



# CS 1112: Introduction To Programming

## Strings II



Dr. Nada Basit

basit[at]Virginia[dot]edu

Spring 2024



# Friendly Reminders

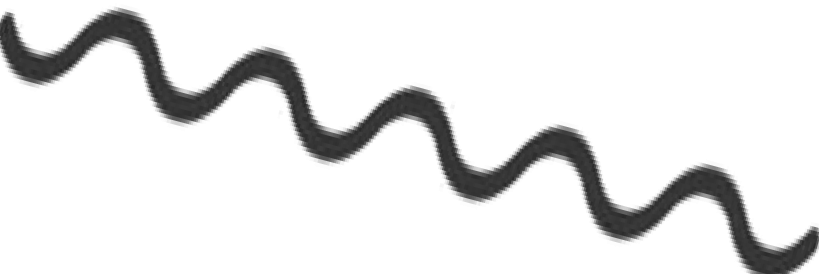
---

- Your **safety** and **comfort** is important!
  - If you choose to wear a mask you are welcome to do so
  - *We will interpret wearing a mask as being considerate and caring of others in the classroom (not that you are sick), and realize that some may choose to mask to remain distanced*
- Be an **active** participant in your learning!  
You're welcome and **encouraged** to ask questions during class!
- If you feel **unwell**, or think you are, **please stay home**
  - *We will work with you!*
  - Get some rest 😊
  - View the recorded lectures – *please allow 24-48 hours to post*
  - *Contact us!*



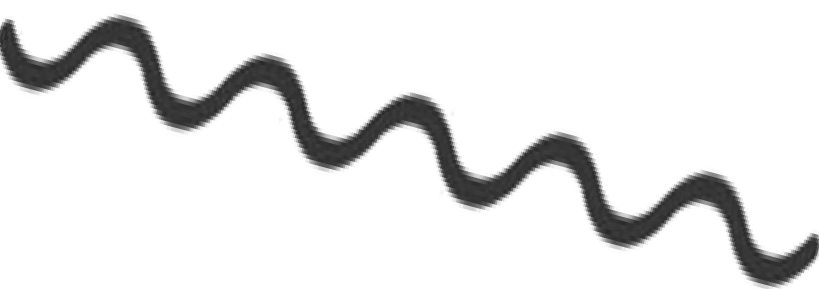
# Announcements



- **PA04** is due by 11:00pm on Wednesday (10/18)!
    - Submit on Gradescope: your .py file, and a PDF of your reflection
    - Please be mindful about submitting the right kind of files (.py file and .pdf file) as well as submitting the .py file that is named correctly (see assignment document for full details)
    - A note about submitting on Gradescope: you can submit an **UNLIMITED** number of times prior to the deadline. Look at the score you got, if you have some points taken off, that's ok, go back and fix your code and resubmit! Do this as often as you like BEFORE the assignment deadline. You cannot resubmit after the deadline.
    - **REMEMBER ALSO:** You have a grace period of 24 hours to submit your PAs!
  - **Quiz 5** will come out this week on Friday (3/15) and due Monday (3/18)
- 

# Exam 1



- **Exam 1** is *almost graded!*
  - Results will be on both Sherlock and Canvas grades
  - We want you to *learn* and *grow*!
    - Learn from your mistakes – opportunity to correct your errors to *gain up to 10 points!*
    - Announcement with details will be released
    - Speak to a TA or submit an .mp3 to the assignment on Canvas
    - **Deadline 12:00pm Monday, March 18, 2024**
- 

# Reminder: Python Strings: Basics

- Strings as collections:
  - Collection of characters
  - **Order** matters
  - Repetition (of characters) is ok
  - Each character is assigned a particular **index**, starting at index zero (0)

hello  
0 1 2 3 4

0	1	2	3	4	5	6
A	S	T	R	I	N	G

-7	-6	-5	-4	-3	-2	-1
A	S	T	R	I	N	G

X = "hello"  
X[i] gives the **i**th  
character in the  
string (starts at 0)

X[0] = first character  
= "h"

