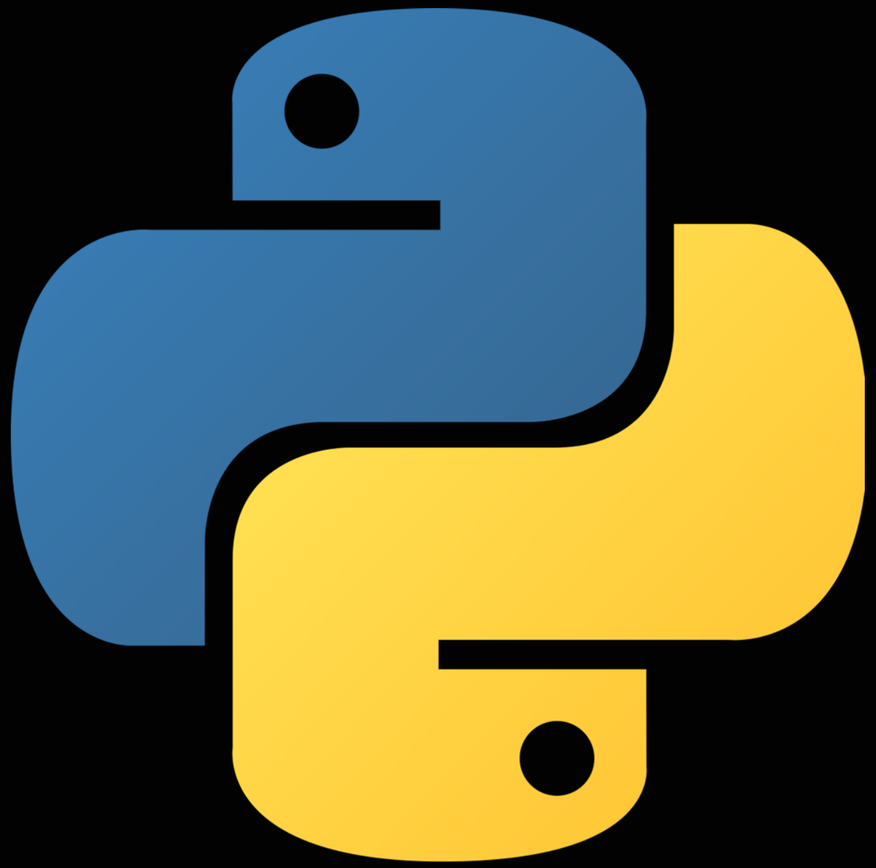


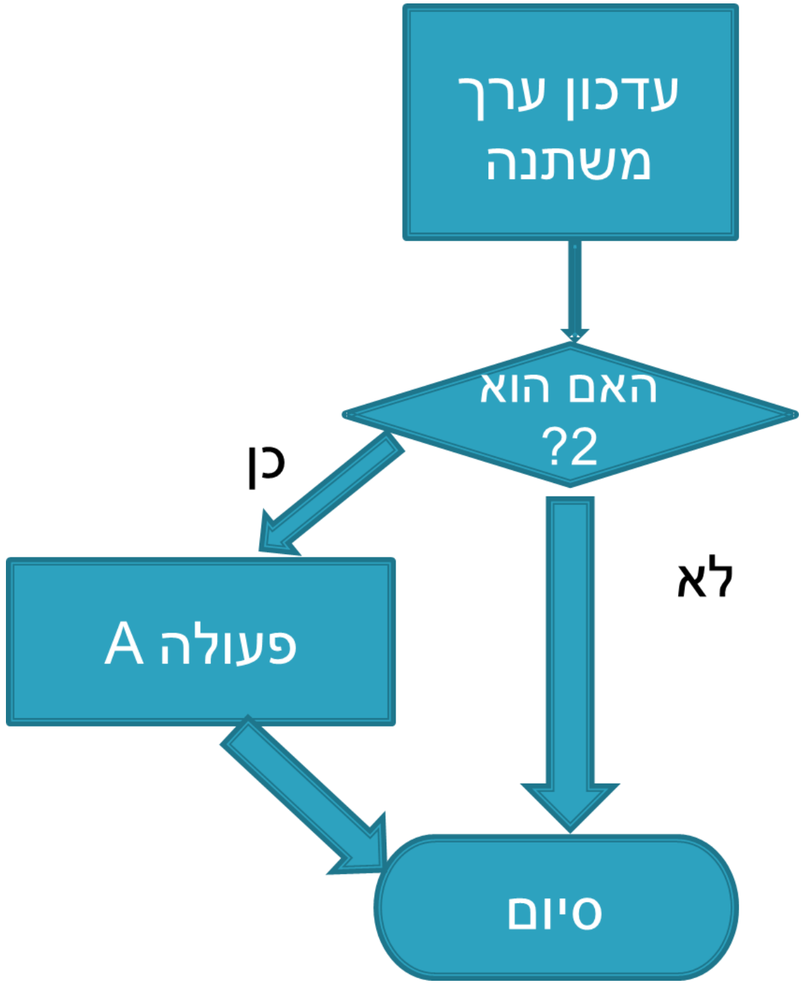
יסודות התכנות בשפת Python (20605; 20606)

