Python

TomerBu

נושאים להיום:

Lists
Loops
Slicing and indexing
List Comprehension
Functions

נתונים 3 מספרים:

איך ניתן למצוא מי המספר הקטן ביותר?

```
x = int(input("Enter a number"))
y = int(input("Enter a number"))
z = int(input("Enter a number"))


smallest = min(x, y, z)
```

נתונים 3 מספרים:

איך ניתן למצוא מי המספר הקטן ביותר?

```
x = int(input("Enter a number"))
y = int(input("Enter a number"))
z = int(input("Enter a number"))

lst = [x, y, z]

smallest = lst[0]

for num in lst:
    if num < smallest:
        smallest = num

print(smallest)
```

נתונים 3 מספרים:

איך ניתן למצוא מי המספר הקטן ביותר?

```
x = int(input("Enter a number"))
y = int(input("Enter a number"))
z = int(input("Enter a number"))

lst = [x, y, z]

smallest = lst[0]

for num in lst:
    if num < smallest:
        smallest = num

print(smallest)
```

החלפה בין ערכים:

קלאסי:

```
x = 400
y = 200

if x > y:
    #swap the values
    #x = y # x = 200, y = 200
    temp = x # temp = 400
    x = y # x = 200
    y = temp # y = 400
```

temp = 400

x = 200

y = 400

החלפה בין ערכים:

פתרון בפייתון:

```
x = 400
y = 200

if x > y:
    #swap the values
    x, y = y, x
```

נתונים 3 מספרים:

איך ניתן למיין אותם:

בלי להשתמש בפונקציה המובנית **sorted**

1. עם משפטי תנאי
2. עם לולאה
3. מצאו איך להשתמש בפונקציה המובנית **sorted**

נתונים 3 מספרים:

איך ניתן למיין אותם:

```
x = 400
y = 200
z = 100

if x > y:
    #swap the values
    x, y = y, x
if y > z:
    #swap the values
    y, z = z, y
if x > y:
    #swap the values
    x, y = y, x
```

x = 200, y = 400, z = 100

x = 200, y = 100, z = 400

x = 100, y = 200, z = 400

נתונים 3 מספרים:

איך ניתן למיין אותם:

```
lst = [500, 200, 100, 500, 300, 100]

for i in range(len(lst) - 1):
    for j in range(len(lst) - 1):
        if lst[j] > lst[j+1]:
            lst[j], lst[j+1] = lst[j+1], lst[j]


print(lst)
```

נתונים 3 מספרים:

איך ניתן למיין אותם:

```
lst = [500, 200, 100, 500, 300, 100]
```

פונקציה שיוצרת לנו רשימה חדשה וממוינת:

```
print(sorted(lst))  # [100, 100, 200, 300, 500, 500]
```

```
lst = [500, 200, 100, 500, 300, 100]
```

פונקציה שממיינת את הרשימה המקורית:

```
lst.sort()

print(lst)  # [100, 100, 200, 300, 500, 500]
```

עוד על תנאים בפייתון:

משפט תנאי מקוצר בפייתון
(לא קיים האופרטור ?:)

פייתון שומרת על קריאות וקוד מובן:

```
# קלוט ציון – אם הציון גדול מ56
# "תדפיס "עברת את המבחן
# "אחרת תדפיס "נכשלת במבחן


grade = int(input("Enter your grade: "))
print("Success" if grade > 56 else "Failure")
```

עוד פעולות על רשימות:
שימוש בIntelisense

```
# re
lst.
lst.
lst.
     append
     reverse
     pop
     extend
     insert
# up remove
lst[ copy
     count
# re index
prin sort
prin clear
     __add__
lst.sort()
```

```
def sort(
    *,
    key: None = None,
    reverse: bool = False
) -> None: ...

def sort(
    *,
    key: (int) -> Support
    reverse: bool = False
) -> None: ...
```

Sort the list in ascending order and return None.

עבודה עם Slices

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

print(fruits[0])  # apple
print(fruits[-1])  # mango

print(fruits[-2])  # kiwi
```

אפשר לעבוד עם אינדקס שלילי:
האיברים מהסוף להתחלה

גישה מאוד נוחה לאיבר האחרון:
במקום
fruits(len(fruits) - 1)

עבודה עם Slices

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

print(fruits[1:3] )  #1(inclusive) to 3(exclusive)
```

התחלה באינדקס 1 כולל

סיום לא כולל ה3

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]


# start defaults to 0
print(fruits[:3])

# end defaults to len(list)
print(fruits[2:])  # 2 to end
```

ב"מ של start הוא 0

ב"מ של end הוא len(lst)

עבודה עם Slices

```
# all the elements but the last #['apple', 'banana', 'cherry', 'kiwi']
print(fruits[:-1])
```

כל הרשימה חוץ מהאיבר האחרון

מה עושה הקוד הבא?

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

fruits2 = fruits
fruits2.clear()
print(fruits)
```

2 מצביעים לאותו אובייקט
2 מצביעים לאותה כתובת בזכרון

למה ליצור עותק?
כך לא נגרום לתופעות לוואי
מי שמשתמש בפונקציה לא רוצה שתעלם לו הבננה!