## Reminder: Indexing details

- (Using the example,  $s = \text{'wilson'}$ )
- Start counting at 0 (**positive** indices)
  - $s[0]$  is 'w'
    - Read as “s sub 0” or “s of 0”
    - “*The zeroth character of s*”
  - $s[1]$  is 'i'
  - $s[2]$  is 'l'
  - (Indices are always **integers**, if you tried  $s[1.0]$  it will give you an error)
- $\text{len}(s)$  gives the number of things in the collection
  - $\text{len}(s)$  is 6
- Use **negative** indices to count from the *end*.
  - $s[-1]$  is 'n'
  - $s[-2]$  is 'o'



Wilson the Volleyball  
From the movie **Cast Away** (2000)



## Reminder: More on String Indexing

- Consider the string “Bananas”

Positive Index	0	1	2	3	4	5	6
Letter	B	A	N	A	N	A	S
Negative Index	-7	-6	-5	-4	-3	-2	-1

- The table shows the positive and negative index of each letter
- The **largest** positive index of a string is the length of the string **\*\*minus 1\*\***
  - Example, in the above, the largest index of the string is **6**
- The **smallest** index is the negative value of the length
  - Example, in the above, the smallest index is **-7**

---

# PYTHON DEMONSTRATION

Let's jump on PyCharm!

Solution to: `strings_ica1.py`

Let's review the solution to the last in-class "lab" activity.



# Slicing

---

- **Indexing** is how to get one thing out of a collection
- **Slicing** is how to get a chunk out of a collection (multiple contiguous items)
  - Give it a **start** index and an **end** index
  - Evaluates to a new collection from **start** (*inclusive*) to **end** (*exclusive*)
- If S is a string, here's the syntax:

`S[start:stop:step]`

*Start position    End position    The increment*

- This returns the portion of the string from index **start** to index **stop**, at a step size **step**.
- **Fun trick** with slicing: **Reversing a string**: `my_string[::-1]` # full index range, backwards!

