פרק 5 : מבנה בקרה

תוכן עניינים

5.1 משפטים ולולאות בפייתון	2
5.1.1 משפטי if , elif ו else	2
5.1.1.א הצהרות ותנאי if בפייתון	2
5.2. קשרים לוגיים and, or , not	5
5.3 לולאות while	6
5.4 לולאת for	8
5.5 איטרטורים -Iterators	10
5.6 פונקציית הטווח – range( )	10
5.7 המילה else בלולאות for	11
5.8 לולאות מקוננות	12
5.9 ההצהרה pass	13
5.10 תרגילים	14

בפרק 5 בתוכנית הלימודים להנדסאים של משרד החינוך במקצוע שפת פייתון רשום:

פרק 5 : מבנה בקרה

יעדים

התנסות בכתיבת אלגוריתמים בשפת Python תוך כדי שימוש במבני בקרה.

תכנים

1. **מבנה בקרה** – if , elif and else
2. **קשרים לוגיים** – and, or and not
3. **לולאת** while
4. **לולאת** for
5. **איטרטורים** – Iterators
6. **פקודות** range
7. **פקודת** pass

סיכום שעות ההוראה : 8 שעות עיוניות ו- 4 שעות התנסותיות. סה"כ 12 שעות.

5.1 משפטים ולולאות בפייתון

הערה: בהמשך הפרק נשתמש במושג **איטרציה** והכוונה היא לאובייקט המבצע פעולה החוזרת על עצמה או סדרה של פעולות החוזרות על עצמן על איברים של קבוצה (מערך, רשימות וכו').

שפת פייתון תומכת בתנאים הלוגיים הרגילים של מתמטיקה:

- Equals: a == b — האם a שווה ל b ?
- Not Equals: a != b — האם a לא שווה ל b ?
- Less than: a < b — האם a קטן מ b ?
- Less than or equal to: a <= b — האם a קטן או שווה ל b ?
- Greater than: a > b — האם a גדול מ b ?
- Greater than or equal to: a >= b — האם a גדול או שווה ל b ?

ניתן להשתמש בתנאים אלו במספר דרכים . לרוב זה עם משפטי if ולולאות.

5.1.1 משפטי if , elif ו else

5.1.1.א הצהרות ותנאי if בפייתון

הצהרת if נכתבת עם מילת המפתח **if** .

דוגמה :

```
a = 133
b = 210
```

```
if  b > a  :
  print("b is greater than a")          # יש לשים לב שהשורה הזו נרשמת עם הזחה יחסית לשורה שמעליה
```

בדוגמה זו אנו משתמשים בשני משתנים a ו- b המשמשים כחלק מ המשפט if כדי לבדוק אם b גדול מ a . a הוא 133 ו b הוא 210 אנחנו יודעים כי 210 הוא גדול מ 133 ולכן מדפיסים b is greater than a (b הוא גדול יותר מ a) .

5.1.1.ב הזחה indentation

שפת פייתון מסתמכת על הזחה (הזזה של השורה הנוכחית עם מקש הרווח spacebar בתחילת שורה) כדי להגדיר טווח/תחום בקוד. שפות תכנות אחרות משתמשות בסוגריים מסולסלים למטרה זו.

דוגמה : **שאלה** : מה היה קורה עם בתוכנית הקודמת לא היינו מבצעים הזחה בין השורה הרביעית לשלישית ?

```
a = 133
b = 200
if b > a:
print("b is greater than a")    # שימו לב  שאין הזחה יחסית לשורה הקודמת
```

תשובה : היינו מקבלים הודעת שגיאה של הזחה כמו שרואים בשורות הבאות:

```
  print("b is greater than a")
  ^
IndentationError: expected an indented block
```

מהו גודל ההזחה ? או במילים פשוטות כמה תווי רווח צריך בין השורה לזו שמעליה ? התשובה היא שמספיק רווח אחד. בסביבות העבודה של תוכנות העובדות עם פייתון ניתן לקבוע את כמות הרווחים. רווח של 4 תווים הוא מספיק נוח לעיניים.

5.1.1.ג מילת המפתח elif

מילת המפתח **elif** היא דרך בפייתון לומר "אם התנאים הקודמים לא היו True , אז נסה את התנאי הזה ".

