משרד החינוך
המזכירות הפדגוגית – אגף מדעים

מחשבה הטכניון – מכון טכנולוגי לישראל
מרכז המורים הארצי למדעי המחשב

מינהלת מל״מ, המרכז הישראלי
לחינוך מדעי טכנולוגי ע״ש עמוס דה שליט

פיתוח חומרי עזר למורה להוראת מדעי המחשב בחטיבת הביניים - פייתון

פיתוח:

מורים מובילים לחטיבת הביניים

בהנחיית ד״ר דורון זוהר

פרויקט 11 פיתוח חומרי עזר למורה להוראת מדעי המחשב בחטיבת הביניים - פייתון

מרכז מורים ארצי במקצוע מדעי המחשב. הפרויקט מבוצע עפ"י מכרז 9/7.2013.
הפרויקט מבוצע עבור האגף לתכנון ופיתוח תוכניות לימודים, המזכירות הפדגוגית, משרד החינוך

יצא לאור במימון ובפיקוח המזכירות הפדגוגית, אגף מדעים במשרד החינוך
ומינהלת מל"מ, המרכז הישראלי לחינוך מדעי טכנולוגי ע"ש עמוס דה-שליט

© כל הזכויות שמורות למשרד החינוך

תוכן עניינים

אלגוריתמיקה באמצעות פייתון – למתחילים

כתיבה: עפר דיין , ויוי טרנר, עריכה: ד"ר דורון זוהר

פלט קלט ומשתנים 2

ביטויים לוגיים פשוטים וביצוע חוזר מותנה 15

ביצוע חוזר מותנה while 17

פונקציות 22

מבוא למחרוזות 27

ביצוע חוזר for על מחרוזות 35

הוראות תנאי ותנאים מורכבים 38

הצעות פתרון למורה 45

מבחן לכיתה ט מחצית א 66

הוראות פלט וסוגי פלט

לאחר הכרה ראשונית של סביבת העבודה נכתוב את התכנית הראשונה בסביבת העבודה.
בשיעור זה נכיר את נתמקד בסוגי פלטים שונים וכיצד ניתן להגדיר פלטים בצורה הרצויה שלנו.

מטרות השיעור

א. התלמידים יעמיקו את הכרתם עם סביבת העבודה של PyCharm
ב. התלמידים ילמדו את הוראות הפלט.
ג. התלמידים יתרגלו שימוש בהוראת פלט.

פלט– Output

באמצעות שימוש הוראת פלט ניתן להציג למשתמש מידע הקשור בביצוע האלגוריתם, כגון בקשת קלטים או הדפסת תוצאות חישוביות. הוראת פלט פשוטה, תדפיס שורה או שורות עוקבות של טקסט ומספרים, החל מהמקום בו נמצא סמן התכנית.

מבנה אלגוריתמי	מבנה ההוראה בשפת python
הדפס [רשימת פלט]	`print [item1][,item2][,item3]...`
הדפסת שורה ריקה	`print`

הצגת טקסט/מחרוזת תווים בין גרשים/גרשיים - יוצג על המסך התוכן כפי שהוגדר בין הגרש או הגרשיים.

הפלט שיוצג	הוראת הפלט
Hello, world!	`print "Hello, world!"`
good Day	`print 'good Day'`

פלט של ביטוי חשבוני/הדפסת תוצאת הביטוי

הפלט שיוצג	הוראת הפלט
8	`print 6+2`
8	`print 2+6`

רשימת פלט – רשימה של אברים מופרדים בפסיקים. האיברים יודפסו ברצף, מופרדים בתו רווח בין פלט לפלט. כל איבר יודפס בהתאם לסוגו, לדוגמה בהגדרת מחרוזת יוצג התוכן כפי שהוגדר בין גרש/ים, בהגדרת ביטוי חשבוני תוצג תוצאת החישוב.

הפלט שיוצג	הוראת הפלט
2+4= 6	`print '2+4=', 2 + 4`
(2+4)*3= 18	`print '(2+4)*3=',(2 + 4) * 3`

התו לוכסן \ - שילוב \ במחרוזת מאפשר פעולה יישום מיוחד לתו המופיע מיד אחריו :

- \n – מעבר לשורה חדשה (new line =).
- \' **או** \" – גורם לתו ' או " להיות כחלק מהמחרוזת. כלומר אין להם במצב זה משמעות של התחלה או סוף מחרוזת.
- \\ הדפסת התו לוכסן \ כחלק מהמחרוזת

הפלט שיוצג	הוראת הפלט
name: Jon family: Smith	`print 'name: Jon\nfamily: Smith'`

תרגילים - פלט

תרגיל ראשון

יש לכתוב תכנית בה יוצג כפלט כרטיס הביקור שלך. יש לעצב את כרטיס הביקור במסגרת כוכביות או תו עיצובי אחר על פי בחירתך. על הכרטיס לכלול את הפרטים הבאים:

- שם פרטי ושם משפחה
- תאריך יום ההולדת
- בית ספר וכיתה
- מקצוע מועדף שלכם

לדוגמה

```
**********************************
*  Name: My_Name Familly_Name    *
*  Birthday: 1/1/2000            *
*  School:Alon Grade:7 Class:5   *
*  Favorit subject: Computer Science *
**********************************
```

תרגיל שני

יש לכתוב תכנית בה מודפסים תרגילי רב ברירה בחשבון פשוט הבודק ידע בחישוב מספרים בני שני 2 ספרות. הבוחן יכלול 4 תרגילים ולכל תרגיל 4 אפשרויות תשובות אפשרויות שונות.

לפניך תכנית חלקית. יש להשלים את התכנית לקבלת הפתרון המבוקש:

```
print '+--------------------------------------+'
print '|Exam of 2 digits multiplication exercises |'
print '|Circle the correct answer              |'
print '|                   Good luck           |'
print '+--------------------------------------+'
```

<u>תרגיל שלישי</u>

יש לכתוב תכנית המדפיסה בוחן באנגלית הבודק ידע במילים נרדפות. על הבוחן לכלול 4 שאלות בחירה, ו-4 תשובות אפשריות לכל שאלה.

לפניך תכנית חלקית. יש להשלים את התכנית לקבלת הפתרון המבוקש:

```
print '+--------------------------------+'
print '|Exam of English Synonyms        |'
print '|Circle the correct answer       |'
print '|        Good luck               |'
print '+--------------------------------+'
```

קלט/פלט ומשתנים

לאחר שהכרנו את סביבת העבודה, המושג פלט ואת סוגי הפלטים שונים נגדיר משתנים, טיפוסיהם והקלטים האפשריים לכל סוג משתנה.

מטרות השיעור

א. התלמידים יכירו את המושג משתנה

ב. התלמידים יכירו מספר טיפוסי משתנים: הגדרתם ועדכון תוכנם.

ג. התלמידים יכירו את המושג קלט תוך שילוב שימוש במשתנים ובהוראות קלט/פלט.

ד. התלמידים יכירו את המושג "השמה" והוראת השמה.

ה. התלמידים יתרגלו שימוש בהוראות קלט/פלט והשמה.

משתנים–Variables

משתנה הוא תא בזיכרון המחשב, המשמש את התכנית המחשב לאחסן בו ערך. במהלך ביצוע התכנית, ניתן לשנות את הערך המאוחסן בו לערך אחר.

למשתנה מספר מאפיינים עיקריים:

- הגדרת **שם משתנה**

  שם המשתנה יהיה שם משמעותי בהתאם למידע המוכל בו, לדוגמה number, name

  שם המשתנה יכול להכיל אותיות לועזיות, ספרות וקו תחתון(_), לדוגמה

  first_name, number_of_students

  שם משתנה אינו יכול להתחיל בספרה.

  לא ניתן להגדיר מספר משתנים בעלי אותו שם בתוכנית.

  אותיות גדולות שונות מאותיות קטנות. לדוגמה, כל שמות המשתנים הבאים שונים זה מזה:

  abc, Abc,aBc, abC, ABc, AbC, aBC, ABC

- **ערך המשתנה**

  בכל שלב במהלך ביצוע התוכנית, המשתנה יכול להכיל ערך אחד בלבד. ערך המשתנה יכול להשתנות ע"י הוראת קלט או הוראת השמה. לאחר שינוי או עדכון של ערך המשתנה הערך הקודם נמחק ולא ניתן לשחזרו (אלא אם כן נשמר במשתנה אחר).

- **טיפוס המשתנה**

  נקבע ע"י הערך המוצב בו.

  הטיפוס משפיע על הפעולות שניתן לבצע על המשתנה. לדוגמה: אם ניתן ערך מספרי למשתנה ניתן לבצע חישובים על משתנה זה. למשתנה לו ניתן ערך מחרוזתי ניתן יהיה לשרשרו לקבלת מחרוזות ארוכות. במקרים מסוימים, ניתן לשנות טיפוס של משתנה ע"י ביצוע פונקציית המרה במהלך התוכנית.

הגדרת משתנים:

טיפוס משתנה מוגדר עם קבלת ערכו הראשוני (קלט או השמה):

הגדרה	טיפוס	הסבר
`a = 5`	Integer	המשתנה a הוגדר כמשתנה שלם עם השמת המספר 5
`num = 6.2`	Float	המשתנה num הוגדר כמשתנה עשרוני עם השמת המספר הממשי 6.2
`name = 'Yael'`	String	המשתנה name הוגדר כמחרוזת עם השמת השם/מחרוזת Yael
`flag = False`	Boolean	המשתנה flag הוגדר כמשתנה בוליאני עם השמת הערך False (שקר)

לאחר הגדרת משתנה ניתן:

- להדפיסו
- לשנות את ערכו ע"י השמת/הצבת ערך חדש (מאותו סוג) או לשנות את טיפוסו ע"י השמת ערך מטיפוס שונה ובכך להמיר את סוגו (בטבלה למטה אפשרויות נוספות להמרות משתנים).
- לקלוט אליו ערך חדש.

משתנה יכול לקבל ערך קבוע, ערך של משתנה אחר, או ביטוי כלשהו, שמורכב משילוב קבועים ומשתנים.

פלט התכנית	קטע תכנית לדוגמה	סוג הפעולה

9
9
	`number = 9`
	`print number`
	`num2 = number`
	`print num2`
		הוראות השמה

number is: 8.1
number was: 5
	`number = 5`
	`num2 = 'empty'`
	`num2 = number`
	`number = 8.1`
	`print 'number is:', number`
	`print 'number was:', num2`
		שינוי ערך משתנה, ושינוי טיפוס

<type 'int'>
<type 'float'>
<type 'str'>
<type 'bool'>
	`var_1 = 3`
	`var_2 = 5.6`
	`var_3 = 'wow'`
	`var_4 = True`
	`print type(var_1)`
	`print type(var_2)`
	`print type(var_3)`
	`print type(var_4)`
		שימוש בפונקציה type() להדפסת סוג המשתנה

פלט התכנית	קטע תכנית לדוגמה	סוג הפעולה
<type 'str'> 6 <type 'int'> 7 <type 'float'> 25.0	var_1 = str(2 * 3) print type(var_1), var_1 var_2 = int(7.25) print type(var_2), var_2 var_3 = float('25') print type(var_3), var_3	המרת טיפוסים באמצעות פונקציות: str() - למחרוזת int () - לשלם float() - לעשרוני

קלט– Input

הוראת קלט מאפשרת לתכנית לפעול באופן מקוון (אינטראקטיבי) מול המשתמש.

ניתן לספק לתכנית נתון/ים באמצעות המקלדת (או כל אמצעי קלט אחר כגון מסך מגע), והיא תגיב בהתאם לאלגוריתם המובנה בתוכה.

לפני ביצוע הקלט יש להגדיר פלט המנחה את המשתמש להכנסת הקלט המבוקש.

את הקלט יש לאחסן במשתנה.

לאחר לחיצה על מקש ה-Enter/אישור, הקלט מאוחסן במשתנה כ**מחרוזת**.

מבנה ההוראה בשפת python	מבנה אלגוריתמי
str_var = raw_input('text')	**משתנה** = raw_input('הנחיה מחרוזת למשתמש')

דוגמאות:

בקשת שם, המשתמש כותב John, והתכנית בתגובה מדפיסה את השם הנקלט:

פלט וקלט התכנית	קטע תכנית לדוגמה
Enter name: John Who is John ?	name= raw_input('Enter name: ') print 'Who is', name,'?'

בקשת מספר, המרת הטקסט למספר ממשי והדפסתו כמספר שלם ע"י זימון הפונקציה int.

פלט וקלט התכנית	קטע תכנית לדוגמה
Enter number:3.14 The integer part is 3	num= float(raw_input('Enter number: ')) print 'The integer part is', int(num)

תרגילים - קלט

תרגיל ראשון

יש לכתוב תכנית הקולטת שמך, כיתתך ושם בית הספר בו את/ה לומדים.