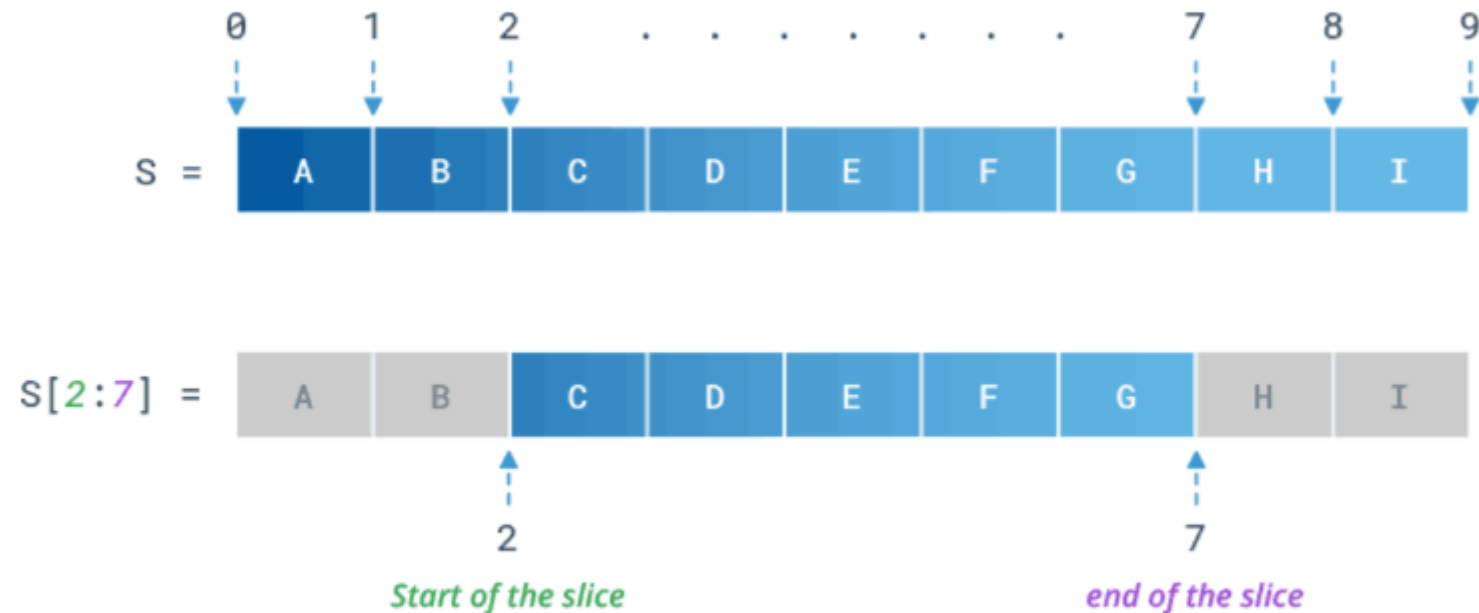
# String/List slicing `::` method

- `str1[start:stop]` # from **start** (incl.) to **stop** (excl.) (through stop-1)
- `str1[start:]` # from **start** (incl.) through the **end** of the collection
- `str1[:stop]` # from the **beginning** through to **stop** (excl.) (through stop-1)
- `str1[:]` # a **copy** of the whole array
- `str1[start:stop:step]` # from **start** (incl.) to **stop** (excl.) (through stop-1), keeping every **step<sup>th</sup>** item

# Slicing a String:

`s[2:7]` # CDEFG

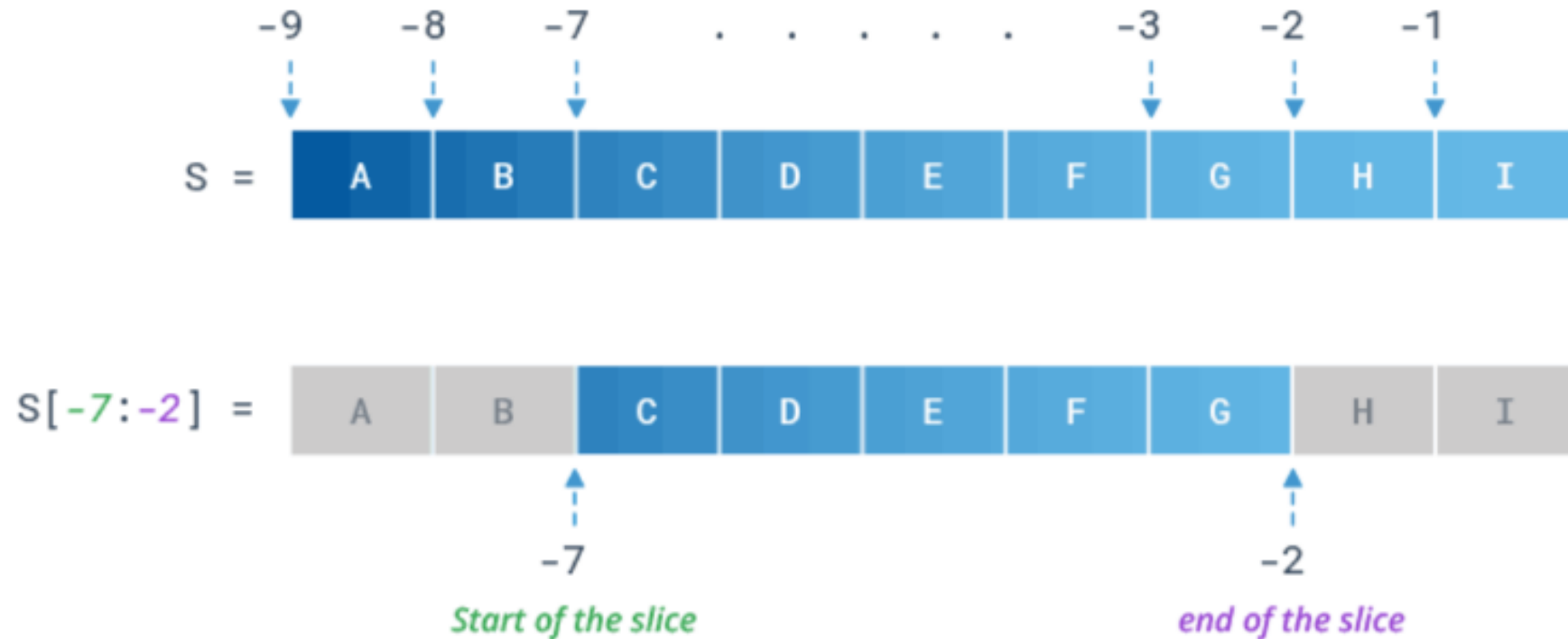
```
S = 'A B C D E F G H I'
print(S[2:7]) # C D E F G
```