דוגמה :

```
a = 120
b = 120
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
```

בדוגמה זו a שווה ל b, כך שהתנאי הראשון אינו נכון אך התנאי elif מתקיים ולכן אנו מדפיסים על המסך a and b are equal כלומר a ו- b הם שווים.

5.1.1.ד משפט else

מילת המפתח else (אחרת בעברית) מתאימה לכל דבר שלא היה True על ידי התנאים הקודמים.

דוגמה :

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print("a is greater than b")
```

בדוגמה זו a גדול מ- b ולכן התנאי הראשון אינו True (כלומר False) . גם מצב elif אינו True, ולכן עוברים למצב else ומדפיסים a is greater than b שאומר ש a גדול מ- b.

ניתן גם לרשום else ללא elif . **לדוגמה :**

```
a = 200
b = 33
if b > a:
   print("b is greater than a")
else:
  print("b is not greater than a")
```

5.1.1.ה משפט if מקוצר

אם יש רק משפט אחד לביצוע אז ניתן להציב אותו באותה שורה של המשפט if .

דוגמה: משפט מקוצר

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

5.1.1.ו משפט if else מקוצר

אם יש רק משפט אחד לביצוע, משפט אחד עבור if ואחד עבור משפט else אפשר לרשום את הכול באותה השורה.

טכניקה זו ידועה בשם מפעילי שלשות – Ternary Operators או ביטויים מותנים Conditional Expressions .

דוגמה : משפט if else מקוצר:

```
a = 12
b = 550
print("A") if a > b else print("B")
```

ההדפסה שנקבל : **B**

ניתן גם לכלול מספר הצהרות else באותה השורה.

דוגמה : מספר else באותה השורה :

```
a = 100
b = 100
print("A") if a > b else print("=") if a == b else print("B")
```

ההדפסה שנקבל : =

5.2. קשרים לוגיים and, or , not

5.2.1 משפט if עם and

מילת המפתח **and** (וגם) היא אופרטור לוגי ומשמשת לשילוב משפטים מותנים:

דוגמה : בדיקה האם a גדול מ b וגם (and) האם c גדול מ a .

```
a = 120
b = 57
c = 370
if a > b and c > a:
  print("Both conditions are True")
```

ההדפסה שנקבל : Both conditions are True

5.2.2 משפט if עם or

מילת המפתח **or** (או) היא אופרטור לוגי ומשמשת לשילוב משפטים מותנים:

דוגמה : בדיקה האם a גדול מ b או (or) האם a גדול מ c .

```
a = 200
b = 33
c = 500
if a > b or a > c:
  print("At least one of the conditions is True")
```

5.2.3 משפט if מקונן - Nested if

ניתן לכתוב משפט if בתוך if . זה נקרא הצהרות if מקוננות (מקונן - מהמילה קן של ציפור) .

דוגמה : נבדוק באיזה תחום ערכים נמצא x ביחס למספרים 10 ו 20 ?

```
x = 21
if x > 10:
  print("Above ten,")
  if x > 20:
    print("and also above 20!")
  else:
    print("but not above 20.")
```

ההדפסה שנקבל :

```
Above ten,
and also above 20!
```

5.2.4 ההצהרה pass

אסור להשאיר הצהרת if ריקה ללא משפט אחריה. אם בכל זאת מסיבה כלשהי (?) רוצים לרשום if ללא משפט אחרי ה if ניתן לרשום את המילה pass כדי לא לקבל הודעת שגיאה מהמפרש (interpreter) .

דוגמה :

```
a = 4
b = 5
if b > a:
  pass
```

5.3 לולאות while

בשפת פייתון יש 2 לולאות בסיסיות : **א. לולאת while ב. לולאת for .**

5.3.1 לולאת while

בלולאת while מבצעים קבוצה של משפטים כל עוד תנאי הוא True.

דוגמה : הדפסה של ערך המשתנה x כל עוד x קטן מ 5 :

```
x = 1
while x < 5:
  print(x)
  x += 1
```

ההדפסה שנקבל:

```
1
2
3
4
```

הערה : יש לזכור להגדיל את x ב 1 בסיום הלולאה כדי לא להיכנס ללולאה אין סופית.

5.3.2 משפט break

עם הצהרת **break** יכולים לעצור את הלולאה גם אם התנאי של הלולאה הוא True .

דוגמה : יציאה מהלולאה כאשר x הוא 3

```
x = 1
while x < 10:
  print(x)
  if x == 3:
    break
  x += 1
```

ההדפסה שנקבל :

```
 1
 2
 3
```

5.3.2 משפט continue

עם המשפט continue יכולים לעצור את האיטרציה (לולאה של פקודות) ולהמשיך עם הבאה .

דוגמה : מעבר לאיטרציה הבאה כאשר x שווה ל 3 .