על התכנית יש להגדיר את המשתנים הבאים:

my_name ,my_class, my_school

בהתאם לקלט על התכנית להדפיס את הפלט הבא:

Good morning <your name> **of class** <your class> **from** <your school> **school**

תרגיל שני

יש לכתוב תכנית הקולטת את שנת לידתך במספר שלם למשתנה .year_of_birth בנוסף יש להגדיר משתנה `current_year` המקבל כהשמה את השנה הנוכחית. על התכנית לחשב ולהדפיס את גילך.

המשך ל**תרגיל שני -** אתגר:

באתר stackoverflow נשאלה השאלה הבאה:

[How to get current time in python and break up into year, month, day, hour, minute?](#)

התשובה שניתנה:

```
import datetime
now = datetime.datetime.now()
print now.year, now.month, now.day, now.hour, now.minute, now.second
```

נסו לשנות את שנת ההשמה (בדוגמה שלנו 2017) לשנה המתעדכנת אוטומטית עם הפעלת התכנית ולאחר קליטת שנת הלידה של המשתמש/ת.

פעולות חשבוניות עם משתנים

לאחר שהכרנו את משמעות המושגים קלט, פלט, משתנה והשימוש במשתנים, נכיר את הפעולות החשבוניות השונות שניתן לבצע על ערכים ומשתנים.

נתרגל פעולות חשבוניות עם משתנים תוך שילוב הוראות קלט, פלט והשמה.

מטרות השיעור

א. התלמידים יכירו סוגי פעולות החשבון השונות.

ב. התלמידים יתרגלו פעולות חשבון שונות בטיפוסי המשתנים השונים (+, - , *, / ,**, %), הגדרה ועדכון ערכי המשתנה, המשמעויות של טיפוסי המשתנים והפעולות השונות עליהם.

ג. התלמידים יתרגלו שימוש בהוראות קלט/פלט והשמה.

ד. התלמידים יכירו את פעולת חילוק מספרים שלמים וחילוק שארית.

ה. התלמידים יפתחו אלגוריתם ויכתבו תכניות שעיקרן יישום והבנה של משתנים, פעולות עם משתנים, הוראות קלט/פלט הקשורות לנושאי תכנית הלימודים מבוא לאינטרנט וסייבר.

פעולות חשבון על מספרים ב-Python

סימן	תיאור הפעולה	הסבר
+	חיבור	
-	חיסור	
*	כפל	
**	חזקה	למשל, 2 בחזקת 3, 2**3=8
/	חילוק	אם המחולק הוא מספר שלם תוצאת החלוקה תהיה מספר שלם, למשל 4 / 2 = 2 5 / 2 = 2
//	חילוק לקבלת המנה	תוחזר תוצאה כמספר שלם בהתאם לטיפוס, למשל 9 // 2 = 4 9.0 // 2 = 4.0
%	שארית החלוקה	תוצאת הפעולה היא משמעותית כאשר היא מתבצעת על מספר שלם והיא השארית שלאחר החלוקה, למשל 5 % 2 = 1 11 % 3 = 2 2 % 4 = 0

לפרטים נוספים יש להיכנס לאתר/קישור Python Basic Operators

<u>תרגילים – פעולות חשבון עם מספרים</u>

<u>תרגיל ראשון</u>

יש לכתוב תכנית ה הקולטת את מסה בק״ג של גוף, את גובהו במ׳ מעל לקרקע. על התכנית לחשב ולהדפיס אנרגית הגובה של גוף זה (E=mgh). יש לצרף לתוצאה את יחידת המידה ג׳ול (J).

<u>תרגיל שני</u>

יש לכתוב תכנית הקולטת רדיוס מעגל. על התכנית לחשב ולהדפיס את היקף המעגל (2πr) ושטחו (πr2).

<u>תרגיל שלישי</u> (למידת חקר)

לפניך מספר קטעי קוד. כל קוד מכיל פקודות השמה ופקודות פלט.

פתרו את התרגיל הבא על פי השלבים הבאים :

א. לכל אחד מהקטעי הקוד יש להגדיר את הפלט שלדעתכם יתקבל (עמודת מה הפלט הצפוי).

ב. העתיקו את קטעי הקוד ורישמו את הפלט המתקבל (עמודת מה הפלט שיתקבל).

ג. בתחתית כל טבלה הגדירו כלל לקבלת הפלטים השונים.

קטע קוד	פלט צפוי?	הפלט שהתקבל?
a = 10 print a / 3 a = 10.0 print a / 3 a = 10 print a / 3.0 a = 10.0 print a / 3.0		
הכלל :		

קטע קוד	פלט צפוי?	הפלט שהתקבל?
a = 10 **b = 'WoW'** print **a + a** print **b + b**		
הכלל :		

קטע קוד	פלט צפוי?	הפלט שהתקבל
a = 10		

b = 10.0 c = ’big’ print a * 3 print b * 3 print c * 3		
הכלל:		

קטע קוד	פלט צפוי?	הפלט שהתקבל
a = 'big' **print a * 0**		
הכלל:		

קטע קוד	פלט צפוי?	הפלט שהתקבל
a = 'big' **b = 'small'** **print a * True + b * False** **a = 5** **b = 6** **print a * True + b * False** **print True + 1** **print False + 1**		
הכלל:		

<u>תרגיל רביעי</u>

לפניך שני קטעי קוד. יש להריץ כל קטע ולהשלים את הפלט המתקבל.

קטע קוד	הפלט שהתקבל
name = 'Rachel' **age = 16** **print '%s %d' % (name, number)** **name = 'Rachel'**	

הפלט שהתקבל	קטע קוד

	`print "%s is my best friend!"%(name)`
	
	`diff = 15`
	`print "My older brother is %d years older.”%(diff)`

<u>תרגיל חמישי</u> (אתגר)

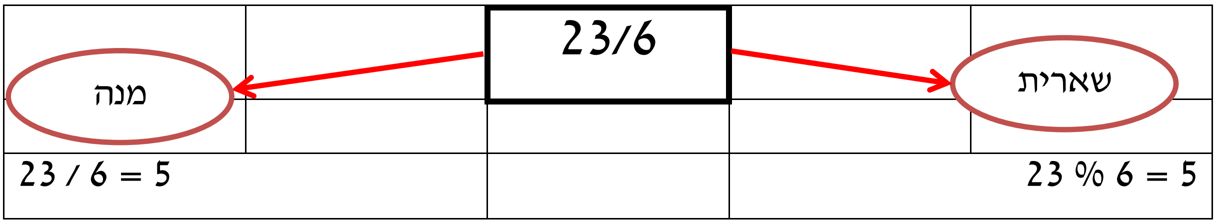
יש לכתוב תכנית בת שתי שורות הקולטת מספר שלם. על התכנית להדפיס even אם נקלט מספר זוגי ו- odd אם נקלט מספר אי זוגי.

שימו לב – יש לפתור את התרגיל ללא שימוש בהוראת אם (IF).

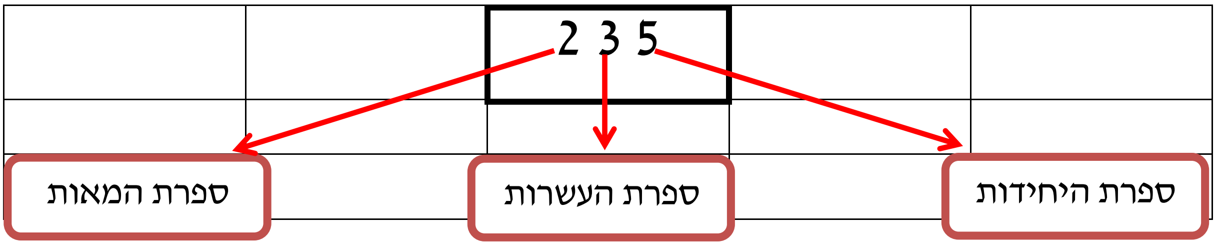
<u>שארית החלוקה במספרים שלמים</u>

מנת החלוקה - כאשר אנו מחלקים מספרים שלמים תוצאת החלוקה היא מספר שלם.

בחלוקה זו תינתן שארית, אם נרצה לחלק את המספר השלם 23 במספר השלם 6



פרוק מספר בן שלוש ספרות:



<u>בחלוקת המספר ב-10 נקבל:</u>

מנה, tens : 23

שארית, units : 5

<u>אם נחלק את המנה ב-10 נקבל:</u>

מנה : 2

שארית : 3

```
num = 235
tens = num / 10
units = num % 10

num = tens
tens = num % 10
num = num / 10
print num ,tens ,units
```

<u>תרגילים – פרוק מספר לספרות</u>

<u>תרגיל ראשון</u>

יש לכתוב תכנית הקולטת מספר שלם דו ספרתי. על התכנית להדפיס את סכום ספרותיו ומכפלתם.

<u>תרגיל שני</u>

יש לכתוב תכנית הקולטת מספר שלם תלת ספרתי. על התכנית להדפיס את מספר שספרותיו בסדר הפוך.

<u>תרגיל שלישי</u>

לאור ניסיונכם בשפות תכנות אחרות כגון SCRATCH או JAVASCRIPT תרגמו את האלגוריתם מילוליים הבאים. הגדירו תיעוד מתאים בראש כל פעולה ותארו את מטרת הפעולה.

1. קלוט מספר שלם למשתנה num
2. כל עוד num > 0
   2.1 הדפס num % 10
   2.2 num ← num / 10

פתרון :

1. הצב 0 במשתנה res
2. קלוט מספר טבעי (שלם חיובי) למשתנה num
3. כל עוד num > 0
   3.1 num ← res + num % 10
   3.2 num ← num // 10
4. הדפס res

פתרון :

ביטויים לוגיים פשוטים וביצוע חוזר מותנה

בפרק הקודם ראינו כי ישנם קטעים באלגוריתם החוזרים על עצמם. בפרק זה נלמד כיצד ניתן לכתוב זאת ובאילו תנאים.

ידע קודם : התלמידים למדו ותרגלו את סביבת העבודה, פקודות קלט/פלט, משתנים ואופרטורים.

מטרות השיעור

א. התלמידים יכירו ויבינו את המשמעות של ביטוי לוגי פשוט.

ב. התלמידים יכירו את המבנה הבסיסי של תנאי לוגי פשוט.

ג. התלמידים יכירו את מבנה ביצוע חוזר מותנה.

ד. התלמידים יתרגלו שילוב ביצוע חוזר מותנה בתוכנית.

ביטוי לוגי פשוט

ביטוי לוגי פשוט מורכב משני ערכים וביניהם אופרטור לוגי כמוצג בטבלה :

	סימן	דוגמה
שווה	==	5 == 5 (True) 5 == 6 (False)
שונה	!=	6 != 5 (True) 5 != 5 (False)
גדול	>	6 > 5 (True) 5 > 5 (False)
קטן	<	6 < 5 (True) 5 < 5 (False)
גדול שווה	>=	6 >= 5 (True) 5 >= 5 (True)
קטן שווה	<=	5 <= 6 (True) 5 <= 5 (True)

תרגיל ראשון

לפנידביטויים ופלטים לוגים. השלימו את הטבלה על פי הערכים או התוצאות בכל קטע קוד. שימו לכל לכל ביטוי נדרשת תוצאה לוגית, כלומר הפלט True או False (למעט סעיפים 15/16 בהם נדרשת תשובה מספרית).

	הפלט - תוצאת הביטוי הלוגי	הביטוי הלוגי
.1		`result = 10 >= 10` `print result`
.2		`result = 3.0 < 3` `print result`
.3		`result = 2**2 != 2*2` `print result`

	הפלט - תוצאת הביטוי הלוגי	הביטוי הלוגי
.4		result = ’a’ == ’A’ print result
.5		result = 10 != 10 print result
.6		a , b = 5 , 3 print a > b
.7		a = ’number’ b = ’number’ print a==b
.8	True	a = 5 print a + _____ > 8
.9	True	a , b = ______, 3 print a + b > 8
.10	False	result = _____ != 10 print result
.11	False	a = ’number’ b = ___________ print a==b
.12		a , b = 5 , 3 print a + b > 8
.13	True	a , b = ___ , ____ print a + b > _____
.14		result = False < True print result
.15		result = True * True print result
.16		result = True * False print result

ביצוע חוזר מותנה (while)

הוראה ביצוע חוזר מותנה, בודקת תנאי לוגי, וכל עוד (while) שהתנאי מתקיים (True) מתבצע קטע קוד מותנה, המשויך להוראה זו.