**i** Note that the item at index 7 'H' is not included.

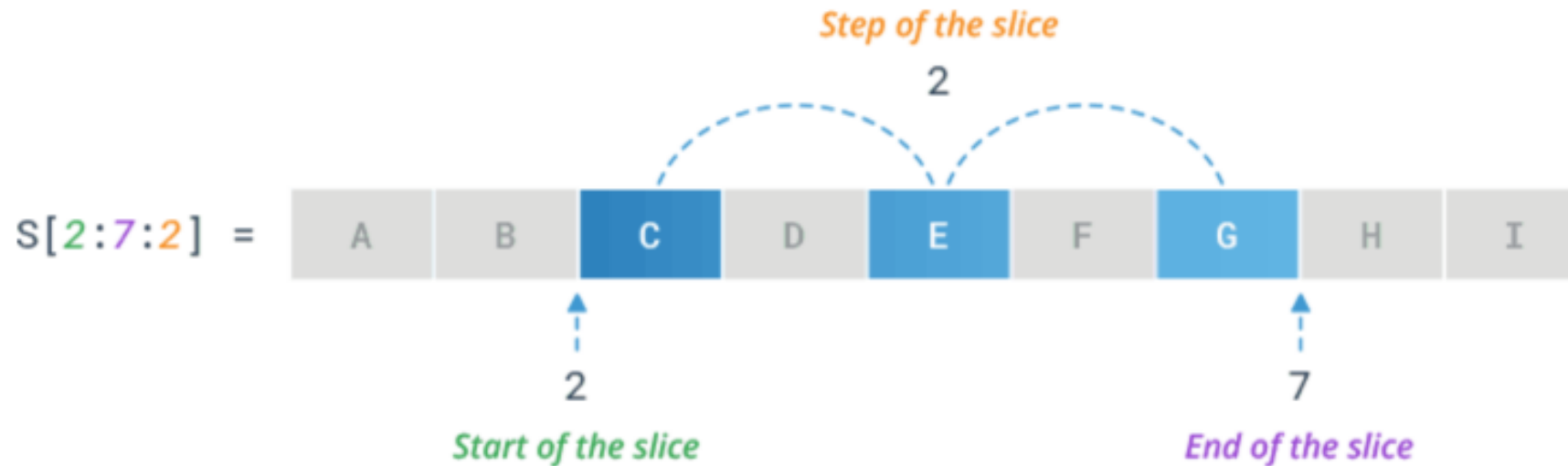
# Slice with Negative Indices

`s[-7:-2]` # CDEFG



# Specify Step of the Slicing

`s[2:7:2]` # CEG



```
# Return every 2nd item between position 2 to 7
S = 'ABCDEFGHI'
print(S[2:7:2]) # CEG
```

```
# Returns every 2nd item between position 6 to 1 in reverse order
S = 'ABCDEFGHI'
print(S[6:1:-2]) # GEC
```

# Slice at the Beginning and End

---

- **Omitting** the **start** index starts the slice from the index 0.
  - Meaning, **S[:stop]** is *equivalent* to S[0:stop]  
*# Slice first three characters from the string*  
S = 'ABCDEFGHI'  
print(S[:3])      **# ABC**
- Whereas, **omitting** the **stop** index extends the slice to the end of the string.
  - Meaning, **S[start:]** is *equivalent* to S[start:len(S)]  
*# Slice last three characters from the string*  
S = 'ABCDEFGHI'  
print(S[6:])      **# GHI**



# Slicing – Example 1

- Consider the string “Oranges”