רועי מימרן

2.1

משפטי תנאי בסיסיים

הסתעפות תכניות



משפטי if

- פקודת if שמבצעת החלטה על סמך ביטוי, שהערך שלו נכון או לא נכון, True או False:

- x = some_processing()
- if x==2:
- 	#Action set A – only if x was equal 2
- #Now do this part anyway
- # סיום

ביטויים לוגיים בפייתון

- במשפט if יבוא תמיד ביטוי שהערך שלו אמת או שקר, נכון או לא נכון, בפייתון זה נקרא True או False
- ביטויים כאלה בפייתון הם מטיפוס **bool**, במילים הם נקראים ביטוי לוגי או ביטוי בוליאני, על שם המתמטיקאי הבריטי ג'ורג' בול
- מבחינה מתמטית בפייתון ובשפות רבות נוספות, ל-True יש ערך 1, ול-False יש ערך 0

אופרטורים לוגיים של השוואה

- אופרטור השוויון (או השוואה): == לדוגמא: x==2 בודק אם x שווה ל-2
- אופרטור אי-שוויון =! יחזיר תמיד תוצאה הפוכה מהשוויון
- גדול מ: > בודק אם הביטוי משמאל גדול מהביטוי מימין, למשל: x>4
- קטן מ: > בודק אם הביטוי משמאל קטן מהביטוי מימין, למשל: x<4
- גדול או שווה: =< האם הביטוי משמאל גדול או שווה מהביטוי מימין: x>=4
- קטן או שווה: => האם הביטוי משמאל קטן או שווה מהביטוי מימין: x<=4
- לא תמיד אפשר לערוך השוואה בין טיפוסים שונים! למשל אי אפשר לבדוק אם מחרוזת גדולה יותר או קטנה יותר ממספר

דוגמא פשוטה של השוואה

- נכתוב תכנית שמקבלת ציון, וכותבת הערה של Excellent אם הציון גדול או שווה מ-90:

```
grade = int (input ("Please enter your grade: "))
if grade>=90:
    print ("Your grade is excellent!")
print ("Thank you")
```

אופרטורים לוגיים מול פעולות השמה

חשוב להבדיל בין פעולות השמה (שלמדנו למשל בהרצאה 1.5), שמציבות או מעדכנות ערך שמחושב בצד ימין לתוך משתנה שנמצא בצד שמאל, לבין פעולות ההשוואה שראינו עכשיו, שלא מציבות ערך במשתנה אלא רק בודקות אם תנאי מתקיים או לא ומחזירות בהתאם True או False.