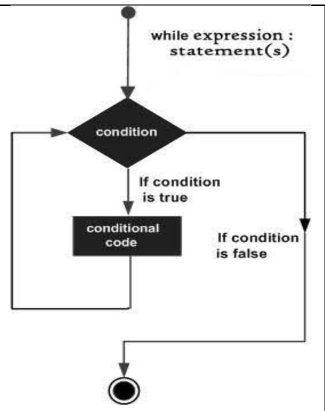
לאחר סיום ביצוע סידרת הפעולות המותנות, נבדק התנאי שוב, וחוזר חלילה.

הוראת ה-while מסתיימת בנקודותיים (:)

הקוד המותנה ייכתב מוזח ב-Tab אחד ביחס להוראת ה-while .

קטע הקוד המותנה מסתיים, כאשר מופיעות שורות קוד שאינן מוזחות.

ניתן להוסיף הוראת "אחרת" **(else)** במידה ונדרש להפעיל קוד שיתבצע פעם אחת בלבד לאחר שהתנאי הפסיק להתקיים.

מבנה אלגוריתמי		**מבנה ההוראה בשפת python**
חזור כל עוד <תנאי לוגי> (שורות של הקוד המותנה בקיום התנאי) אחרת (שורות של הקוד המותנה באי קיום התנאי)		while <condition>:[code line] [code lines] [else: code lines]

דוגמאות לשימוש בהוראת while :

קטע תכנית המונה עד הערך 2 ומדפסי הודעה בהתאמה :

הפלט שיוצג	**קטע התכנית**
The count is: 0 The count is: 1 The count is: 2 Bye bye!	count = 0 while count < 3: print 'The count is:', count count = count + 1 print "Bye bye!"

דוגמה לשימוש בהוראת while לביצוע חוזר לעולמים.

קטע הקוד מחשב סכום סידרת פיבונצ'י, עד שתופסק ע"י פעולה יוזמה של המשתמש ע"י Ctrl+C, לחצן העצירה או סגירת חלון הפלטים :

קטע התכנית	הפלט שיוצג

a , b = 1,1
count = 2
print 'Fibonacci item 1 is 1'
print 'Fibonacci item 2 is 1'
while True:
  a , b = b, a + b
  print 'Fibonacci item',count,
    'is',b
  count = count + 1
print "Bye bye!"
	The sum of 1 to 1 is 1
	The sum of 1 to 2 is 3
	The sum of 1 to 3 is 6
	The sum of 1 to 4 is 10
	וכך הלאה, הפלט 'Bye bye!' לעולם לא יודפס

תרגילים – ביצוע חוזר מותנה (while)

תרגיל ראשון

פרוק מספר לספרותיו

תכננו וכתבו תכנית הקולטת מספר שלם בעל מספרות שאינו ידוע. על התכנית להדפיס את ספרותיו החל מספרת האחדות ועד לספרה המשמעותית ביותר.

לאחר הדפסת הספרה המשמעותית ביותר יש להדפיס End of job

לדוגמה, עבור הקלט 569 תדפיס התכנית:

```
9
6
5
End of job
```

תרגיל שני

א. סכום הספרות, כל ספרה בשורה נפרדת

תכננו וכתבו תכנית הקולטת מספר שלם בעל מספרות שאינו ידוע. על התכנית להדפיס את סכום ספרות המספר.

לאחר הדפסת הספרה המשמעותית ביותר יש להדפיס End of job

לדוגמה, עבור הקלט 64925 תדפיס התכנית:

```
26
End of job
```

ב. כל הספרות באותה שורה עם הפרדת רווח בין ספרה לספרה

יש לשנות את התכנית כך שהפלט יראה כך:

```
9 6 5
End of job
```

תרגיל שלישי

סכום הספרות הזוגיות :

יש לכתוב תכנית הקולטת מספר שלם בעל מספרות שאינו ידוע. על התכנית להדפיס את סכום הספרות הזוגיות במספר.

לאחר הדפסת הספרה המשמעותית ביותר יש להדפיס End of job

לדוגמה, עבור הקלט 64925 תדפיס התכנית :

```
10
End of job
```

תרגיל רביעי

מספר הספרות הזוגיות והאי זוגיות :

לפניך קטע תכנית אותו כתב תלמיד. מטרת הקטע לקלוט מספר שלם ולהדפיס בשורה אחת כמה ספרות זוגיות וכמה ספרות אי זוגיות יש במספר.

לדוגמה, עבור הקלט 64925 תדפיס התכנית :

```
Even = 2 Odd = 3
```

יש להשלים את קטע התכנית :

```
num = int(raw_input('Enter an integer number:'))
count = _____
number_of_even_digits = _______
while num > 0:
  count ______
  if (_________________):
      number_of_even_digits += 1
  _________________
print 'Even = ', _________________________________
print 'Odd = ',  _________________________________
```

תרגיל חמישי

הרכבת מספר :

כתבו תכנית הקולטת מספר שלם כך שבכל בקשת קלט תינתן ספרה אחת החל מספרת האחדות, כלומר בתחילה תקלט ספרת אחדות, לאחר מכן ספרות העשרות וכך הלאה עד אשר נלחץ Enter מבלי שהוכנס דבר.

סיום הקליטה יודפס המספר כפי שספרותיו הוכנסו.

לדוגמה, עבור הקלט הבא (הסימן ↵ מסמן כי המשתמש לחץ על מקש ה-Enter) :

```
Enter digit: 0↵
Enter digit: 0↵
Enter digit: 1↵
Enter digit: ↵
```

יודפס :

```
The number is: 100
```

תרגיל שישי

(שאלה זו בהתאם לתכנית הלימודים מבוא לסייבר באמצעות שפת Python - מימוש המרות בסיסים).

כידוע בחיי יום יום אנו משתמשים בבסיס ספירה עשרוני (בעל עשר ספרות 0-9).

את המספר 8724 בבסיס עשרוני ניתן לפרק באופן הבא :

$8*1000 + 7*100 + 2*10 + 4*1$

בסיס ספירה בינארי מכיל את הספרות 0 ו-1. לדוגמה המספר 110011. להמרת מספר בינארי זה לערכו העשרוני יש לבצע פעולה זהה :

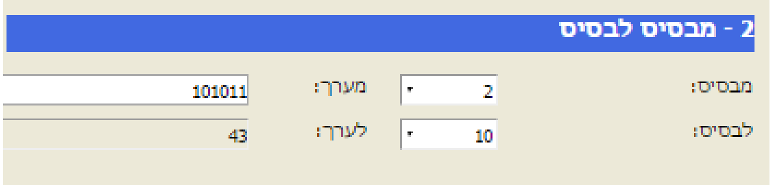
101011

$1*2^5 + 0*2^4 + 1*2^3 + 0*2^2 + 1*2^1 + 1*2^0$

בסכימת תוצאת המכפלות יתקבל הערך העשרוני של המספר הבינארי – בדוגמה שלנו

$(43)_{10} = (32+0+8+0+2+1)_{10} = (101011)_2$

לבדיקת הערכים יש להיכנס למחשבון מבסיס לבסיס להקיש את הערכים הנדרשים :



א.

יש לכתוב תכנית מספר הקולטת מספר בינארי. על התכנית להדפיס את ערכו העשרוני של המספר הנקלט.

אם נקלט המספר 101011 יש להדפיס 43 (כפי שהוכח קודם לכן).

ב.

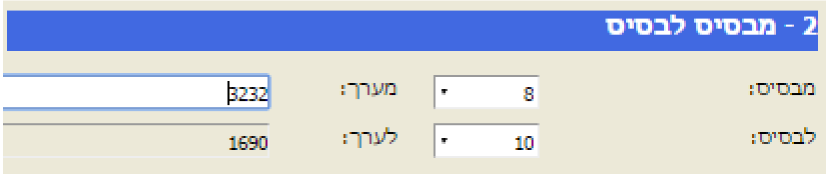
יש לשנות את התכנית כך שהתכנית תקלוט שני ערכים :

מספר בבסיס כלשהו

את בסיס המספר.

על התכנית לחשב ולהדפיס את ערכו העשרוני של המספר הנקלט.

לדוגמה אם נקלט המספר 3232 בבסיס 8 יש להדפיס את ערכו העשרוני - 1690



ג.

יש לכתוב תכנית הקולטת מספר שלם (בכל גודל) ואת הבסיס שאליו יש להעביר את ייצוגו.

ניתן להניח שהבסיס תמיד קטן מ-10.

על התכנית להדפיס את ספרות המספר בייצוגו החדש (ספרת היחידות, אח"כ העשרות, המאות וכו).

לבסוף תדפיס הודעה באיזה בסיס המספר מיוצג.

לדוגמה, עבור הקלט של שלם 14 ובסיס 2, יודפס:

```
0
1
1
1
in base 2
```

ד.

יש לשנות את התכנית כך שתדפיס את המספר המקורי בבסיס 10 ואת ייצוגו בבסיס החדש. למשל, עבור שלם 14 ובסיס 2, יודפס:

```
14 in base 10 is:
1110 in base 2
```

פונקציות

בשלב זה התלמידים יודעים לכתוב תכנית Python הקולטת נתונים, מעבדת נתונים, תוך שימוש בפקודות השמה, תנאים, לולאת (ביצוע חוזר) מותנות והדפסת התוצאות.
בשלב זה מומלץ לתכנן ולחלק את התכנית לתתי משימות ולכן נדרש להפוך את דרך התכנון וכתיבת התוכנה למודולרית.

מטרות השיעור

א. התלמידים יכירו את המושג פונקציה.
ב. התלמידים ילמדו להגדיר פונקציה פשוטה.
ג. התלמידים יתרגלו כתיבת תכניות המורכבות מפונקציות אותן יש לזמן מהתוכנית הראשית.

הפונקציה

קרויה גם שגרה Subroutin, פרוצדורה Procedure, שיטה Method.
רצף של פקודות המאוגדות יחדיו, במטרה לבצע מטלה מוגדרת, מימוש של אלגוריתם.
בפרקים הקודמים נעזרנו בפונקציות מובנות בשפת ה-Python כדוגמת:

`raw_input` – המדפיסה הנחיות למסך, וקולטת מהמקלדת.
`type` – המחזירה את טיפוס המשתנה.
`int, float, str` – הממירות טיפוסי משתנים.

ניתן להרחיב את מגוון הפונקציות, באמצעות הגדרה של פונקציות נוספות.

מדוע פונקציה?

- היכולת להגדיר פונקציות חדשות בשפה מאפשרת לנו להעשיר את השפה ולכתוב תכניות ברורות יותר. פונקציות חדשות מאפשרות לנו לכתוב תכניות מודולאריות, לקבל תכניות קריאות וברורות יותר. זה נוח לניפוי שגיאות והכנסת שינויים בעתיד.
- תכנות טוב מבוסס גם על חלוקת תכנית למספר קטעים, כך שלכל קטע מטרה מוגדרת אחת. קטע זה יוגדר כפונקציה.
- כאשר קטע קוד עושה יותר מדי פעולות, או כאשר קטעי קוד חוזרים על עצמם במקומות שונים, יש לחלק את הקוד לפונקציות קטנות יותר.
- כך ניתן לפשט תכנית שמבצעת משימות מורכבות לתוכנית המפעילה פונקציות, שבה כל פונקציה לבדה מבצעת משימה פשוטה.

היתרונות האפשריים בהגדרות ושימוש בפונקציות:

- חלוקת קוד בעל שורות רבות תהפוך את הקוד נוח לקריאה והבנה.
- הקוד פשוט יותר לתחזוקה - קל יותר לשנות או לתקן כל פונקציה המתמקדת בפעולה מסוימת מאשר בכל הקוד. כאשר מאתרים בעיה בפעולה מסוימת, כל שנדרש הוא לתקן את הפונקציה האחראית לפעולה זו, מבלי שהדבר ישפיע על שאר חלקי התוכנית.

- הקוד ניתן לשימוש חוזר. אם הפונקציה נבדקה ועובדת, ואנו נדרשים לפעולתה בחלק אחר של התוכנית, או בתכנית אחרת, אין צורך לכתוב את הקוד מחדש.

הגדרת מבנה הפונקציה:

שם הפונקציה   פרמטרים

```
def <function_name>([<arguments>]):
    <lines of code>
    [return] [value(s)]
```

קטע הקוד שמבצעת הפונקציה מוזח ב-TAB ביחס לשורת ההגדרה

ערך(ים) שהפונקציה מחזירה לתכנית

דוגמאות לשימוש בפונקציות :

מטרה	הסבר	מבנה ודוגמה

הגדרת פונקציה ללא פרמטרים	הפונקציה המדפיסה פנים מחייכות ופנים עצובות	`def faces ():`
		`  print '☺'`
		`  print '☹'`

פונקציה המקבלת פרמטרים	הפונקציה המקבלת שני מספרים ומדפיסה תרגיל כפל ואת התוצאה	`def multiply (num1,num2):`
		`  num3 = num1 * num2`
		`  print num1,'*',num2,'=',num3`

פונקציה המקבלת פרמטרים ומחזירה ערך	פונקציה המקבלת רדיוס של מעגל ומחזירה את שטחו
	
	הפונקציה המקבלת שני מספרים ומחזירה את מכפלת המספרים
		`def circle_area(radius):`
		`  area = 3.14 * radius ** 2`
		`  return area`
		
		`def multiply (num1,num2):`
		`  num3 = num1 * num2`
		`  return num3`

<u>זימון פונקציה</u>

זימון פונקציה פירושו קריאה והפעלת הפונקציה מהתוכנית. הדבר יעשה, בדומה לפונקציה מובנית, על ידי שילוב שם הפונקציה בקוד התכנית.