כשיוצרים עותק
אפשר לבצע כל פעולה על העותק
מבלי לגרום לתופעות לוואי

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

fruits2 = fruits.copy()
fruits2.remove("banana")
print(fruits)
print(fruits2)
```

שימוש בSlices ליצירת עותק:

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

fruits2 = fruits[:]  # copy the list
```

slice[start:end:step]

הפרמטר השלישי נקרא step

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

# # [1, 3, 5, 7, 9]
print(numbers[::2]) # step 2
```

ברירת המחדל של step היא 1

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]


print(numbers[::-1]) # step -1
```

קפיצה שלילית:
ערך ב"מ של start הוא האיבר האחרון
ערך ב"מ של end הוא האיבר הראשון

אפשר לעבוד עם אינדקסים ומחרוזות:

```
phrase = "Monty Python!"
print(phrase[:-1]) #Monty Python

print(phrase[::-1]) #nohtyP ytnoM
```

איך נעתיק רק חלק מהאיברים לרשימה חדשה:

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

# copy all the fruits but the banana
fruits_copy = fruits[:1] + fruits[2:]

# copy all the fruits but the banana
fruits_copy = []
for fruit in fruits:
    if fruit != "banana":
        fruits_copy.append(fruit)
```

נתונה רשימה עם מספרים:
רוצים להעתיק את המספרים שגדולים מ56:

או שרוצים להעתיק רק את המספרים הזוגיים:

```
from random import randint
random_numbers = [randint(1, 100), randint(1, 100), randint(1, 100), randint(1, 100)]

even_numbers = []
for number in random_numbers:
    if number % 2 == 0:
        even_numbers.append(number)
```

List Comprehension

```
from random import randint
random_numbers = [randint(1, 100), randint(1, 100), randint(1, 100), randint(1, 100)]


even_numbers = [num for num in random_numbers if num % 2 == 0]
```

האופרטור in והאופרטור not in

```
#in ניתן לבדוק האם מחרוזת מכילה מחרוזת נתונה באמצעות האופרטור
#not in ניתן לבדוק האם מחרוזת מכילה מחרוזת נתונה באמצעות האופרטור


print("a" in "apple")  # True
print("b" in "apple")  # False

print("banana" not in "apple pie")  # True


# also works with lists:
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]
print("apple" in ["apple", "banana", "cherry"])  # True
print("apple" in fruits)  # True
```

מיני תרגיל:

נתונה הרשימה הבאה:

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]
```

יש ליצור רשימה חדשה עם כל הפירות שמכילים את האות a
יש להשתמש בlist comprehension

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

fruits_copy = [fruit for fruit in fruits if "a" in fruit]
print(fruits_copy)  # ['apple', 'banana', 'mango']
```

מיני תרגיל:

משחק איש תלוי:

אפשר להגדיר בנק מילים ולהגדיר מילה רנדומלית.

להציג למשתמש _____

```
from random import choice

fruits = ["apple", "banana", "cherry", "kiwi", "mango", "orange", "pear",
"pineapple", "strawberry", "watermelon"]

# random fruit:
random_word = choice(fruits)

underscores = ["_" for letter in random_word]
print(underscores)
```

```
underscores = ["_" for _ in random_word]
print(underscores)
```

כשלא משתמשים במשתנה
לא ניתן לו שם
כך יובהר למתכנתים בצוות שלא עשינו שימוש במשתנה

מיני תרגיל:

האופרטור כפל עובד גם על מחרוזות :)

```
print("🍏"*3) #🍏🍏🍏
```

```
print("🍏"*len(random_word))
```

המרת מחרוזת לרשימה של אותיות:

```
list("abcdefg") # ['a', b', 'c', 'd', 'e', 'f', 'g']
```

```
fruits = ["apple", "banana", "cherry", "kiwi", "mango"]

# random fruit:
random_word = choice(fruits)


apples = list("🍏"*len(random_word))
print(apples) #['🍏', '🍏', '🍏', '🍏', '🍏', '🍏', '🍏', '🍏', '🍏']
```