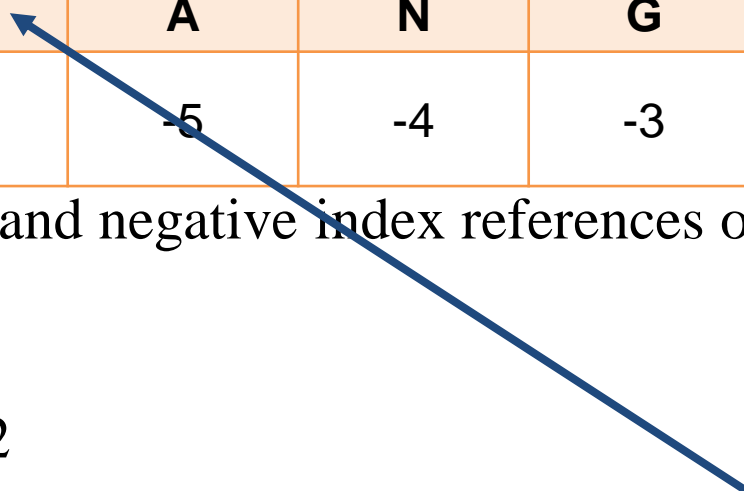
Positive Index	0	1	2	3	4	5	6
<b>Letter</b>	<b>O</b>	<b>R</b>	<b>A</b>	<b>N</b>	<b>G</b>	<b>E</b>	<b>S</b>
Negative Index	-7	-6	-5	-4	-3	-2	-1

- This table shows the positive and negative index references of each letter
- Example: `str[1:-2:2]`
- Start = 1; Stop = -2; Step = 2

# Slicing – Example 1

- Consider the string “Oranges”

Positive Index	0	1	2	3	4	5	6
Letter	O	R	A	N	G	E	S
Negative Index	-7	-6	-5	-4	-3	-2	-1

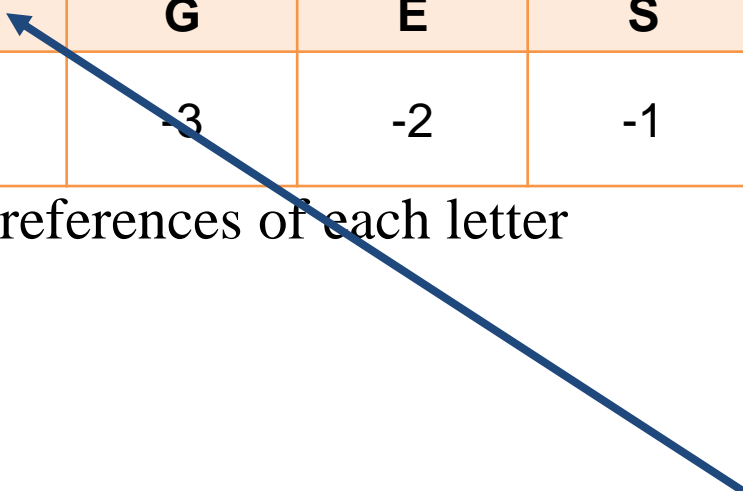


- This table shows the positive and negative index references of each letter
- Example: `str[1:-2:2]`
- Start = 1; Stop = -2; Step = 2
- Solution:** `r`
  - We start at index 1

# Slicing – Example 1

- Consider the string “Oranges”

Positive Index	0	1	2	3	4	5	6
Letter	O	R	A	N	G	E	S
Negative Index	-7	-6	-5	-4	-3	-2	-1

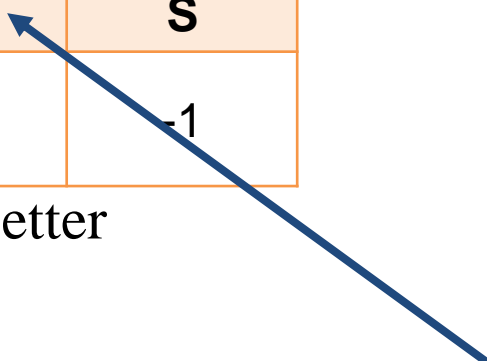


- This table shows the positive and negative index references of each letter
- Example: `str[1:-2:2]`
- Start = 1; Stop = -2; Step = 2
- Solution:** `rn`
  - We move by 2 (step is 2)

