |
| 23 | 17 | | 55 | 37 | (| 87 | 57 | F | 119 | 83 | T |
| 24 | 18 | | 56 | 38 |) | 88 | 58 | G | 120 | 84 | U |
| 25 | 19 | | 57 | 39 | ! | 89 | 59 | H | 121 | 85 | V |
| 26 | 1A | | 58 | 3A | " | 90 | 5A | I | 122 | 86 | W |
| 27 | 1B | | 59 | 3B | \$ | 91 | 5B | J | 123 | 87 | X |
| 28 | 1C | | 60 | 3C | % | 92 | 5C | K | 124 | 88 | Y |
| 29 | 1D | | 61 | 3D | & | 93 | 5D | L | 125 | 89 | Z |
| 30 | 1E | | 62 | 3E | ' | 94 | 5E | M | 126 | 8A | [|
| 31 | 1F | | 63 | 3F | (| 95 | 5F | N | 127 | 8B | \ |
| 32 | 20 | Space | 96 | 60 | < | | | | | | |
| 33 | 21 | ! | 97 | 61 | > | | | | | | |
| 34 | 22 | " | 98 | 62 | ? | | | | | | |
| 35 | 23 | \$ | 99 | 63 | @ | | | | | | |
| 36 | 24 | % | 100 | 64 | A | | | | | | |
| 37 | 25 | & | 101 | 65 | B | | | | | | |
| 38 | 26 | ' | 102 | 66 | C | | | | | | |
| 39 | 27 | (| 103 | 67 | D | | | | | | |
| 40 | 28 |) | 104 | 68 | E | | | | | | |
| 41 | 29 | ! | 105 | 69 | F | | | | | | |
| 42 | 2A | " | 106 | 70 | G | | | | | | |
| 43 | 2B | \$ | 107 | 71 | H | | | | | | |
| 44 | 2C | % | 108 | 72 | I | | | | | | |
| 45 | 2D | & | 109 | 73 | J | | | | | | |
| 46 | 2E | ' | 110 | 74 | K | | | | | | |
| 47 | 2F | (| 111 | 75 | L | | | | | | |
| 48 | 30 |) | 112 | 76 | M | | | | | | |
| 49 | 31 | ! | 113 | 77 | N | | | | | | |
| 50 | 32 | " | 114 | 78 | O | | | | | | |
| 51 | 33 | \$ | 115 | 79 | P | | | | | | |
| 52 | 34 | % | 116 | 80 | Q | | | | | | |
| 53 | 35 | & | 117 | 81 | R | | | | | | |
| 54 | 36 | ' | 118 | 82 | S | | | | | | |
| 55 | 37 | (| 119 | 83 | T | | | | | | |
| 56 | 38 |) | 120 | 84 | U | | | | | | |
| 57 | 39 | ! | 121 | 85 | V | | | | | | |
| 58 | 3A | " | 122 | 86 | W | | | | | | |
| 59 | 3B | \$ | 123 | 87 | X | | | | | | |
| 60 | 3C | % | 124 | 88 | Y | | | | | | |
| 61 | 3D | & | 125 | 89 | Z | | | | | | |
| 62 | 3E | ' | | | | | | | | | |
| 63 | 3F | (| | | | | | | | | |
| 64 | 40 |) | | | | | | | | | |
| 65 | 41 | ! | | | | | | | | | |
| 66 | 42 | " | | | | | | | | | |
| 67 | 43 | \$ | | | | | | | | | |
| 68 | 44 | % | | | | | | | | | |
| 69 | 45 | & | | | | | | | | | |
| 70 | 46 | ' | | | | | | | | | |
| 71 | 47 | (| | | | | | | | | |
| 72 | 48 |) | | | | | | | | | |
| 73 | 49 | ! | | | | | | | | | |
| 74 | 4A | " | | | | | | | | | |
| 75 | 4B | \$ | | | | | | | | | |
| 76 | 4C | % | | | | | | | | | |
| 77 | 4D | & | | | | | | | | | |
| 78 | 4E | ' | | | | | | | | | |
| 79 | 4F | (| | | | | | | | | |
| 80 | 50 |) | | | | | | | | | |
| 81 | 51 | ! | | | | | | | | | |
| 82 | 52 | " | | | | | | | | | |
| 83 | 53 | \$ | | | | | | | | | |
| 84 | 54 | % | | | | | | | | | |
| 85 | 55 | & | | | | | | | | | |
| 86 | 56 | ' | | | | | | | | | |
| 87 | 57 | (| | | | | | | | | |
| 88 | 58 |) | | | | | | | | | |
| 89 | 59 | ! | | | | | | | | | |
| 90 | 5A | " | | | | | | | | | |
| 91 | 5B | \$ | | | | | | | | | |
| 92 | 5C | % | | | | | | | | | |
| 93 | 5D | & | | | | | | | | | |
| 94 | 5E | ' | | | | | | | | | |
| 95 | 5F | (| | | | | | | | | |
| 96 | 60 |) | | | | | | | | | |
| 97 | 61 | ! | | | | | | | | | |
| 98 | 62 | " | | | | | | | | | |
| 99 | 63 | \$ | | | | | | | | | |
| 100 | 64 | % | | | | | | | | | |
| 101 | 65 | & | | | | | | | | | |
| 102 | 66 | ' | | | | | | | | | |
| 103 | 67 | (| | | | | | | | | |
| 104 | 68 |) | | | | | | | | | |
| 105 | 69 | ! | | | | | | | | | |
| 106 | 6A | " | | | | | | | | | |
| 107 | 6B | \$ | | | | | | | | | |
| 108 | 6C | % | | | | | | | | | |
| 109 | 6D | & | | | | | | | | | |
| 110 | 6E | ' | | | | | | | | | |
| 111 | 6F | (| | | | | | | | | |
| 112 | 70 |) | | | | | | | | | |
| 113 | 71 | ! | | | | | | | | | |
| 114 | 72 | " | | | | | | | | | |
| 115 | 73 | \$ | | | | | | | | | |
| 116 | 74 | % | | | | | | | | | |
| 117 | 75 | & | | | | | | | | | |
| 118 | 76 | ' | | | | | | | | | |
| 119 | 77 | (| | | | | | | | | |
| 120 | 78 |) | | | | | | | | | |
| 121 | 79 | ! | | | | | | | | | |
| 122 | 7A | " | | | | | | | | | |
| 123 | 7B | \$ | | | | | | | | | |
| 124 | 7C | % | | | | | | | | | |
| 125 | 7D | & | | | | | | | | | |
| 126 | 7E | ' | | | | | | | | | |
| 127 | 7F | (| | | | | | | | | |