לדוגמה:

התכנית הבאה, מבקשת כקלט רדיוס. אם הקלט חיובי תדפיס התכנית את שטח המעגל. לאחר מכן תחזור התכנית על פעולה זו שוב ושוב. במקרה בו נקלט רדיוס 0 או מספר שלילי, התכנית תסתיים, ותדפיס 'Done'. התכנית עושה שימוש בפונקציה circle_area שהוגדרה בדוגמה לעיל:

```
def circle_area(radius):
  area = 3.14 * radius ** 2
  return area
```

מבנה התכנית הראשית

```
# The main Program
r = int(raw_input('Enter circle radius:'))
while r > 0
    a = circle_area(r)
    print 'The circle area is:',a
    r = int(raw_input('Enter circle radius:'))
print 'Done'
```

זימון פונקציות מובנות

זימון פונקציה

תיעוד הפונקציה

בתחילת כל פונקציה יש לשלב תיעוד/הערה המתארת את תפקידה העיקרי של הפונקציה. במידת הצורך יש לתאר את הפרמטרים הנדרשים, הערך המוחזר על ידה או חישוב מורכב המוכל בפונקציה.

הערה בת שורה אחת תחל עם סולמית.

הערה בת כמה שורות תוכלל בין זוג של שלשת גרשים גרשיים.

דוגמאות:

הערות שהוגדרו כך שבכל שורה מוגדרת הערה נפרדת.	`# This is one-line remark.` `# This is another one-line remark.`

הערה המתפרסת על מספר שורות.	`'''`
	`This is a remark`
	`That can be extended`
	`To several lines`
	`'''`

כמו בהגדרת מחרוזת ניתן להגדיר שלוש גרשיים במקום שלוש גרש בודדה	`""" This is a remark`
	`That can be extended`
	`To several lines """`

טיפ בעבודת התכנות!

בעת בדיקת תכנית שאינה פועלת כראוי, ניתן להגדיר סימוני ההערה כדי "להעלים" חלקי קוד חשודים, ולבודד את תקינות שאר חלקי הקוד.

כתיבת התכנית הראשית

לניצול היתרונות של כתיבת פונקציות, ושימוש חוזר בהן, ראוי לסגל כתיבת תכנית ראשית באופן הבא:

א. קוד התכנית הראשית ייכתב בפונקציה שתיקרא, למשל, main.

ב. בקובץ תכתב תכנית קצרה שתזמן של הפונקציה main, רק במידה והקובץ הורץ.

בדרך זו אם הפונקציות יזומנו מתכניות הנמצאות בקבצים אחרים, התכנית הראשית לא תופעל.

יש לכתוב את התכנית הראשית כך:

```
def main():
  <lines of main code>
```

```
if __name__ == '__main__':
  main()
```

הערה - בסביבת הענן של repl.it יש להחליף את `__main__` ב- `__builtin__`

דוגמה:

כך נכתוב תכנית הכוללת זימון פונקציה והתכנית הראשית (על פי דוגמאות קודמות):

```
def circle_area (radius)
  #calculate the area of a circle
  area=3.14 * radius ** 2
  return area

def main():
  '''
  calculate the area of a circle
  over and over, until input is not
  a positive number
  '''
  r = int(raw_input('Enter circle radius:'))
  while r > 0:
    a= circle_area(r)
    print 'The circle area is:',a
    r = int(raw_input('Enter circle radius:'))
   print 'done'

if __name__=='__main__':
  main()
```

תרגילים – כתיבת פונקציות

תרגיל ראשון

א.
בתרגול קודם הגדרת תכנית הממירה מבסיס 10 לבסיס כלשהו.
הפכו תכנית זו לפונקציה בעל החתימה הבאה:

base10_to_anyBase (num, base)

הפונקציה תקבל כפרמטר מספר שלם ומספר המייצג בסיס הנדרש לייצוג. על הפונקציה להחזיר מספר שלם שערכו בייצוג בבסיס הנדרש. יש לזכור לתעד את הפונקציה.

ב.
באופן דומה בתרגול קודם הגדרת תכנית הממירה מספר שלם מבסיס כלשהו לבסיס 10.
הפכו תכנית זו לפונקציה בעלת החתימה הבאה:

anyBase_to_base10 (num10, base)

הפונקציה תקבל כפרמטר מספר שלם ומספר המייצג בסיס הנדרש לייצוג. על הפונקציה להחזיר שלם שערכו מיוצג בבסיס 10. זכרו לתעד.

ג.
יש לכתוב תכנית ראשית המזמנת את שתי הפונקציות הנ״ל. יש לזמן כל פונקציה עבור 3 זוגות של מספרים ובסיסים. על התכנית להדפיס את הערכים והבסיסים לפני ההמרה ולאחריה.

לדוגמה:

12321 in base 4 is: 441 in base 10

765 in base 8 is: 501 in base 10

4321 in base 5 is: 586 in base 10

223 in base 10 is: 11011111 in base 2

999 in base 10 is: 1330 in base 9

2017 in base 10 is: 5611 in base 7

ד.
יש לכתוב תכנית הקולטת מספר שלם בבסיס 10 ומדפיסה את ייצוגו בכל הבסיסים 2 – 9. יש לזמן את הפונקציה `base10_to_anyBase` שהגדרה קודם לכן.

לדוגמה, עבור הקלט 100, על פלט התכנית להראות כך:

Enter an integer number: 100

100 in base 10 is: 121 in base 9

100 in base 10 is: 144 in base 8

100 in base 10 is: 202 in base 7

100 in base 10 is: 244 in base 6
100 in base 10 is: 400 in base 5
100 in base 10 is: 1210 in base 4
100 in base 10 is: 10201 in base 3
100 in base 10 is: 1100100 in base 2

מבוא למחרוזות

מחרוזת היא אחת מטיפוסי הנתונים העיקריים ב-Python.
בפרקים הראשונים טיפלנו במחרוזות כמשתנה וכחלק מפעולות קלט ופלט.
בפרק זה נלמד פעולות נוספות ומורכבות יותר אשר ייקלו על התכנות בסביבה זו.
הוראת הביצוע חוזר פעמים, מאפשרת, לסרוק וטפל בערכי המחרוזת.

מטרות השיעור

א. התלמידים יכירו את מבנה המחרוזות.
ב. התלמידים יכירו פעולות בסיסיות על מחרוזות.
ג. התלמידים יתרגלו כתיבת קוד לעיבוד מידע במחרוזות.
ד. התלמידים יכירו את הוראת הביצוע החוזר for.
ה. התלמידים יממשו המרת קוד ספירה מ/אל בסיסי ספירה.

מחרוזת

מחרוזת היא רצף סדרתי של תווים אלפאנומריים, התחום ע״י זוג גרשים או גרשיים.

```
var1 = 'Hello World'
var2 = "I am programming with Python "
```

מכיוון שהתווים מסודרים ברצף, ניתן להתייחס אל כל אחד מהם באמצעות אינדקס המציין את מרחקם מהתו הראשון, או לחילופין באמצעות אינדקס שלילי המציין את מרחקם מסוף המחרוזת.
לדוגמה עבור המחרוזת הבאה:

```
v = 'Water'
```

אנדקסי המיקום יהיו:

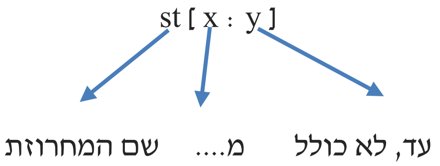
0	1	2	3	4
W	**a**	**t**	**e**	**r**
-5	-4	-3	-2	-1

ניתן לגשת לתו מסוים על ידי ציון מקומו במחרוזת. לדוגמה:

הפלט שיוצג	הוראת הפלט
W	`print v[ 0 ]`
W	`print v[ -5 ]`
t	`print v[ 2 ]`
e	`print v[ -2 ]`
e	`print v[ 3 ]`
T	`print v[ -3 ]`
rat	`print v[ 4 ] + v[ 1 ] + v[ 2 ]`

<u>פעולות על מחרוזות ופונקציות מובנות</u>

ניתן להתייחס לקטע (מספר תווים או מילים) בתוך המחרוזת על ידי ציון אינדקס מ- ואינדקס עד (לא כולל), המופרדים בנקודתיים, על פי המבנה הבא:



לדוגמה:

הסבר	הפלט שיוצג	הוראת הפלט
0 1 2 3 4 W a t e r	at	`print v[1:3]`
0 1 2 3 4 W a t e r + f	Waf	`s = v[0:2] + 'f'`
-5 -4 -3 -2 -1 W a t e r	ate	`print v[-4:-1]`
	Wafer	`s = v[0:2] + 'f' + v[3:5]` `print s`
	WaWa	`s = v[0:2] * 2` `print s`

תרגיל ראשון

לפניך שורת הקוד הבאה :

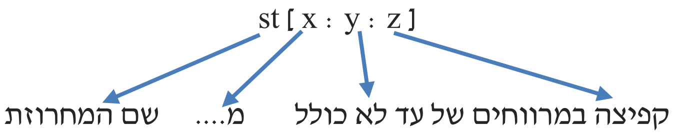
st = 'ILovePython'

0	1	2	3	4	5	6	7	8	9	10
I	**L**	**O**	**V**	**e**	**P**	**y**	**t**	**h**	**o**	**N**
-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

לכל שורת קוד יש להגדיר את הפלט המתקבל :

	שורות הקוד	פלט
1.	`s = st[0:2] + v[5]` `print s`	
2.	`s = st[1:5]` `print s`	
3.	`s = st[-10:-6]` `print s`	
4.	`s = st[-6:-4]` `print s`	
5.	`s = st[-11:-6] + 'Me'` `print s`	

קיימת אפשרות להתייחס לתווים הנמצאים במרחקים קבועים במחרוזת. זאת על ידי הוספת אינדקס דילוג.



לדוגמה :

קוד התכנית	הפלט שיוצג

`digits = '0123456789'`
`even = digits[0:10:2]`
`trio = digits[3:10:3]`
`print even`
`print trio`
	
	02468
	369

בדוגמה הנ"ל, מכיוון שהפעולות עבור odd ו-trio הן על כל אברי המחרוזת, ניתן היה לכתוב אותן גם בדרך הבאה :

קוד התכנית	הפלט שיוצג
`digits = '0123456789'`	

even = digits[::2]	
trio = digits[3::3]	
print even	02468
print trio	369

שימוש באינדקס דילוג שלילי מאפשר החזרת התווים בסדר הפוך, מהסוף המחרוזת לתחילתה.
לדוגמה,

קוד התכנית	**הפלט שיוצג**
digits = '0123456789'	
print digits[::-1]	9876543210

בטבלה הבאה מופיעה רשימה חלקית של פעולות אפשרויות לביצוע על מחרוזות (חלק מהפעולות הזכרנו בתרגול קודם).