# Slicing – Example 1

- Consider the string “Oranges”

Positive Index	0	1	2	3	4	5	6
Letter	O	R	A	N	G	E	S
Negative Index	-7	-6	-5	-4	-3	-2	-1



- This table shows the positive and negative index references of each letter
- Example: `str[1:-2:2]`
- Start = 1; Stop = -2; Step = 2
- Solution:** `rn`
  - We move by 2 (step is 2), and then **stop** (we exclude the -2 index)

# What is printed?

---



```
my_string = "I love pizza"
```



```
print(my_string[2:-1:3])
```

# What is printed?

---



```
my_string = "I love pizza"
```



```
print(my_string[2:-1:3])
```

lei



# What is printed?

---



```
my_string = "I love pizza"
```



```
print(my_string[4:])
```

# What is printed?

---



```
my_string = "I love pizza"
```



```
print(my_string[4:])
```

```
ve pizza
```

# What is printed?

---



```
my_string = "I love pizza"
```



```
print(my_string[2:6])
```

# What is printed?

---



```
my_string = "I love pizza"
```



```
print(my_string[2:6])
```

love

# Slicing – Example 2

- Consider the string “I love pizza”

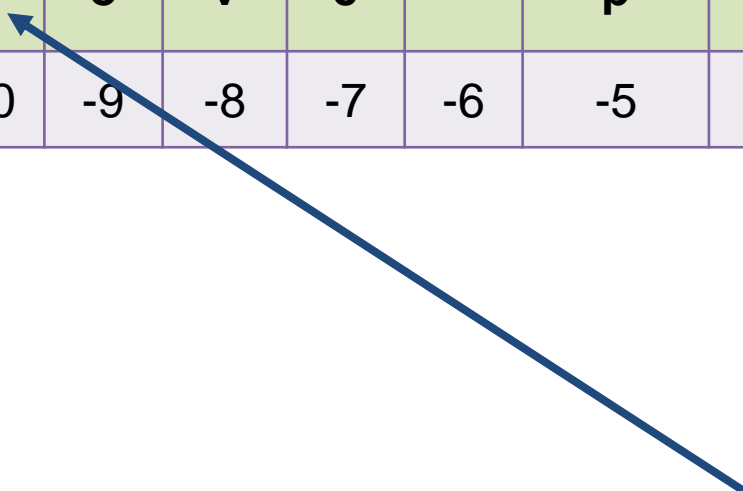
Positive Index	0	1	2	3	4	5	6	7	8	9	10	11
Letter	“I”	“ ”	“l”	“o”	“v”	“e”	“ ”	“p”	“i”	“z”	“z”	“a”
Negative Index	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

- Example: `str[2:-1:3]`
- Start = 1; Stop = -1; Step = 3

# Slicing – Example 2

- Consider the string “I love pizza”

Positive Index	0	1	2	3	4	5	6	7	8	9	10	11
Letter	“I”	“ ”	“l”	“o”	“v”	“e”	“ ”	“p”	“i”	“z”	“z”	“a”
Negative Index	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1



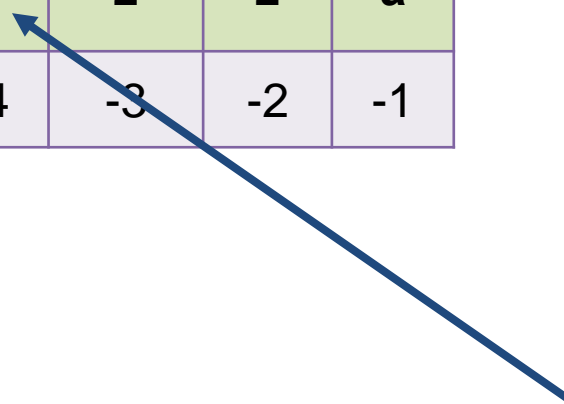
- Example: `str[2:-1:3]`
- Start = 1; Stop = -1; Step = 3
- Solution:** `l`
  - We start at index 2



# Slicing – Example 2

- Consider the string “I love pizza”