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

In Pairs or Triples...

Some review and some novel challenges:

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


Python Tutor

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

(Demo with pythonTutor)

User Input

Covered in detail in Lab 2:

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

(Demo with pythonTutor)

Side Note: '+' for numbers and strings



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

Recap

- On lecture slip, write down a topic you wish we had spent more time (and why).
- In Python, we introduced:

Name _____ EmailID _____ (Scribble Sample Date, FY)

1. [a] What will the following Python code print:

```
names = ["Jan","Feb","Mar","Apr","May","  
Jun","Jul","Aug","Sep","Oct","Nov","Dec"]  
hall = monthDict  
print(hall.get('april'))  
print(hall.get('JULY'))  
print(names[4-1].lower())  
print(names[2-4])  
stairs = 10  
print(stairs//stairs+1)  
tenis = 2  
print(names[(len(names)-1)%2])
```

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

Lecture Slip

Name:

EmpID:

CSci 127 Sample Final, F17

1. (a) What will the following Python code print:

```
months = ["Jan", "Feb", "Mar", "Apr", "May", \
"Jun", "Jul", "Aug", "Sep", "Oct", "Nov", "Dec"]
half = months[6]
print(half.upper())
print(half[0])
print(months[-1].lower())
print(months[2:4])
start = 9
print(months[start-1])
term = 3
print(months[(start+term-1)%12])
```

Output:



Frequently Asked Questions

From lecture slips & recitation sections.

- When is the midterm?
There is no midterm. Instead there's 14 in-class quizzes.
- When is the final?
Tuesday, 22 May, 9-11am.
- Can I submit late homework?
No. Instead we drop the 5 lowest grades.
- I missed class. Do you need documentation?
*No. Missing lecture slip & quiz grades are replaced by your final exam score.
If you will miss ≥ 3 weeks ($> 20\%$), see us about taking this in a future term.*
- Why do I have to work in groups?
It's great practice to explain technical work to others.
- Can I work ahead?
Yes! All programs are available, on gradescope, 4 weeks before the deadline.
- You said “when you take second semester...” I just took this class for Pathways...
*This is Pathways, but we hope that you will be a CS major/minor.
We also hope: “Get your education don't forget whence you came...”*