```
x = 0
while x < 5:
  x += 1
  if x == 3:
    continue
  print(x)
```

ההדפסה שנקבל :

```
 1
 2
```

4

5

הסבר : כאשר x=3 לא מדפיסים אותו אלא ממשיכים לבצע את לולאת ה while .

5.3.3 משפט else

עם משפט else ניתן לבצע בלוק של קוד פעם אחת כאשר התנאי הוא כבר לא True .

דוגמה : הדפסה של הודעה כאשר התנאי הוא False

```
x = 1
while x < 5:
  print(x)
  x += 1
else:
    print( " x כבר לא גדול מ 5 " )
```

ההדפסה שנקבל : x כבר לא גדול מ 5

5.4 לולאת for

לולאת for משמשת לעבור על רצף של משפטים מספר רצוי של פעמים. רצף הפקודות יכול להיות ב list או tuple או set או dictionary או מחרוזת.

עם לולאת for ניתן לבצע קבוצה של משפטים ב list או tuple או set או dictionary או מחרוזת כל פעם עבור פריט אחר.

לולאת for איננה דורשת משתנה שישמש כאינדקס שאותו יש לאתחל עם ערך לפני כניסה ללולאה ולקדם אותו ב 1 בסיום איטרציה .

דוגמה : הדפסה של כל פריט ברשימה colors .

```
colors = ["red", "blue", "green"]
for x in colors:
   print(x)
```

ההדפסה שנקבל :

```
red
blue
green
```

5.4.1 לולאה במחרוזת

גם מחרוזות הן אובייקטים שניתן לבצע על התווים שבהן איטרציה .

דוגמה : הדפסה של תו אחרי תו במחרוזת green :

```
for x in "green":
  print(x)
```

ההדפסה שנקבל :

```
g
r
e
e
n
```

5.4.2 break ההצהרה

עם break ניתן לעצור (לשבור) ולסיים לולאה לפני שהיא מגיעה לסיומה.

דוגמה 1: הדפסה של הפריטים ברשימה colors ושבירת הלולאה כאשר מגיעים לפריט green

```
colors = ["red", "blue", "green", "white", "black"]
for x in colors:
  print(x)
  if x is "green" :
    break
```

ההדפסה שנקבל :

```
red
blue
green
```

דוגמה 2 : הדפסה של הפריטים ברשימה colors ושבירת הלולאה כאשר מגיעים לפריט green ללא הדפסה של green

```
colors = ["red", "blue", "green", "white", "black"]
for x in colors:
  if x is "green" :
    break
  print(x)
```

ההדפסה שנקבל :

```
red
blue
```

5.4.3 ההצהרה continue

עם ההצהרה continue עוצרים את האיטרציה הנוכחית וממשיכים עם האיטרציה הבאה.

דוגמה : נדפיס את כל הפריטים ברשימה colors אבל ללא green .

```
colors = ["red", "blue", "green", "white", "black"]
for x in colors:
  if x is "green" :
    continue
  print(x)
```

ההדפסה שנקבל :

```
red
blue
white
black
```

5.5 איטרטורים -Iterators

Iterator - איטרטור הוא אובייקט בעזרתו מבצעים מעבר בלולאה על איברים של קבוצה נתונה. הקבוצה יכולה להיות מערך, רשימה, tuple , מילון, set . כל מעבר בלולאה נקרא **איטרציה**.

איטרציה היא פעולה שחוזרת על עצמה במהלך פתרון של בעיה. בעברית – חזרור .

דוגמה:

```
for x in range (5) :
  הצהרה 1
  הצהרה 2
  הצהרה 3
```

במקרה הזה האיטרטור x מקבל ערך התחלתי של 0 . הערך הסופי שלו הוא 4 (לא כולל 5) . הלולאה מתבצעת 5 פעמים עבור הערכים 0 עד 4 . בכל פעם יש מעבר על הצהרות 1 עד 3 . המעבר על ההצהרות 1 עד 3 נקרא איטרציה.

במקרה הזה יש לנו 5 איטרציות.

5.6 פונקציית הטווח – range( )

בעזרת הפונקציה range( ) אנחנו עוברים בלולאה על קבוצת משפטים מספר רצוי של פעמים.

הפונקציה range( ) מחזירה רצף מספרים המתחיל ב 0 כברירת מחדל ומתקדמת ב 1 (ברירת המחדל) ומסתיימת במספר שצוין.

דוגמה 1: הדפסה של מספרים בטווח מ 0 עד 4 .