המשתנים s ו-t מייצגים תווים או מחרוזות.

x הוא תו או רצף תווים.

n מייצג מספר חיובי שלם.

k, j ו-i מייצגים מספרים שלמים

הפעולה	תיאור הפעולה
x in s	מחזירה True אם האיבר x (תו או רצף תווים) מוכל בתוך מחרוזת s, בכל מקרה אחר תחזיר False
x not in s	מחזירה False אם האיבר x(תו או רצף תווים) נמצא בתוך מחרוזת s, בכל מקרה אחרת יוחזר True (פעולה הופכית לפעולה הקודמת)
s + t	מחזירה מחרוזת שהיא שרשור של שתי המחרוזות s ו-t
s*n, n*s	לכל n טבעי (שלם גדול מאפס) תוחזר מחרוזת שהיא שרשור של s לעצמו כפול n פעמים. לכל n קטן או שווה לאפס תוחזר מחרוזת ריקה
s[i]	החזרת התו ה-i במחרוזת s (מתחיל מ-0).
s[i: j]	החזרת תת מחרוזת של מחרוזת s ממקום i ועד j לא כולל.
s[i: j: k]	החזרת תת מחרוזת של s ממקום i ועד j לא כולל, בקפיצות של k
s.index(x)	מחזירה מספר שלם שהוא אינדקס המיקום של x (המכיל תו או רצף תווים) במחרוזת s
s.count(x)	מחזירה שלם, שהוא מספר החזרות של x (המכיל תו או רצף תווים) במחרוזת s

בטבלה הבאה חלק מהפונקציות המובנות ב-python לטיפול בנתונים סדרתיים :

הפונקציה	תיאור הפונקציה
len(s)	מחזירה מספר שלם המייצג את מספר התווים במחרוזת
min(s)	מחזירה את התו בעל ערך ה-ASCII הנמוך ביותר במחרוזת s
max(s)	מחזירה את התו בעל ערך ה-ASCII הגבוה ביותר במחרוזת s
cmp(s,t)	משווה שתי מחרוזות, s ו-t באופן מילוני, ומחזירה : -1 עבור s < t 0 עבור s = t 1 עבור s > t

כאמור זאת רשימה חלקית לפעולות מובנות לטיפול במחרוזות. את רשימת הפעולות המלאה ניתן למצוא באתר ארגון ה-python :

https://docs.python.org/2/library/stdtypes.html#string-methods

לדוגמה :

s.upper()	מחזירה עותק של המחרוזת s בה כל האותיות האנגליות גדולות
s.lower()	מחזירה עותק של המחרוזת s, בה כל האותיות האנגליות קטנות

תרגילים – מחרוזות

תרגיל שני

לפניך שורת הקוד הבאה:

st = ʹStudyingPythonʹ

לכל שורת קוד יש להגדיר את הפלט המתקבל:

	שורות הקוד	פלט
1.	`s = st[::]` `print s`	
2.	`s = st[::-1]` `print s`	
3.	`s = st[::2]` `print s`	
4.	`s = st[::-2]` `print s`	
5.	`s = st[:5]` `print s`	
6.	`s = st[:-6]` `print s`	
7.	`s = st[8:]` `print s`	
8.	`s = st[8::2]` `print s`	
9.	`s = st[8::-2]` `print s`	
10.	`s = st[:9-1]` `print s`	

תרגיל שלישי

ב"סיבוב" מחרוזת התו בתחילת המחרוזת מועבר לסופה.

לדוגמה, המחרוזת abcde לאחר "סיבוב" תראה bcdea

יש לכתוב פעולה המקבלת מחרוזת. על הפעולה לסובב את המחרוזת עד לקבלת המחרוזת המקורית.

בדוגמה שלנו יינתנו הפלטים הבאים:

bcdea

cdeab

deabc

eabcd

abcde

תרגיל רביעי

א.

יש לכתוב פונקציה המקבלת מחרוזת טקסט, הופכת אותה, ומחזירה אותה כשהאות הראשונה והאחרונה הן אותיות אנגליות גדולות.

לדוגמה עבור המחרוזת abcdef יוחזר FedcbA

ב.

יש לכתוב תכנית הקולטת מחרוזות עד אשר נקלטה המחרוזת stop. לכל מחרוזת יש להדפיס המחרוזת כשהאות הראשונה והאות הראשונה גדולות.

יש להשתמש בפונקציה שנכתבה בסעיף הקודם.

תרגיל חמישי

יש לכתוב תכנית הקולטת מחרוזות עד לקבלת הקלט done. כל מחרוזת יש להדפיס באופן הבא:

בשורה ראשונה את מחצית התווים

בשורה שנייה את המחצית השנייה באותיות גדולות.

לדוגמה, עבור הקלט abcdef יש להדפיס

abc

DEF

תרגיל שישי

א.

חזרו לפונקציה שכתבתם בתרגול קודם - פונקציה הממירה מבסיס 10 לבסיס כלשהו. יש לשנות את הפונקציה כך שתמיר ערך מבסיס 10 לבסיס כלשהו בתחום 2 (בינארי) ועד 16 (הקסאדצימאלי):

base10_to_anyBase (num , base)

הפונקציה מקבלת כפרמטרים מספר שלם, ובסיס נדרש לייצוג (2 עד 16).

הפונקציה תחזיר מחרוזת המכילה את המספר בייצוג בבסיס הנדרש.

הנחיות לפתרון:

- הגדרת מחרוזת: up2hex = '0123456789ABCDEF'
- הגדרת משתנה עבור מחרוזת המספר המבוקש, ואיתחולו במחרוזת ריקה:

numBase=''

- בהליך המרת הבסיס יש להיעזר בשארית כאינדקס לקבלת התו המתאים מהמחרוזת up2hex.
- יש לשרשר תו זה למחרוזת המספר הנבנית ב-numBase.

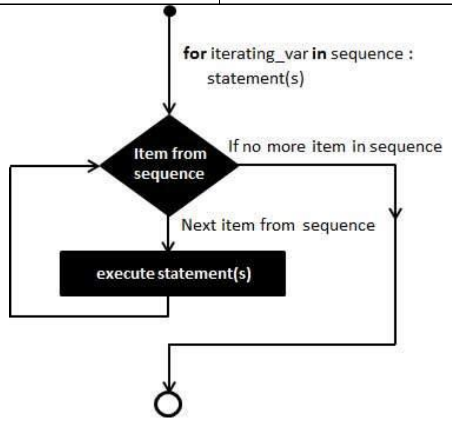
ב.

יש לכתוב תכנית הקולטת מספרים שלמים בבסיס 10. לכל מספר יש להדפיס את ייצוגו בכל הבסיסים 2 – 16. חובה להשתמש בפונקציה שהוגדרה בסעיף קודם base10_to_anyBase.

ביצוע חוזר פעמים (for) על מחרוזות

לעיתים אנו נדרשים לעבור על פני רצף סידרתי של איברים כדי לבחון ו/או לעבד כל איבר בנפרד.
הוראת ה-for מאפשרת לעבור על טיפוס מחרוזת, תו אחרי תו, עד לסוף המחרוזת. בכל פעם שנשלף תו, מתבצע הקוד המשויך ללולאת ה-for.

מבנה אלגוריתמי	**מבנה ההוראה בשפת python**
חזור עבור כל איבר var בסדרה seq_var (שורות של הקוד)	for <var> in <seq_var>: [code lines]



הוראת ה-for מסתיימת בנקודתיים (:)
אם הקוד המשויך ללולאת ה-for כולל הוראה אחת ניתן לכתבו לאחר הנקודתיים.
אם הקוד המותנה כולל יותר מהוראה אחת יש לכתוב אותו כך שהוא מוזח ב-Tab אחד ביחס להוראת ה-for .
כמו בהוראות הקודמות, קטע הקוד המשויך ללולאה מסתיים כאשר מופיעות שורות קוד שאינן מוזחות.

דוגמאות לשימוש בהוראת for וסריקת מחרוזות:

הסבר	הפלט	קטע התכנית
הקטע סורק מחרוזת מתחילתה ועד סופה ומדפיס כל תו במחרוזת ומיקומו (אינדקס). בסיום מודפס Done	C 0 y 1 b 2 e 3 r 4 Done	str = 'Cyber' for chr in str: print chr , str.index(chr) print 'Done'
הקטע סורק מחרוזת מסופה ועד תחילתה ומדפיס כל תו במחרוזת ומיקומו (אינדקס). בסיום מודפס Done	r 4 e 3 b 2 y 1 C 0 Done	str = 'Cyber' for chr in str[::-1]: print chr,str.index(chr) print 'Done'

הסבר	הפלט	קטע התכנית

הקטע ממיר את המספר השלם למחרוזת ומדפיס את הערכים החל מהספרה המשמעותית ביותר	3
	4
	5
	6
		`num = 3456`
		`for k in str(num):`
		`  print k`
		`print 'Done'`

הפיכת מחרוזת המכילה תווים מספריים למספר שלם.
בסיום מודפסים בשורה ראשונה המספר ושורה שנייה ערכו מוכפל ב-2
	4356	`decimal = '0123456789'`
		`snum = '4356'`
		`num = 0`
		`for x in snum:`
		`  num = num * 10 +`
		`decimal.index(x)`
		`print num`
		`print num * 2`

<u>**תרגילים – For ומחרוזות**</u>

<u>**תרגיל ראשון**</u>

יש לכתוב פונקציה המקבלת מחרוזת המכילה תווים מספריים.

חתימת הפעולה:

```
def sum_string_number (st):
```

על הפעולה להחזיר את סכום התווים המספריים.

לדוגמה, עבור המחרוזת 1234 יוחזר 10

<u>**תרגיל שני**</u>

לפניך מספר קטעי קוד. לכל קטע יש להגדיר את הפלט:

פלט	שורות הקוד	

	`st = 'abcde'`
	`for x in st:`
	`  print st.index(x)`
		1.

	`st = 'abcde'`
	`for x in st:`
	`  print x, x.upper(x)`
		2.

	`st = 'aBcdE'`
	`for x in st:`
	`  print x.islower()`
		3.

	`st = 'abcde'`
	`for x in st[2:]:`
	`  print x`
		4.

	שורות הקוד	פלט

5.	`st = 'abcde'`
	`for x in st[::-1]:`
	`  print x`	

תרגיל שלישי

א.

חזרו לפונקציה הממירה מבסיס כלשהו לבסיס 10 :

anyBase_to_base10 (num10, base)

יש לכתוב את הפונקציה מחדש כך שתמיר כל בסיס בתחום 2 - 16, לבסיס 10.

שימו לב כי הקלט יכול להכיל אותיות גדולות או קטנות. לדוגמה :

$1ca3_{16}$

$FF23_{16}$

הנחיות לכתיבת הפונקציה :

- הגדרת המחרוזת : up2hex = '0123456789ABCDEF'
- קליטת המספר להמרה כמחרוזת ולהמרת תווי האותיות לאותיות גדולות.
- מעבר תו אחר תו.
- היעזרו בפעולה index ובמחרוזת up2hex כדי לקבל את ערכו המספרי של התו.
- חיבור מכפלת משתנה צובר בבסיס :

```
num10 * base + Up2hex.index(x)
```

ב.

כתבו תכנית המדפיסה לכל בסיס מ-2 ועד 16 את המספר בן 4 הספרות הכי גדול לאותו בסיס (1111 לבסיס 2, 2222 לבסיס 3, וכך הלאה עד ffff לבסיס 16) ואת המספר בייצוגו בבסיס 10 לאחר שהומר באמצעות הפונקציה שהוגדרה בסעיף קודם.

פלט התכנית לדוגמה :

1111 in base 2 is: 15

2222 in base 3 is: 80

3333 in base 4 is: 255

וכך הלאה עד

ffff in base 16 is: 65535

הוראות תנאי ותנאים מורכבים

הוראת תנאי בודקת ופועלת על פי קיומו (או אי קיומו) של תנאי לוגי. במידה והתנאי מתקיים (True) יתבצע קטע קוד מותנה המשויך להוראת זו.

בניגוד להוראת ה-while שבה הקוד המותנה מתבצע שוב ושוב כל עוד התנאי מתקיים, בהוראת תנאי, הקוד המותנה מתבצע פעם אחת בלבד לאחר הבדיקה. לאחר מכן התכנית ממשיכה הלאה.

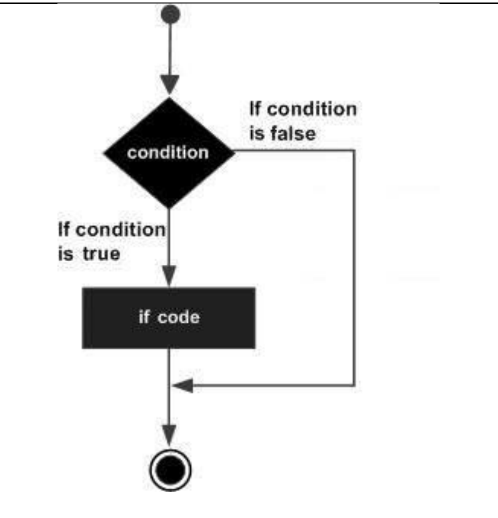
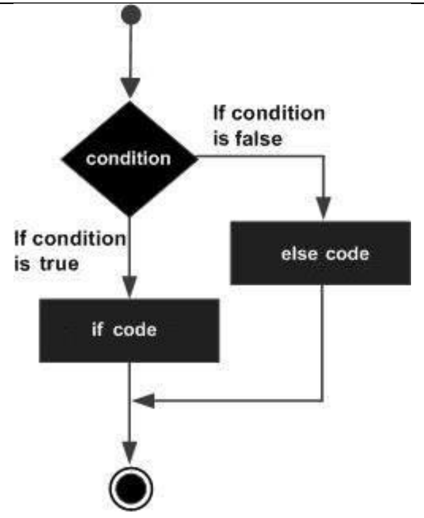
מטרות השיעור

א. התלמידים יכירו הוראות תנאי ויתרגלו אותן.