Positive Index	0	1	2	3	4	5	6	7	8	9	10	11
Letter	“I”	“ ”	“l”	“o”	“v”	“e”	“ ”	“p”	“i”	“z”	“z”	“a”
Negative Index	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1



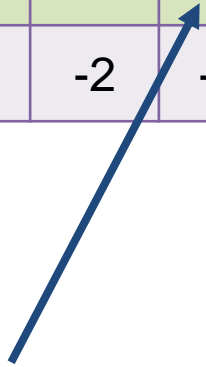
- Example: `str[2:-1:3]`
- Start = 1; Stop = -1; Step = 3
- Solution:** `lei`
  - We move by 3 (step is 3)

# Slicing – Example 2

- Consider the string “I love pizza”

Positive Index	0	1	2	3	4	5	6	7	8	9	10	11
Letter	“I”	“ ”	“l”	“o”	“v”	“e”	“ ”	“p”	“i”	“z”	“z”	“a”
Negative Index	-12	-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

- Example: `str[2:-1:3]`
- Start = 1; Stop = -1; Step = 3
- Solution:** `lei`
  - We move by 3 (step is 3)



out of bounds

# Slicing – Example 3 (with negative step)

- Consider the string “Oranges”

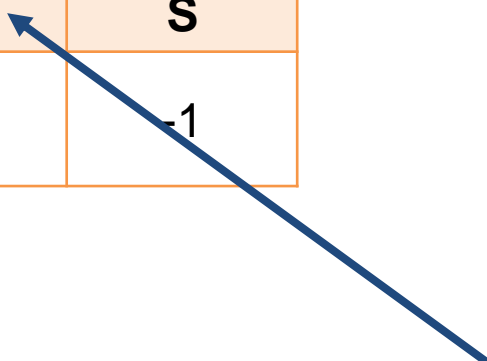
Positive Index	0	1	2	3	4	5	6
Letter	O	R	A	N	G	E	S
Negative Index	-7	-6	-5	-4	-3	-2	-1

- Example: `str[-2:1:-2]`
- Start = -2; Stop = 1; Step = -2

# Slicing – Example 3 (with negative step)

- Consider the string “Oranges”

Positive Index	0	1	2	3	4	5	6
Letter	O	R	A	N	G	E	S
Negative Index	-7	-6	-5	-4	-3	-2	-1

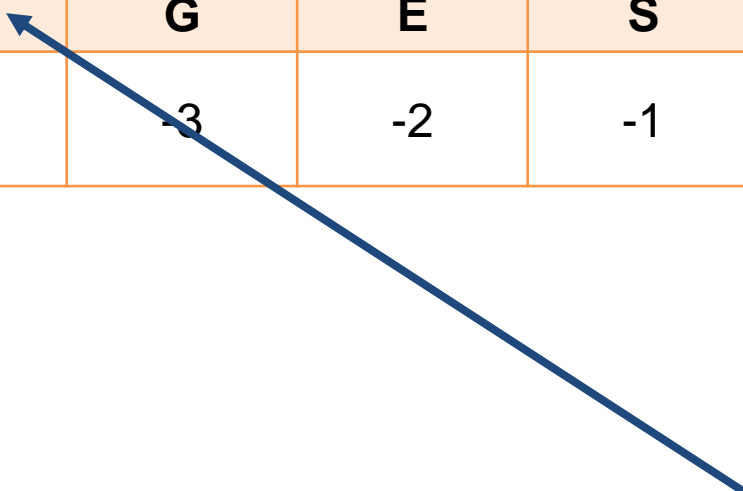


- Example: `str[-2:1:-2]`
- Start = -2; Stop = 1; Step = -2
- Solution:** `e`
  - We start at index -2

# Slicing – Example 3 (with negative step)

- Consider the string “Oranges”

Positive Index	0	1	2	3	4	5	6
Letter	O	R	A	N	G	E	S
Negative Index	-7	-6	-5	-4	-3	-2	-1