```
for x in range(5):
```

```
  print(x)
```

ההדפסה שנקבל :

```
0
1
2
3
4
```

כדאי לשים לב שההדפסה איננה כוללת את המספר 5 !

ברירת המחדל של ( )range היא 0 כהתחלה אבל ניתן לציין ערך התחלתי שונה כמו range(2,5) ואז נתחיל מ 2 .

דוגמה 2: הדפסה של מספרי מ 2 עד 5 (לא כולל 5) : for x in range(2,5)

ההדפסה שנקבל :

```
2
3
4
```

ברירת המחדל של הפונקציה ( )range היא הגדלה ב 1 של המשתנה . ניתן לקבוע את כמות ההגדלה על ידי ערך שלישי ב ( )range כמו range(2,7,2) ואז עוברים מהפריט השני עד השביעי (לא כולל השביעי) עם קידום של 2 .

דוגמה 3 : הדפסת המספרים מהאיבר השני עד השביעי לא כולל השביעי עם קפיצה על כל פריט שני.

```
for x in range(1,7,2) :
  print(x)
```

ההדפסה שנקבל :

```
1
3
5
```

5.7 המילה else בלולאות for

מילת המפתח else מציינת בלוק של הוראות שיש לבצע כאשר לולאת ה loop מסתיימת.

דוגמה 1 : יש להדפיס את המספרים בטווח מ 2 עד 15 עם קידום של 3 ובסיום להוציא הודעה שהמשימה בוצעה.

```
for x in range(2,15,3) :
  print(x)
else :
  print("המשימה בוצעה");
```

ההדפסה שנקבל :

2

5

8

11

14

המשימה בוצעה

אם בבלוק ה for מתבצע break אז בלוק ה else לא יתבצע !

דוגמה 2 : מילת break בלולאת for שגורמת לאי ביצוע של בלוק else .

```
for x in range(2,15,3) :
  print(x)
  if x==8 :
    break
else :
  print("המשימה בוצעה");
```

ההדפסה שנקבל :

2

5

8

5.8 לולאות מקוננות

לולאה מקוננת היא לולאה בתוך לולאה. "הלולאה הפנימית" תתבצע פעם אחת עבור כל איטרציה של "הלולאה החיצונית" .

דוגמה : הדפס את הרשימה numbers עבור כל צבע ברשימה colors

```
colors = ["red", "blue", "green"]
numbers= [1, 2, 3]
for x in colors:
  for y in numbers :
    print(x, y)
```

ההדפסה שנקבל :

```
  red 1
  red 2
  red 3
  blue 1
```

```
blue 2
blue 3
green 1
green 2
```

5.9 ההצהרה pass

לולאת for לא יכולה להיות ריקה (ללא משפטים לבצע). אם מסיבה כלשהי רושמים לולאת for ללא משפט/ים לביצוע יש לרשום pass כדי לא לקבל שגיאת מפרש (אינטרפרטר) .

דוגמה :