ב. התלמידים יכירו תנאים לוגיים מורכבים ויתרגלו אותם.

ג. התלמידים יממשו מחשבון המרה מבסיס כלשהו לבסיס כלשהו

הוראת תנאי if

<table>
<tr><th>מבנה אלגוריתמי</th><th colspan="2">מבנה ההוראה בשפת python</th></tr>
<tr><td>אם <תנאי לוגי>
(שורות קוד המותנה בקיום התנאי)
[אחרת
(שורות קוד המותנה באי קיום התנאי)]</td><td colspan="2">if <condition>: [code line]
[code lines]
[else:
code lines]</td></tr>
<tr><td>
</td><td>או</td><td>
</td></tr>
</table>

שימו לב כי בדומה להוראות קודמות, הוראת ה-if מסתיימת בנקודתיים (:)

הקוד המותנה ייכתב מוזח ב-Tab אחד ביחס להוראת ה-if .

קטע הקוד המותנה מסתיים, כאשר מופיעות שורות קוד שאינן מוזחות.

ניתן להוסיף הוראת "אחרת" (else) במידה ונדרש להפעיל קוד שיתבצע והתנאי אינו להתקיים (False).

דוגמה לשימוש בהוראת if פשוטה:

התשלום ללינת לילה באכסניית נוער הנו 100 ₪. בכל חדר 4 מיטות. מדריך קבוצה רוצה לדעת כמה תעלה הלינה לקבוצה. אם מספר האנשים בקבוצה, אינו מתחלק ל-4 בדיוק, נדרש חדר נוסף לאחרים. התכנית תקלוט כמה אנשים בקבוצה, תחשב ותדפיס את עלות הלינה של כל חברי הקבוצה.

```
group_size = int(raw_input('Enter group size: '))
```

```
rooms = group_size / 4
if (group_size % 4) > 0:
  rooms = rooms + 1
print 'The cost is:', rooms * 100, 'NIS'
```

דוגמה לשימוש בהוראת if - else:

קטע תכנית הבודק משתנה בוליאני של חיישן תאורת שמש. אם ערכו True, כלומר יש אור שמש, המערכת קוראת לפונקציה שתכבה את תאורת הרחוב. אחרת קוראת לפונקציה שתדליק את תאורת הרחוב. המערכת מדווחת על מצב התאורה.

```
if sense:
  turn_lights_off()
  print 'lights off'
else:
  turn_lights_on()
  print 'lights on'
```

<u>תנאי מקונן ו-elif</u>

לעיתים לאחר בדיקת תנאי, נדרשת בדיקת תנאי נוסף. במקרה זה בתוך הקוד המותנה, תופיע הוראת תנאי נוספת (קינון פונקציה).

דוגמה לשימוש בהוראת if מקוננת:

השוואת שתי מחרוזות הדפסת החרוזת הארוכה. במידה ואורכי המחרוזות שווים תודפס הודעה מתאימה.

```
text1 = raw_input('Enter text: ')
text2 = raw_input('Enter another text: ')
if len(text1) > len(text2):
  print text1,'is longer'
else:
  if len(text1) < len(text2):
    print text2,'is longer'
  else:
    print 'Both texts are equal'
```

במקרים בהם נדרשת בדיקה מרובה ניתן להעזר במבנה הבא:

מבנה אלגוריתמי	מבנה ההוראה בשפת python

אם <תנאי לוגי>
(שורות קוד המותנה בקיום התנאי)
אחרת אם <תנאי לוגי>
שורות קוד המותנה בקיום התנאי)
אחרת אם <תנאי לוגי>
שורות קוד המותנה בקיום התנאי)
וכך הלאה . . .
	if <condition>:
	    [code line]
	    [code lines]
	[elif <condition>:
	    code lines]
	[elif <condition>:
	    code lines]
	[elif <condition>:
	    code lines]
	. . . and so on

דוגמה לשימוש בהוראת if…elif:

```
text1 = raw_input(’Enter text: ’)
text2 = raw_input(’Enter another text: ’)
if len(text1) > len(text2):
  print text1,’is longer’
elif: len(text1) < len(text2):
  print text2,’is longer’
else:
  print ’Both texts are equal’
```

תרגילים – תנאים

תרגיל ראשון

יש לכתוב תכנית הקולטת את צבע הרמזור להולכי רגל. אם הצבע ירוק (Green), יש להדפיס Pass, בכל מקרה אחר יודפס Stop.

שימו לב כי ייתכנו מספר סוגי קלטים לקליטת צבע ירוק - לדוגמה, green, GREEN, GrEen או כל קומבינציה של אותיות קטנות וגדולות.

תרגיל שני

יש לכתוב תכנית הקולטת את צבע הרמזור לרכבים. אם הצבע ירוק (Green), יש להדפיס Go, אם הצבע צהוב יש להדפיס Proceed with caution , אם הצבע אדום יש להדפיס Stop ולכל צבע אחר יש להדפיס Error.

שימו לב כי בשאלה זו ייתכנו מספר סוגי קלטים לקליטת כל צבע, למשל לצבע ירוק תיתכן קליטה כגון green, GREEN, GrEen או כל קומבינציה של אותיות קטנות וגדולות.

תרגיל שלישי

יש לכתוב תכנית הקולטת ציון (מספר שלם) בין 0 ל-100.

אם הציון 100 תדפיס Excellent, אם הציון שווה או יותר מ-90 תדפיס Very good, אם הציון שווה או יותר מ-80 תדפיס Good, אם הציון שווה או יותר מ-60 תדפיס You should do better , אחרת תדפיס You should relearn.

ניתן להניח שהקלט חוקי.

תרגיל רביעי

יש לכתוב פונקציה המקבלת מחרוזת המכילה תווים מספריים.

חתימת הפעולה:

```
def plat_with_string_number (st)
```

על הפעולה:

א. להדפיס את התווים המספריים

ב. להחזיר את סכום התווים המספריים האי זוגיים.

לדוגמה, עבור המחרוזת 12345

יש להדפיס 4 2 ולחזיר את הערך 8 (1+3+5)

<u>תרגיל חמישי</u>

א.

יש לכתוב פונקציה המקבלת מחרוזת המכילה תווים מספריים המייצגים מספר שלם כלשהו ומספר שלם המייצג בסיס ספירה בין 2 ל-16 בו מיוצג המספר.

חתימת הפונקציה :

```
def check_num (numBase, base):
```

על הפונקציה לבדוק האם `numBase` חוקי בבסיס `base`, כלומר אינו כולל ספרות שמחוץ לבסיס.

לדוגמה מספר בבסיס 2 יכול לכלול רק את הספרות 0 ו-1.

על הפונקציה להחזיר True אם המספר תקין או False אם המספר מכיל לפחות ספרה אחת שגוייה.

רמז : עברו על המספר/מחרוזת תו אחרי תו ובדקו האם ערכו נמוך מערך הבסיס.

ב.

יש לכתוב תכנית הקולטת מספר שלם ובסיס בו מיוצג מספר זה. במידה והמספר שגוי יש להדפיס הודעה מתאימה ולקלוט מספר חדש עד לקבלת מספר תקין.

הניחו כי קלט הבסיס (`base`) חוקי.

תנאי לוגי מורכב

תנאי לוגי מורכב הוא תנאי הכולל לפחות שני תנאים פשוטים ובניהם יחס/קשר לוגי.

היחסים הלוגיים הם:

הפעולה	משמעות	משמעות
<ביטוי> **and** <ביטוי>	וגם	מחזירה True רק עם כל הביטויים True. בכל מקרה אחר מחזירה False
<ביטוי> **or** <ביטוי>	או	מחזירה True אם לפחות אחד מהביטויים True. בכל מקרה אחר מחזירה False
<ביטוי> **not**	לא	הופכת את המצב הלוגי של הביטוי: מחזירה True אם הביטוי False. מחזירה False אם הביטוי True.

דוגמאות:

התנאי הבודק האם x נמצא בטווח שבין 5 ל-10 (10 > x > 5):

```
x > 5 and x < 10
```

התנאי הבודק האם x נמצא בטווח שבין 5 ל-10 (כולל):

```
x >= 5 and x <= 10
```

התנאי הבודק האם האור ברמזור הוא אדום או ירוק:

```
color=='green' or color=='red'
```

במקום התנאי `a != b`

ניתן לרשום את התנאי: `not(a == b)`

דוגמה לשימוש בתנאי לוגי מורכב:

שעת מנוחה מוגדרת בין 14 ל-16 (כולל) או בין 23 ל-6 למחרת. כל שעה מחוץ לשעות אלו מוגדרת כשעת פעילות.

התכנית הבאה קולטת שעה ומדפיסה האם זו שעת מנוחה או שעת פעילות.

```
time = int(raw_input('Enter current hour:'))
if (time >= 14 and time < 16) or (time >= 23 or time < 6):
  print 'It is rest time'
else:
  print 'It is active time'
```

<u>תרגילים – תנאים מורכבים</u>

<u>תרגיל ראשון</u>

שנה מעוברת (29 לפברואר) מתרחשת כל 4 שנים למעט מתחילת מאה (למשל 1800, 1900) אשר לא תוגדר כשנה מעוברת. אולם אם זו תחילה של מאה רביעית בסדרה (למשל 400, 800... 2000...) זו תוגדר כשנה מעוברת.

יש לכתוב תכנית הקולטת מספר שלם ארבע ספרתי המייצג שנה. על התכנית לבדוק ולהדפיס הודעה מתאימה האם זו שנה מעוברת (leap year) או לא.

בפתרון נסו להגדיר הוראת תנאי מורכבת אחת בלבד.

<u>תרגיל שני</u>

א.

יש לכתוב פונקציה המקבלת כפרמטר מספר שלם המייצג בסיס ספירה.

חתימת הפונקציה: check_base(base)

על הפונקציה להחזיר True אם הבסיס הוא בין 2 ל-16. בכל מקרה אחר יוחזר False.

ב.

יש לכתוב תכנית הקולטת מספר שלם המייצג בסיס בין 2-16. עבור קלט חוקי יש להדפיס הודעה מתאימה ולהמשיך בקליטת ערך הבסיס עד לקליטת בסיס תקין.

יש להיעזר בפונקציה שהוגדרה בסעיף קודם.

הצעות פתרון למורה

תרגילים - פלט

תרגיל ראשון

```
print '***********************************'
print '*  Name: My_Name Familly_Name       *'
print '*  Birthday: 1/1/2000               *'
print '*  School:Alon Grade:7 Class:5      *'
print '*  Favorit subject: Computer Science *'
print '***********************************'
```

תרגיל שני

```
print '+---------------------------------------+'
print '|Exam of 2 digits multiplication exercises |'
print '|Circle the correct answer              |'
print '|               Good luck               |'
print '+---------------------------------------+'
print '\n1) 21*19=','\na.',21*18,'\nb.',21*19,'\nc.',21*20,'\nd.',21*21
print '\n2) 13*15=','\na.',13*15,'\nb.',11*15,'\nc.',17*15,'\nd.',14*15
print '\n3) 33*74=','\na.',32*74,'\nb.',31*74,'\nc.',19*74,'\nd.',33*74
print '\n4) 86*52=','\na.',87*51,'\nb.',87*52,'\nc.',86*52,'\nd.',86*51
```

תרגיל שלישי

```
print '+-------------------------------+'
print '|Exam of English Synonyms       |'
print '|Circle the correct answer      |'
print '|         Good luck             |'
print '+-------------------------------+'
print '\n1) What is the synonym of Vacant'
print '\na. Empty\nb. Full\nc. Regular\nd. Holly'
print '\n2) What is the synonym of Huge'
print '\na. Baby\nb. embrace\nc. Little\nd. Enormous'
print '\n3) What is the synonym of Afraid'
print '\na. Caurege\nb. Scared\nc. Strip\nd. Money'
print '\n4) What is the synonym of Evil'
print '\na. God\nb. Good\nc. Bad\nd. Bed'
```

תרגילים - קלט

תרגיל ראשון

```
my_name = raw_input('Enter your name:')
my_class = raw_input('Enter your class:')
my_school = raw_input('Enter your school:')
print 'Good morning',my_name,'of class',my_class,
     'from',my_school,'school'
```

תרגיל שני

```
year_of_birth = int(raw_input('Enter your birth year:'))
current_year = 2017
print 'Your age is:', current_year-year_of_birth
```

תרגילים – פעולות חשבון עם מספרים

תרגיל ראשון

```
mass = float(raw_input('Enter object mass (kg):'))
height = float(raw_input('Enter object height (m):'))
print 'The object\'s Eh =', 10 * mass * height,  'J'
```

תרגיל שני

```
r = float(raw_input('Enter circle radius:'))
print 'The circumference is:', 2 * 3.14 * r
print 'The area is:', 3.14 * r ** 2
```

<u>תרגיל שלישי</u> (למידת חקר)

הפלט שהתקבל?	פלט צפוי?	קטע קוד
3 3.3333333333 3.3333333333 3.3333333333		a = 10 print a / 3 a = 10.0 print a / 3 a = 10 print a / 3.0 a = 10.0 print a / 3.0

הכלל:

בפעולת חשבון בה לפחות ערך אחד מהאופרנדים עשרוני תינתן תוצאה עשרונית.

