יחידה 4

תנאים

שלושה פרקים, 10 דקות. עורך טקסט, בלוקים, ו-if.

מפת היחידה

12 חלקים ▲

פרק	דקה	מה קורה שם
4.0	0:00	אינטרפרטר מול עורך טקסט
4.0	0:34	קובץ `.py` והרצה
4.1	0:00	מה זה בלוק
4.1	0:31	שורת פקודה ונקודתיים
4.1	**1:03**	**הזחה: ארבעה רווחים**
4.1	1:34	`IndentationError`
4.1	2:43	`pass`
4.2	0:32	`if` ובקרת זרימה
4.2	1:36	`else`
4.2	**1:57**	**`elif`**
4.2	3:07	למה לא תנאי בתוך תנאי
4.2	3:30	תנאי עם `and`

4.0 · עורך טקסט

עד עכשיו הקוד נכתב באינטרפרטר. זה טוב לשורה אחת ולבדיקה מהירה, אבל תוכנית של עשרות שורות כותבים ב**עורך טקסט** ושומרים כקובץ עם הסיומת `.py`.

	אינטרפרטר	עורך טקסט
למה	שורות קצרות, ניסוי	תוכנית שלמה
נשמר	לא	קובץ `.py`
פלט	מיד אחרי כל שורה	אחרי הרצה של כל הקובץ
ערך בלי `print`	מוצג	**לא מוצג**

השורה האחרונה בטבלה היא ההבדל שתופס הכי הרבה אנשים: באינטרפרטר `"hi".upper()` מציג `'HI'` . בקובץ, בלי `print` , לא יוצג כלום.

כל עורך עובד, גם פנקס רשימות. עורך מתאים לתכנות צובע את הקוד, משלים שמות ומסמן טעויות. ההרצה תלויה בעורך. בקורס זה כפתור run.

מכאן והלאה הדוגמאות בקורס מגיעות בשתי הצורות: קצרות באינטרפרטר, וארוכות בעורך.

4.1 · בלוקים

בלוק הוא קבוצת שורות ששייכות לאותו רצף לוגי. שני חוקים:

1. **השורה הראשונה מסתיימת בנקודתיים.** זו שורת הפקודה, כמו `if` .
2. **השורות של הבלוק מוזחות** ביחס אליה. הקורס, כמו PEP 8, משתמש ב**ארבעה רווחים**.

```
>>> finished_chapter = True
>>> learners = 10
>>> if finished_chapter:
...     print("Well done!")
...     learners = learners + 1
...
Well done!
>>> learners
11
```

שתי השורות המוזחות רצות רק אם התנאי נכון. ההזחה היא מה שאומר לפייתון איפה הבלוק נגמר. בשפות אחרות עושים את זה בסוגריים מסולסלים. בפייתון זה הרווחים.

הזחה היא תחביר

```
>>> debugging = "oops"
>>>   print(debugging)
Traceback (most recent call last):
```

```
  ...
IndentationError: unexpected indent
```

רווחים בתחילת שורה שלא אמורה להיות מוזחת הם **שגיאה**, ולא עניין של סגנון. ולא מערבבים רווחים וטאבים באותו קובץ.

pass

בלוק לא יכול להיות ריק. כשכותבים שלד של תוכנית ועוד לא יודעים מה ייכנס לבלוק, כותבים `pass` , והתוכנית רצה:

```
>>> grade = 75
>>> if grade > 90:
...     print("Good job")
... else:
...     pass  # TODO: what to say below 90
...
```

ההערה שאחרי `#` היא תזכורת לעצמכם. פייתון מתעלמת ממנה.

if, else, elif · 4.2

תנאי מאפשר להחליט אילו שורות ירוצו. זה נקרא **בקרת זרימה**.

```
>>> weather = "sunny"
>>> if weather == "rainy":
...     print("Take an umbrella")
... elif weather == "sunny":
...     print("Take a hat and sunglasses")
... else:
...     print("Take a light jacket")
...
Take a hat and sunglasses
```

	מתי רץ	חובה?
`if`	התנאי נכון	כן, תמיד ראשון
`elif`	כל מה שלפניו לא התקיים, והתנאי שלו כן	לא. כמה שרוצים
`else`	שום דבר לפניו לא התקיים	לא. רק אחד, ותמיד אחרון

פייתון בודקת לפי הסדר ועוצרת בראשון שמתקיים. אם גם `if` וגם `elif` נכונים, רק הבלוק של `if` ירוץ.

בחירה של המשתמש

```
choice = input("Pick a number: ")
if choice == "1":
    print("You chose 1")
elif choice == "2":
    print("You chose 2")
elif choice == "3":
    print("You chose 3")
else:
    print("You chose another number")
```

שימו לב ל- `"1"` במרכאות: `input` מחזיר מחרוזת, כמו ביחידה 2. השוואה ל- `1` בלי מרכאות לא תתקיים אף פעם.

למה elif ולא תנאי בתוך תנאי

אותה תוכנית בלי `elif` דורשת `if` בתוך `else` בתוך `else` , וכל רמה מוזחת עוד ארבעה רווחים. זה נקרא **קינון** (nesting), והקורס מגדיר אותו כמסורבל. `elif` שומר את כל האפשרויות באותה רמה.

תנאי מורכב

כל מה שלמדנו על `and` ו- `or` ביחידה 2 עובד גם כאן:

```
>>> months_since_last_eclipse = 18
>>> moon_position = 42
>>> sun_position = 42
>>> if ((months_since_last_eclipse == 18)
...         and (moon_position == sun_position)):
...     print("Solar eclipse!")
... else:
...     print("Just a regular day")
...
Solar eclipse!
```

מלכודת שהסרטון רק מזכיר: משתנה שמקבל ערך רק בתוך אחד הבלוקים לא קיים אם הבלוק הזה לא רץ. ניסיון להשתמש בו אחר כך נותן `NameError` . הפתרון הפשוט: לתת לו ערך לפני התנאי.

בדיקה עצמית

1. `x = 5` . בתנאי הראשון `x > 3` מדפיס `A` , וב- `elif x > 1` מדפיס `B` . מה יודפס?

2. מה חסר כאן: `if x > 3 print("big")` ?

3. אפשר לכתוב `else` בלי `if` לפניו?

4. למה `if input() == 5:` לא יתקיים אף פעם?

תשובות. 1: רק `A` . פייתון עוצרת בתנאי הראשון שמתקיים, גם אם `x > 1` נכון. 2: נקודתיים אחרי התנאי, ושורה חדשה מוזחת. 3: לא. `else` ו- `elif` באים תמיד אחרי `if` . 4: `input` מחזיר מחרוזת, ו- `"5"` אינו שווה ל- `5` .

סיכומים אישיים מתוך סרטוני הקורס Self.py בקמפוס IL, ולא חומר רשמי של הקורס.