```
for x in [0, 1, 2]
    pass
```

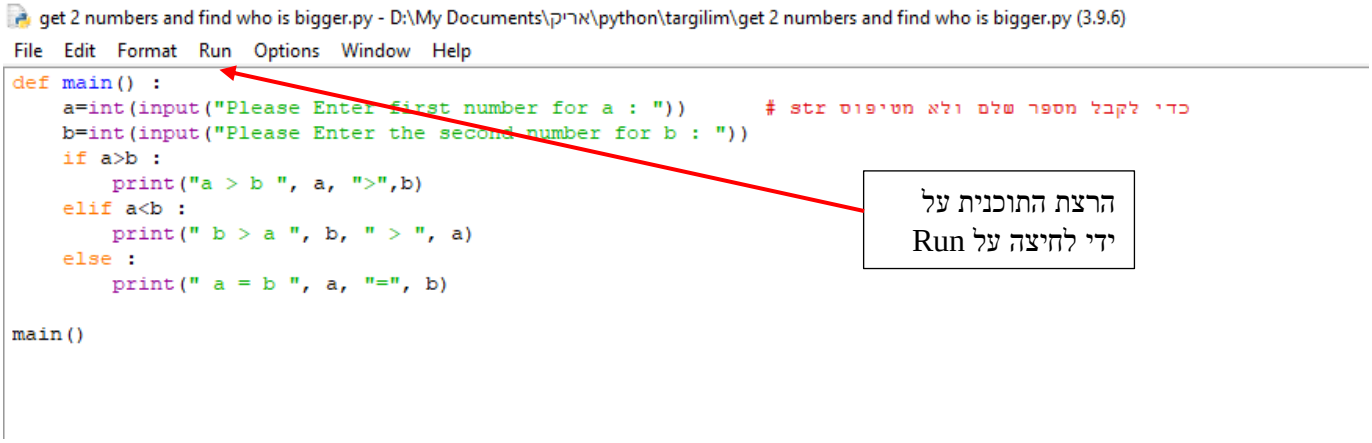
5.10 תרגילים

לפני שנתחיל לפתור את התרגילים, נזכיר כיצד מתבצע קלט מהמשתמש בפייתון. החל מפייתון 3.8 הקלט מתבצע בעזרת המתודה input( ) . אם משתמשים בגירסת פייתון 2.7 נשתמש במתודה raw_input .

כאשר מגיעים לפונקציה input( ) התוכנית נעצרת וממתינה שהמשתמש יקיש במקלדת ערך/ים . בסיום ההקשה המשתמש מקיש על Enter והתוכנית ממשיכה.

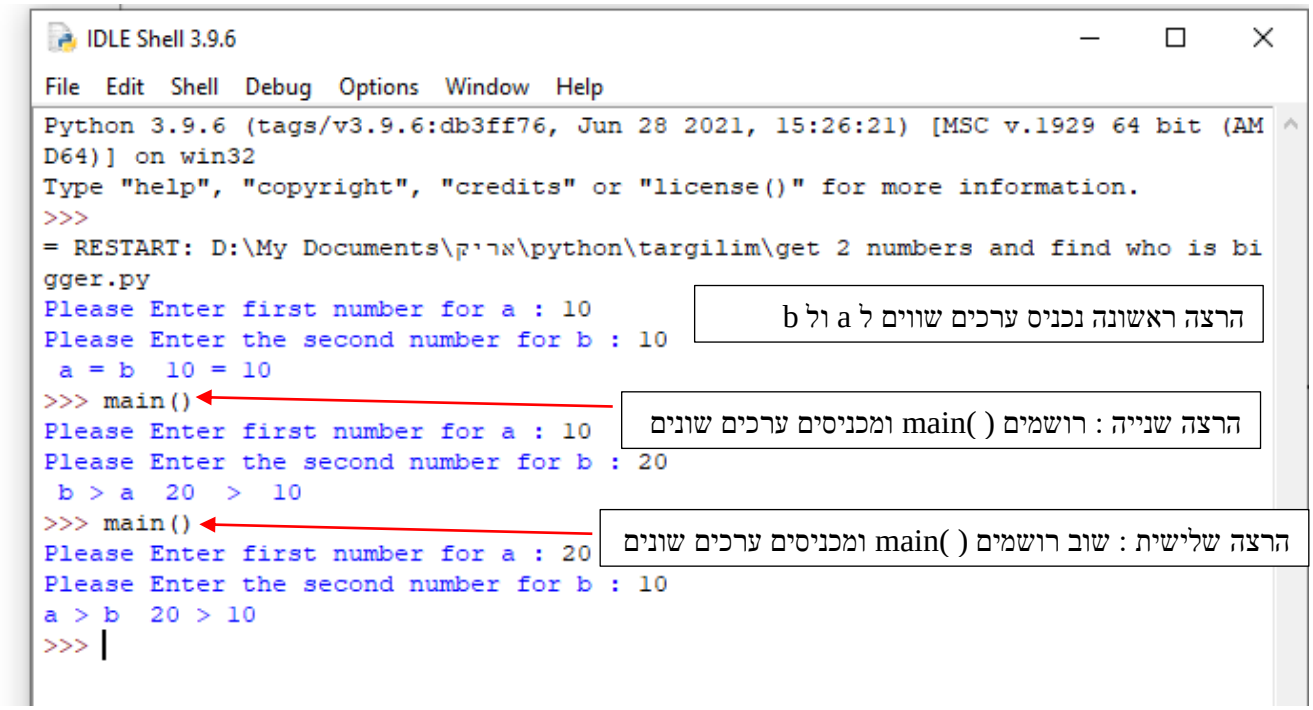
דוגמה : נבצע קלט של 2 מספרים ונדפיס מיהו הגדול מבין השניים . נרשום את התוכנית בתוכנה IDLE Python 3.8 64-bit ונשמור אותה עם File ו Save As כפי שרואים באיור הבא :

איור 1 : הרצת התוכנית בעזרת Run



איור 1 : התוכנית והרצה שלה בעזרת RUN .

כאשר נלחץ על Run נקבל את המסך הבא שהוא מסך ה Shell שעליו ירוצו התוכניות. נריץ את התוכנית כאשר פעם אחת נכניס את אותם ערכים ל a ו b ואחר כך נשנה את הערכים ונקבל :



איור 2 : הרצת התוכנית 3 פעמים עם נתונים שונים והתוצאות שנקבל.

הערה : בפקודת print התוכנית מדפיסה את מה שביקשנו ויורדת שורה. ניתן להוסיף לפקודת print את המילים ' end=' ואז ההדפסה איננה יורדת שורה אלא מדפיסה בהמשך ההדפסה הקודמת.

דוגמה לירידת שורה בסוף הדפסה:

```
>>> for n in range (3):
    print(n)


0
1
2
```

דוגמה : לא יורדים שורה בסוף הדפסה:

```
>>> for n in range (3):
    print(n, end=' ')


0 1 2
```

ניתן גם במקום ' end=' לרשום כל תו רצוי .

התרגילים

1. קלוט 3 מספרים המייצגים (בהתאמה) את האיבר הראשון של הסדרה החשבונית הנקראת באנגלית Arithmetic progression או Arithmatic sequence . ההפרש בין האיברים וכמות האיברים בסדרה. הדפס את כל האיברים ואת סכומם. ידוע ש : $a_n=a1+(n-1)*d$, $s_n=n*(a1+an)/2=n[2a1-(n-1)*d]/2$

2. קלוט 3 מספרים ממשיים המייצגים (בהתאמה) את האיבר הראשון של סדרה הנדסית (geometric series), היחס בין האיברים וכמות האיברים. הדפס את כל האיברים ואת סכומם. $A_n=a1*q^{n-1}$, $s_n=a1*(q^n-1)/(q-1)$. בסדרה אינסופית ($s_n=a1/(q-1)$. (תזכורת - חזקה רושמים **).

3. קלוט מספר שלם מהמשתמש המתאר סך של כסף מזומן . הדפס את האפשרות לכמות השטרות והמטבעות המינימאלית להמרת הסכום. לרשותך שטרות של 200, 100 , 50, 20 ומטבעות של 10 , 5 ו 1 ₪ .

4. כתוב תוכנית הקולטת 3 מספרים שלמים ומוציאה הודעה האם הם עוקבים ? (4,5,6 או 9,8,7)

5. קלוט 3 מספרים והדפס אותם לפי סדר עולה. א. עם אפשרות החלפה ביניהם. ב. ללא החלפה .

6. כתוב תוכנית המקבלת 3 צלעות של משולש ומוצאת מהו סוג המשולש : האם זה משולש ? שווה שוקיים ? שווה צלעות ? ישר זווית ? ישר זווית + שווה שוקיים (זכור : סכום שתי צלעות במשולש תמיד גדולה מהצלע השלישית)

7. קלוט שני מספרים ותו בקרה כמו + או - או * או / או % ובצע את הפעולה על שני המספרים לפי תו הבקרה.

8. רשום תוכנית שתקבל מספר המתאר שעות ותציג כמה דקות וכמה שניות במספר. לדוגמא המספר 10.5 יש בו 630 דקות (10.5* 60 ו 10.5*3600 שניות)

9. רשום תוכנית הקולטת ציוני תלמידים. כמות התלמידים איננה ידועה אך ידוע שכאשר מקישים מספר שלילי הסתיימה

הכנסת הציונים. התוכנית תדפיס את כמות הציונים ואת ממוצע הציונים.

10. קלוט 2 מספרים וחשב את מכפלתם ללא שימוש בפעולת כפל.

11. קלוט 2 מספרים וחשב את המנה והשארית ללא שימוש בפעולת חלוקה.

12. קלוט 20 מספרים ומצא מיהו המספר הקטן, מיהו הגדול וכמה פעמים כל אחד מהם נקלט.

13. רשום תוכנית שתקלוט מהמשתמש מספר ותודיע האם המספר הוא ראשוני. התוכנית מסתיימת כאשר המשתמש מקיש את המספר 0 . מספר ראשוני הוא מספר טבעי גדול מ 1 , כלומר מספר שלם וחיובי שמתחלק רק בעצמו ובמספר 1 ללא שארית. הוא לא יכול להיות מכפלה של שני מספרים טבעיים שקטנים ממנו !