בפעולת חשבון בין אופרנדים שלמים תינתן תוצאה שהיא שלם (ללא ערך השארית).

הפלט שהתקבל?	פלט צפוי?	קטע קוד
  20 WoWWoW		**a = 10** **b = 'WoW'** print **a + a** print **b + b**

הכלל:

בפעולת חיבור של משתנים מספריים ניתן סכומם.

בפעולת חיבור של מחרוזות משרשרת את המחרוזת למחרוזת אחת המרכיבה את כל המחרוזות על פי סדר הופעתן בפעולת החיבור.

קטע קוד	פלט צפוי?	הפלט שהתקבל
a = 10 b = 10.0 c = ’big’ print a * 3 print b * 3 print c * 3		30 30.0 Bigbigbig
הכלל: בפעולה כפל של משתנים מספריים תינתן תוצאת מכפלתם. בפעולת כפל של מספר במחרוזת תשורשר המחרוזת לעצמה כמספר הפעמים של המכפיל.		

קטע קוד	פלט צפוי?	הפלט שהתקבל
a = 'big' **print a * 0**		**(מחרוזת ריקה)**
הכלל: הכפלת מחרוזת ב-0 יוצר מחרוזת ריקה.		

קטע קוד	פלט צפוי?	הפלט שהתקבל
a = 'big' **b = 'small'** **print a * True + b * False** **a = 5** **b = 6** **print a * True + b * False** **print True + 1** **print False + 1**		big **5** **2** **1**
הכלל: ערכו של True שווה ל-1 וערכו של False שווה ל-0.		

תרגיל רביעי

לפניך שני קטעי קוד. יש להריץ כל קטע ולהשלים את הפלט המתקבל.

קטע קוד	הפלט שהתקבל?

`name = 'Rachel'`
`age = 16`
`print '%s %d' % (name, number)`
	Rachel 16

`name = 'Rachel'` `print "%s is my best friend!"%(name)`	Rachel is my best friend!
`diff = 15` `print "My older brother is %d years older."%(diff)`	My older brother is 15 years old.

תרגיל חמישי (אתגר)

```
num = int(raw_input('Enter an integer number:'))
print (num + 1) % 2 * 'even' + num % 2 * 'odd'
```

תרגילים – פרוק מספר לספרות

תרגיל ראשון

```
num = int(raw_input('Enter 2 digits integer number:'))
units = num % 10
tens = num / 10
print 'The digits sum is:', units + tens
print 'The product of the digits :', units * tens
```

תרגיל שני

```
num = int(raw_input('Enter 3 digits integer number:'))
units = num % 10
print digit, # שימו לב כי לפסיק בסוף הפעולה
tens = num / 10 % 10
print tens,
hund = num / 10
print hund
```

תרגיל שלישי

3. קלוט מספר שלם למשתנה num
4. כל עוד num > 0
   4.1 הדפס num % 10
   4.2 num ← num / 10

פתרון:

```
num = int(raw_input('Enter integer number:'))
```

```
while (num > 0):
   print num % 10
   num = num + 10
```

5. הצב 0 במשתנה res
6. קלוט מספר טבעי (שלם חיובי) למשתנה num
7. כל עוד num > 0
   7.1 num ← res + num % 10
   7.2 num ← num // 10
8. הדפס res

פתרון:

```
res = 0

num = int(raw_input('Enter a positive integer:'))
```

```
while (num > 0):
   res = res + num % 10
   num = num // 10
print res
```

ביטויים לוגיים פשוטים וביצוע חוזר מותנה

תרגיל ראשון

	הפלט - תוצאת הביטוי הלוגי	הביטוי הלוגי
.1	True	result = 10 >= 10 print result
.2	False	result = 3.0 < 3 print result
.3	False	result = 2**2 != 2*2 print result
.4	False	result = 'a' == 'A' print result
.5	False	result = 10 != 10 print result
.6	True	a , b = 5 , 3 print a > b
.7	True	a = 'number' b = 'number' print a==b
.8	True	a = 5 print a + ____ > 8
.9	True	a , b = _____, 3 print a + b > 8
.10	False	result = _____ != 10 print result
.11	False	a = 'number' b = __________ print a==b
.12	False	a , b = 5 , 3 print a + b > 8
.13	True	a , b = ___ , ____ print a + b > ____
.14	False	result = False < True print result
.15	1	result = True * True print result
.16	0	result = True * False

	הפלט - תוצאת הביטוי הלוגי	הביטוי הלוגי
		print result

תרגילים – ביצוע חוזר מותנה (while)

תרגיל ראשון

```
num = int(raw_input('Enter an integer number:'))
while num > 0:
  print num % 10
  num=num / 10
print 'End of job'
```

תרגיל שני

א.

```
num = int(raw_input('Enter an integer number:'))
sum = 0
while num > 0:
  sum += num % 10
  num = num /10
print 'Sum = ', sum
print 'End of job'
```

ב.

```
num = int(raw_input('Enter an integer number:'))
while num > 0:
  print num % 10,
  num = num / 10
print
print 'End of job'
```

תרגיל שני

```
num = int(raw_input('Enter an integer number:'))
sum = 0
while num > 0:
  if (num % 2 == 0):
      sum += num % 10
  num = num /10
```

```
print 'Sum = ', sum
print 'End of job'
```

<u>תרגיל שלישי</u>

```
num = int(raw_input('Enter an integer number:'))
count = 0
number_of_even_digits = 0
while num > 0:
   count += 1
   if (num % 2 == 0):
         number_of_even_digits += 1
   num = num /10
print 'Even = ', number_of_even_digits,
# Emphasize comma at the end of the command
print 'Odd = ', count - number_of_even_digits
```

<u>תרגיל רביעי</u>

```
digit = raw_input('Enter digit: ')
count,num = 0,0
while digit <> '':
   num = num + (int(digit) * (10**count))
   count+= 1
   digit = raw_input('Enter digit: ')
print 'The number is: ',num
```

<u>תרגיל חמישי</u>

(שאלה זו בהתאם לתכנית הלימודים מבוא לסייבר באמצעות שפת Python - מימוש המרות בסיסים)

א.

```
num = int(raw_input('Enter binary number:'))
sum, count = 0, 0
while num > 0:
   sum = sum + num % 10 * 2 ** count
   num = num / 10
print 'The value in base 10 is :', sum
```

ב.

```
num = int(raw_input('Enter number:'))
base = int(raw_input('Enter the number's base:'))
sum, count = 0, 0
while num > 0:
  sum = sum + num % 10 * base ** count
  num = num / 10
print 'The value in base 10 is :', sum
```

ג.

```
num10 = int(raw_input('Enter an integer number:'))
base = int(raw_input('Enter the target base:'))
while num10 > 0:
  baseDigit = num10 % base
  print baseDigit
  num10 = num10 / base
print 'in base', base
```

ד.

```
num10 = int(raw_input('Enter an integer number:'))
base = int(raw_input('Enter the number base:'))
print num10,' in base 10 is:'
numBase = 0
Count = 0
while num10 > 0:
  baseDigit = num10 % base
  num10 = num10 / base
  numBase = numBase + baseDigit * (10 ** count)
  count = count + 1
print numBase,'in base',base
```

תרגילים – כתיבת פונקציות

תרגיל ראשון

א.

```
def base10_to_anyBase (num10, base):
  # transfer a number in base-10 to number in base base
  numBase=0
  count=0
  while num10 > 0:
    baseDigit = num10 % base
    num10 = num10 / base
    numBase = numBase + baseDigit * (10 ** count)
    count = count + 1
  return numBase
```

ב.

```
def anyBase_to_base10 (numBase, base):
  # transfer a number in base-base to number in base 10
  num10=0
  count=0
  while numBase > 0:
    baseDigit = numBase % 10
    numBase = numBase / 10
    num10 = num10 + baseDigit * (base ** count)
    count = count + 1
  return num10
```

ג.

```
def main ():
  # test program for base conversion functions
  nm , bs = 12321 , 4
  print nm , 'in base', bs, 'is:' ,anyBase_to_base10(nm , bs),
  print 'in base 10'
  nm , bs = 765 , 8
  print nm ,'in base', bs, 'is:' ,anyBase_to_base10(nm , bs),
  print 'in base 10'
  nm , bs = 4321 , 5
  print nm , 'in base', bs, 'is:', anyBase_to_base10(nm , bs),',
  print in base 10'
  nm , bs = 223 , 2
```

```
  print nm , 'in base 10', 'is:', base10_to_anyBase(nm , bs),
  print 'in base', bs
  nm , bs = 999 , 9
  print nm , 'in base 10', 'is:', base10_to_anyBase(nm , bs),
  print 'in base', bs
  nm , bs = 2017 , 7
  print nm , 'in base 10', 'is:', base10_to_anyBase(nm , bs),
  print 'in base',bs
if __name__=='__main__':
  main()
```

ד.

```
def main ():
  # print 10 base number presentation for each base 2-9
  nm = int(raw_input('Enter an integer number:'))
  bs = 9
  while bs > 1:
    print nm,'in base 10','is:',
    print base10_to_anyBase (nm , bs),
    print 'in base', bs
    bs = bs - 1

if __name__=='__main__':
   main()
```

מבוא למחרוזות

תרגיל ראשון

לפניך שורת הקוד הבאה:

st = ′ILovePython′

0	1	2	3	4	5	6	7	8	9	10
I	**L**	**O**	**V**	**e**	**P**	**y**	**t**	**h**	**O**	**N**
-11	-10	-9	-8	-7	-6	-5	-4	-3	-2	-1

לכל שורת קוד יש להגדיר את הפלט המתקבל:

	שורות הקוד	פלט
1.	`s = st[0:2] + v[5]` `print s`	ILP
2.	`s = st[1:5]` `print s`	Love
3.	`s = st[-10:-6]` `print s`	Love
4.	`s = st[-6:-4]` `print s`	Py
5.	`s = st[-11:-6] + ′Me′` `print s`	ILoveMe

תרגיל שני

לפניך שורת הקוד הבאה:

st = ′StudyingPython′

לכל שורת קוד יש להגדיר את הפלט המתקבל:

	שורות הקוד	פלט
1.	`s = st[::]` `print s`	StudyingPython
2.	`s = st[::-1]` `print s`	nohtyPgniydutS
3.	`s = st[::2]` `print s`	SuynPto
4.	`s = st[::-2]` `print s`	Nhygidt
5.	`s = st[:5]` `print s`	Study
6.	`s = st[:-6]` `print s`	Studying

	שורות הקוד	פלט
7.	s = st[8:] print s	Python
8.	s = st[8::2] print s	Pto
9.	s = st[8::-2] print s	PnyuS
10.	s = st[:9-1] print s	Studying

תרגיל שלישי

```
def round_string (st):
  temp = st[1:] + st[0]
  while st != temp:
    print temp
    temp = temp[1:] + temp[0]
  print temp
```

תרגיל רביעי

א.

```
def reverse_text (txt):
  # revers the order of text,
  # and capitalize the 1st and last letters
  s= txt[::-1]
  s=s[0].upper()+s[1:-1]+ s[-1].upper()
  return s
```

ב.

```
def main ():
  # ask repeatedly for text, print it using reverse_text
  # until stop text is entered
  text = raw_input('Enter text:')
  while text != 'stop':
    print reverse_text(text)
    text = raw_input('Enter text:')

if __name__=='__main__':
   main()
```

תרגיל חמישי

```
def main ():
  # ask repeatedly for text, print its 1st half
  # then prints the other half in capital letters
  text = raw_input('Enter text:')
  while text != 'done':
    print text[:len(text)/2]
    print text[len(text)/2:].upper()
    text = raw_input('Enter text:')

if __name__=='__main__':
   main()
```

תרגיל שישי

א.

```
def base10_to_anyBase (num10, base):
  # transfer a number in base-10 to number in base base
  # base can any number between 2 - 16
  up2hex = '0123456789ABCDEF'
  numBase = ''
  count = 0
  while num10 > 0:
    baseDigit = num10 % base
    num10 = num10 / base
    numBase = up2hex[baseDigit:(baseDigit+1)] + numBase
  return numBase
```