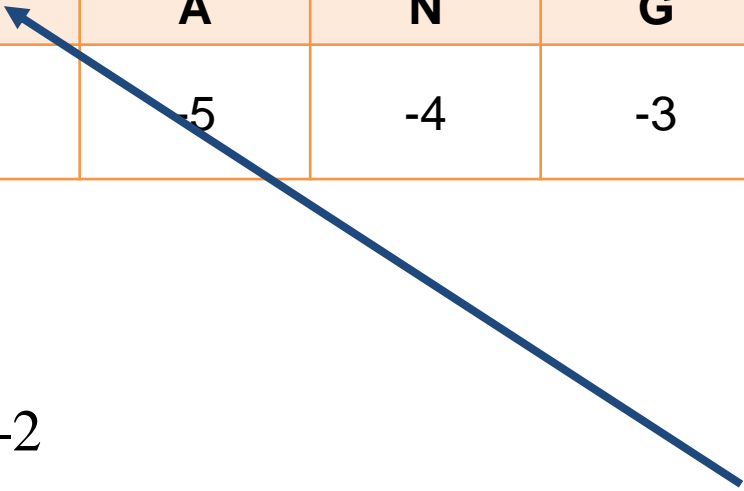


- Example: `str[-2:1:-2]`
- Start = -2; Stop = 1; Step = -2
- Solution:** `en`
  - We move back 2 (step is -2)

# Slicing – Example 3 (with negative step)

- Consider the string “Oranges”

Positive Index	0	1	2	3	4	5	6
Letter	O	R	A	N	G	E	S
Negative Index	-7	-6	-5	-4	-3	-2	-1



- Example: `str[-2:1:-2]`
- Start = -2; Stop = 1; Step = -2
- Solution:** `en`
  - We move by 2 (step is 2), and then **stop** (exclusive)



# Slicing – Example 4

- Consider the string “Oranges”

Positive Index	0	1	2	3	4	5	6
<b>Letter</b>	<b>O</b>	<b>R</b>	<b>A</b>	<b>N</b>	<b>G</b>	<b>E</b>	<b>S</b>
Negative Index	-7	-6	-5	-4	-3	-2	-1

- Example: `str[1:-2:2]` vs. `str[-2:1:-2]`
- Solution:** `rn` vs. `en`

# Slicing – Example 5

---

- Example of `str[start:end:step]`
- `my_str = "Python"`
- `print(my_str[1:5:2])`
- Output:     `yh`

# Slicing – Example 6

---

- Example of `str[start:end:step]`
- `my_str = "GEEKSFORGEEKS"`
- `print(my_str[-1:-12:-2])`
- Output: ???

# Slicing – Example 6

---

- Example of `str[start:end:step]`

• `my_str = "GEEKSFORGEEKS"`



• `print(my_str[-1:-12:-2])`

• Output: **SEGOSE**

# Slicing – Example 7

## Printing a String in Reverse Order

---

- Example of `str[start:end:step]`
- `my_str = "University of Virginia"`
- `print(my_str[::-1])` # Remember, this reverses the string!
- Output: `ainigriV fo ytisrevinU`



# Quick & Fun Survey Questions

Get to know your peers! 😊

**Books or Movies or Both?**

---

# PYTHON DEMONSTRATION

Let's jump on PyCharm!

`strings.py` - More string slicing examples!

# Activity for Today!

- In **pairs** or groups **up to three** work on the following activity.
- **string\_ica2.py**
- *Practice writing solutions that require string methods and string slicing*

Remember to **check-in** with a TA before leaving class today!

In-Class “lab” Activity!



# Reminder: CS Laptop Loaner Program

---

- This course requires students to have a **laptop**
- I realize that not everybody might have one (nor necessarily need one for their desired major / path...)
- If you do not have a laptop for any reason... *not to worry!*
- The CS department's Systems staff has a notebook / laptop loaner program and will be able to loan you a notebook / laptop computer for the duration of the semester if you don't have one or if you cannot afford one.
  - Also available if your laptop is broken and under repair, we can arrange for you to receive a loaner laptop for a week or two until your own laptop is fixed

---

Interested? Link: [https://www.cs.virginia.edu/wiki/doku.php?id=cs\\_laptop\\_loaner](https://www.cs.virginia.edu/wiki/doku.php?id=cs_laptop_loaner)

*I am happy to be your sponsor. Please let me know.*