14. הדפס את לוח הכפל מ 1 ועד 10 עם שורות ועמודות.

15. כתוב תוכנית הקולטת מספר שלם מהמשתמש ומוצאת את השורש השלם הקרוב ביותר למספר. א. עם פעולת חזקה 0.5** ב. עם לולאות

16. כתוב תוכנית הקולטת מספר ממשי ומוצאת את השורש שלו . א. עם פעולת חזקה 0.5** ב. עם לולאות

17. קלוט את המקדמים A,B,C של משואה ריבועית והדפס את שורשי המשוואה. הנוסחה : $x_{1,2} = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

18. הדפס את כל המספרים בני 3 ספרות שכל הספרות שלהן שונות אחת מהשנייה.

19. הדפס את כל המשולשים שהם ישרי זווית שצלעותיהם הם מספרים שלמים עד 50 .

20. קלוט מספר והדפס את כל המספרים הקטנים ממנו המתחלקים ב 3 והשארית שלהם היא 2 .

21. קלוט מספר עשרוני והצג את המספר ההפוך שלו. לדוגמא 123 ייתן 321 . התוכנית מסתיימת כאשר המשתמש מקיש 0 .

22. קלוט מספר שלם עשרוני ובדוק האם המספר הוא פלינדרום . פלינדרום הוא מספר (או מילה או משפט או רצף סמלים) שאם קוראים אותו מימין לשמאל או משמאל לימין הקריאה זהה. לדוגמה המספר 12321 או המילה לבלבל וכו' .

23. קלוט 2 מספרים (כל מספר בין 1 ל 10) והדפס מלבן של כוכביות * המספר הראשון כמות שורות והשני כמות עמודות). לדוגמא : אם קלטנו 2 ו 4 נקבל :

****

****

24. סדרת פיבונאצ'י (Fibonacci)היא סדרה של איברים ששני איבריה הראשונים הם 1, 0 וכל איבר לאחר מכן שווה לסכום שני קודמיו. איבריה הראשונים של הסדרה הם : 0 1 1 2 3 5 8 13 21 וכך הלאה. יש לרשום תוכנית שתדפיס את 20 האיברים הראשונים של הסדרה.

25. קלוט מספר מהמשתמש ורשום את כל האפשרויות של החלוקה שלו (ללא שארית) . התוכנית מסתיימת כאשר המשתמש מקיש על המספר 0 .

26. לרשום תוכנית שתקלוט מספר ותדפיס את העצרת שלו . עצרת היא מכפלת כל המספרים הטבעיים (מספר שלם חיובי) מ - 1 ועד למספר נתון. : n!=1*2*3*...*n

27. בחנות כלשהי רוצים לדעת את ההכנסות וההוצאות היומיות. כתוב תוכנית הקולטת בלולאה נתונים (בכל פעם נתון אחד). הנתון יכול להיות מספר או/ו תו. יש להתעלם מנתונים שאינם מספרים ולחשב ולהדפיס את כמות המספרים החיוביים שהוכנסו, כמות המספרים השליליים, את סכום המספרים החיוביים ואת סכום המספרים השליליים ואת הרווח היומי . התוכנית תסתיים כאשר יוקש הנתון END או end . הערות : 1. המתודה isdigit( ) מחזירה True אם כל התווים הם ספרות. 2. המספר 0 הוא לא חיובי ולא שלילי. 3. כשקולטים מספר שלילי המינוס הוא תו ויש להתגבר על זה...