בטבלה הבאה מפורטות כמה דוגמאות משני הסוגים:

פעולות השוואה – נכון או לא נכון	פעולות השמה – מציבות או מעדכנות ערך למשתנה
x == 4	x = 4
y != 2	x += 6
x < y	x -= 1
z >= 14	a *= 2
x+y <= 9	y -= 2

פעולות על ערכים לוגיים

- פעולת not לפני ביטוי בוליאני – תהפוך את הערך שלו, כלומר True יהפוך ל־False ולהיפך. למשל:

- if not x<4:

- התנאי יתבצע דווקא אם x לא קטן מ־4

- if not "x" in my_str:

- תחזיר True אם האות x לא נמצאת במחרוזת my_str

פעולות על ערכים לוגיים - המשך

- אם שמים את המילה and בין שני ביטויים לוגיים – המשמעות היא: **וגם**
- כלומר הביטוי יהיה True רק אם גם הביטוי מימין וגם הביטוי משמאל יהיו True
- דוגמא:

```
if x<2 and y>4:
```

- התנאי יתקיים אם x קטן משניים **וגם** y גדול מארבע
- שאלה למחשבה: אם x לא קטן משתיים, האם התכנית צריכה לבדוק את החלק השני, כלומר אם y גדול מארבע? האם יש טעם בביצוע הבדיקה הזו, האם היא תשפיע על המשך ריצת התכנית?
- תשובה: אין צורך ולכן הבדיקה לא תתבצע

פעולות על ערכים לוגיים - המשך

- אם שמים את המילה **or** בין שני ביטויים לוגיים – המשמעות היא: **או**. הביטוי שמתקבל יהיה True אם הביטוי הימני הוא True, או הביטוי השמאלי הוא True (או כמובן שניהם). למשל:

- if cash_a>0 or cash_b>0:
- print ("I still have cash to pay")

- ההודעה תצא אם cash_a גדול מאפס או cash_b גדול מאפס
- גם כאן שאלה: אם הביטוי מצד שמאל של מילת or הוא True, האם יש צורך לבדוק את נכונות הביטוי מצד ימין? ומה אם הביטוי מצד שמאל הוא False?
- אם הביטוי מצד שמאל של or הוא True, אז התוצאה היא בכל מקרה True

סדר הקדימות של אופרטורים לוגיים

- בפעולות חשבון כגון חיבור, חיסור, כפל וחזקה, למדנו את סדר הפעולות שקובע מה יתבצע קודם (כאשר אין סוגריים)
- גם באופרטורים בוליאניים יש סדר פעולות:
- not יתבצע ראשון
- and יתבצע לאחר מכן
- or יתבצע בסוף
- לדוגמא בביטוי: **x or y and not z,** קודם יתבצע not על z, לאחר מכן תבוצע פעולת ה־and עם y, ועל התוצאה יתבצע or עם x. אם נרצה סדר אחר – ניתן לשים סוגריים, למשל: **(x or y) and not z**

דוגמא – השוואת תאריכים

- נניח שקיבלנו שתי מחרוזות date_1, date_2 המבטאות תאריכים בפורמט המקובל בארץ, באותה שנה, למשל 24/05 ו־28/10 ואנו רוצים לבדוק אם התאריך הראשון או השני מאוחר יותר, בעזרת השוואת החודש והיום בנפרד

```
day1, month1 = int(date_1[0:2]), int(date_1[3:5])
day2, month2 = int(date_2[0:2]), int(date_2[3:5])
if month1<month2 or month1==month2 and day1<day2:
    # התאריך הראשון יוצא מוקדם יותר
    print ("Date 1 is earlier")
```

משפט if קצר בשורה אחת

- במשפט if, אם אפשר לכתוב בשורה אחת את מה שאנחנו רוצים לבצע במידה והתנאי יתקיים, אז אפשר לכתוב הכל בשורה אחת:

```
if a > b: print("a is greater than b")
```

- במקרה כזה לא מתבצעת הזחה כלל, השורה הבאה תתחיל באותה עמודה כמו ה־if, והיא תתבצע כמובן בלי קשר לנכונות או אי נכונות של התנאי ב־if
- במקרה כזה, יש לשקול אם התכנית נראית מספיק קריאה כשהתנאי והתוצאה רשומים באותה שורה – אם הם קצרים כמו כאן, זה יכול להיראות קריא. אם לא, כדאי בכל זאת לפצל לשתי שורות, כדי להימנע משורה ארוכה מדי