ב.

```
def main ():
  # print 10 base number presentation for each base 2-16
  nm=int(raw_input('Enter an integer number:'))
  bs=16
  while bs > 1:
    print nm,'in base 10','is:',base10_to_anyBase(nm, bs),
    print 'in base',bs
    bs= bs - 1

if __name__=='__main__':
   main()
```

תרגילים – For ומחרוזות

תרגיל ראשון

```
def sum_string_number (st):
  sum = 0
  for x in st:
     sum += int(x)
  return sum
```

תרגיל שני

לפניך מספר קטעי קוד. לכל קטע יש להגדיר את הפלט :

	שורות הקוד	פלט
1.	st = 'abcde' for x in st: print st.index(x)	0 1 2 3 4
2.	st = 'abcde' for x in st: print x, x.upper(x)	a A b B c C d D e E
3.	st = 'aBcdE' for x in st: print x.islower()	True False True True False
4.	st = 'abcde' for x in st[2:]: print x	c d e
5.	st = 'abcde' for x in st[::-1]: print x	E d c b a

תרגיל שלישי

א.

```
def anyBase_to_base10 (numBase, base):
  # transfer a number in base-base (base can be 2 to 16)
  # to a number in base 10
  up2hex = '0123456789ABCDEF'
  num10 = 0
  numBase = numBase.upper()
  for x in numBase:
    num10 = num10 * base + Up2hex.index(x)
  return num10
```

ב.

```
def main ():
  # print the highest 4 digit number for each base 2 - 16
  # and transfer it to a number in base 10
  bases = '123456789abcdef'
  for base in bases:
    baseNum = bases.index(base) + 2
    print base * 4,
    print 'in base',
    print baseNum,
    print 'is',
    print anyBase_to_base10 (base * 4, baseNum)

if __name__=='__main__':
   main()
```

תרגילים – תנאים

תרגיל ראשון

```
color = raw_input('Enter the pedestrian crossing light
color: ')
if color.upper() == 'GREEN':
  print'Pass'
else:
  print'Stop'
```

תרגיל שני

```
color = raw_input('Enter the pedestrian crossing light
color: ')
if color.upper() == 'GREEN':
  print'Go'
elif color.upper() == 'YELLOW':
      print 'Proceed with caution'
elif color.upper() == 'RED':
      print 'Stop'
else:
     print'Error'
```

תרגיל שלישי

```
mark = int(raw_input('What is your mark: '))
if mark == 100:
  print' Excellent'
elif  mark >= 90:
  print' Very good'
elif  mark >= 80:
  print' Good'
elif  mark >= 60:
  print' You should do better'
else:
  print' You should relearn'
```

תרגיל רביעי

```
def plat_with_string_number (st):
   sum = 0
   for x in st:
      if int(x) % 2 == 0
          print x,
      bs= bs - 1

if __name__=='__main__':
   main()
```

תרגיל חמישי

```
def check_num (numBase, base):
  # check numBase number to be valid in its base
  up2hex = '0123456789ABCDEF'
  result = True
  numBase = numBase.upper()
  for x in numBase:
    # 1st check if it is a valid digit
    if x not in up2hex:
      result = False
    # 2nd check if it value is less than the base
    elif up2hex.index(x) > (base-1):
      result = False
  return result

def main ():
  # ask for base (2-16) and valid number at this base
  # if invalid, notify, and ask again for valid number
  b = int(raw_input('Enter counting base (2-16):'))
  n = raw_input('Enter integer in that base:')
  while check_num (n,b)==False:
    print n, 'is not a valid number in base', b
    n = raw_input('Enter integer in that base:')

if __name__=='__main__':
   main()
```

תרגילים – תנאים מורכבים

תרגיל ראשון

```
year = int(raw_input('Enter a year: '))
if ((year%4==0) and not(year%100==0)) or (year%400==0):
  print year,'is a Leap year'
else:
  print year,' is not a Leap year'
```

תרגיל שני

א.

```
def check_base (base):
  # check if base is in 2-16 range/
  if base >= 2 and 16 >= base:
    return True
  else:
     return False
```

או לחלופין

```
def check_base (base):
  # check if base is in 2-16 range/
   return base >= 2 and 16 >= base
```

ב.

```
def check_base (base):
  # check if base is in 2-16 range/
  if base >= 2 and 16 >= base:
    return True
  else:
    return False

def check_num (numBase, base):
  # check numBase number to be valid in its base
  up2hex = '0123456789ABCDEF'
  result = True
  numBase = numBase.upper()
  for x in numBase:
    # 1st check if it is a valid digit
    if x not in up2hex:
      result=False
    # 2nd check if it value is less than the base
    elif up2hex.index(x) > (base-1):
      result = False
  return result

def main ():
  # ask for base (2-16) and number at this base
  # if input is invalid, notify, and ask again
  B = int(raw_input('Enter counting base (2-16):'))
  while not check_base (b):
    print b, 'is not a valid base'
    b = int(raw_input('Enter counting base (2-16):'))
  n = raw_input('Enter integer in that base:')
  while check_num (n,b) == False:
    print n, 'is not a valid number in base', b
    n = raw_input('Enter integer in that base:')

if __name__=='__main__':
   main()
```

מבחן במדמ"ח לכיתה ט' – מחצית א'

הנחיות:

- משך הבחינה 60 דקות (75 דקות להארכת זמן).
- יש לקרוא היטב את כל השאלה ולענות בגוף הבחינה.
- בצד כל שאלה מוגדר הניקוד לתשובה נכונה. בבחינה ניתן לצבור עד 110 נק'. הציון האפשרי הגבוה ביותר 100.
- כל חומר עזר אישי ניתן לשימוש.

בהצלחה, עופר דיין

<u>שאלה ראשונה</u> (12 נק')

א. לפניך 8 מספרים בבסיסים שונים. יש להקיף בעיגול את המספרים **הלא תקינים**, לדוגמה 231_2 שכן הספרות 2 ו- 3 אינן מופיעות בבסיס 2:

א.	8192_{10}
ב.	$A9H7F_{16}$
ג.	$F0E1C2_{16}$
ד.	217_8
ה.	12_{12}
ו.	3210_4
ז.	49_5
ח.	101000_2

ב. בין כל זוג מספרים בסעיפים הבאים יש להגדיר את הסימן המתאים גדול, קטן או שווה (> , < , =):

א.	12_{10}	16_8
ב.	27_{16}	27_8
ג.	$1A_{16}$	32_8
ד.	50_8	101000_2

<u>שאלה שנייה</u> *(16 נק')*

לפניך סדרות בבסיסים שונים.

לכל סדרה, בהתאם לבסיס, יש להוסיף את שני האברים העוקבים. לפני הוספת האברים יש למצוא את החוקיות של האברים בבסיס הנתון.

א. $172_8, 174_8, 176_8$, ___$_8$, ___$_8$

ב. $9B_{16}, 9C_{16}, 9D_{16}, 9E_{16}$, ___$_{16}$, ___$_{16}$.

ג. $10000_2, 10011_2, 10110_2$, ___$_2$, ___$_2$

ד. $0_4, 3_4, 12_4, 21_4$, ___$_4$, ___$_4$

שאלה שלישית (12 נק׳)

לפניך טבלה חלקית המכילה מספרים בשלושה בסיסים – בסיס בינארי, בסיס אוקטלי ובסיס הקסדצימלי. יש להשלים את הטבלה כך שערך המספר יופיע בשלושת הבסיסים, בהתאמה.

בסיס בינארי (2)	**בסיס אוקטלי (8)**	**בסיס הקסדצימלי (16)**
1101001011		
	3654	
		201

שאלה רביעית (8 נק׳)

לפניך שורת קוד המקבלת כקלט מספר בינרי (לדוגמה 10110) ומאכסנת אותו במשתנה num :

num = int(raw_input("Enter binary number"))

יש להקיף בעיגול אלו מהתנאים הבאים מזהים האם הקלט מתחלק ב-4 ללא שארית ?

	תנאי
א.	num % 4 == 0
ב.	num % 25 == 0
ג.	num % 100 == 0
ד.	num / 10 == 0 **and** num % 10 ==0
ה.	num / 10 == 0 **or** num % 10 ==0
ו.	num / 10 == 0 and num / 10 % 10 ==0

שאלה חמישית (9 נק׳)

לפניך שורת קוד :

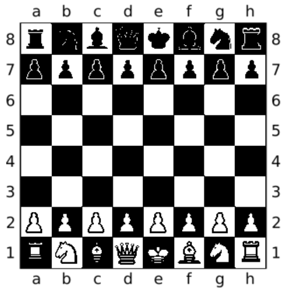
x = 'I know python'

לכל אחת מפקודות הפלט הבאות יש להגדיר את הפלט שיוצג על המסך :

	פקודה	פלט
א.	print len(x)	
ב.	print x[3: 7]	
ג.	print x[10:]	

ד.	print len(x[: 6])	
ה.	print len(x[-5:])	
ו.	print x[: 10: -1]	

שאלה שישית (18 נק')



משחק שחמט מיוצג ע"י לוח 8x8 בו השורות מסומנות בספרות מ-1 ועד 8 והטורים מסומנים באותיות a עד h.

נתונה פונקציה **b_king**(x,y) המקבלת כפרמטר שתי קואורדינאטות על הלוח, שורה ועמודה, x ו-y בהתאמה ומחזירה true אם בקואורדינטות אלו ניצב מלך שחור. בכל מצב אחר יוחזר false.

לפניך קטע תכנית חלקי אשר מטרתו לחפש ולהדפיס את מיקום המלך על לוח השחמט.

לדוגמה, עבור לוח השחמט שבתמונה יודפס e8.

בקטע יש שימוש בפונקציית **b_king**. יש להשלים את החלקים החסרים בקטע התכנית.

```
columns = 'abcdefgh'
x_count = ________
while x_count < ________ :
  x_label = columns[x_count]
  x_count += 1
  y_count = ________
  while y_count  < ________ :
    y_count += 1
    print (x_label + str(y_count))* bking(_______ ,
_______)
```

שאלה שביעית (13 נק')

במתמטיקה, **עֲצֶרֶת** היא מכפלת כל המספרים הטבעיים מ-1 ועד למספר נתון (כולל).

למשל, "4 עצרת" היא המכפלה $1 \times 2 \times 3 \times 4 = 24$

מה הערך של "5 עצרת" (ניתן לרשום גם 5!)? ________________________________

יש לכתוב קטע תכנית הקולט מספר שלם n. על קטע התכנית לחשב ולהדפיס את n עצרת (n!).

לדוגמה, אם נקלט המספר 4 יש להדפיס

4! = 24

הנחה: הקלט תקין.

<u>שאלה שמינית</u> (10 עד 15 נק')

תמר כתבה פונקציה בשם cypher המקבלת כפרמטר מחרוזת ומחזירה מחרוזת מוצפנת.

בפונקציה מוגדרות שתי מחרוזות, האחת abc מכילה את כל אותיות האלף-בית לפי הסדר, והשנייה bca מכילה את כל אותיות האלף-בית כשהן מעורבבות.

הפונקציה סורקת את המחרוזת תו אחרי תו. לכל תו מוצאת הפונקציה את מקומו במחרוזת abc ושולפת את התו המקביל לו במחרוזת bca. זהו התו המוצפן.

את התו המוצפן היא מצרפת למחרוזת חדשה המכילה את הטקסט המוצפן.

להלן הפונקציה שתמר כתבה:

```
def cypher (txt):          #הפונקציה מקבלת  טקסט להצפנה בתוך מחרוזת
   abc = 'abcdefghijklmnopqrstuvwxyz'
   bca = 'bcaefdhigkljnomprsquvtxywz'
   count = 0
   new_txt=''
   while count < len(txt):
      char = txt[count]                      #הוצאת תו אחד מטקסט
      k = abc.index(char)                    # מציאת מיקום תו
      coded_char = bca[k]                    #הוצאת התו המקביל לו
      new_txt = new_txt + coded_char         #צירוף התו המקודד להודעה המוצפנת
      count = count + 1
   return new_txt          #החזרת מחרוזת מוצפנת
```

לבדיקת הפונקציה הגדירה וזימנה תמר את שורת הקוד הבאה:

```
print cypher ("arrive at noon")
```

התקבלה הודעת שגיאה. מה הסיבה? נימוק.

איזה שינוי יש לבצע בקטע התכנית כדי שהפונקציה תחזיר מחרוזת מוצפנת תקינה? ניתן להסביר בקצרה או לרשום שורות קוד ולתעד אותן.

סעיף בונוס:

לאחר התיקון תמר הריצה את התכנית וקיבלה את הפלט הבא:

```
bssgtf bu ommo
```

לצורך פיענוח ובדיקת הקוד המוצפן שהתקבל, תמר שכפלה את הפונקציה ונתנה לעותק את השם החדש:

```
decipher
```

תמר זימנה את הפונקציה באמצעות שורת הקוד הבאה:

```
print decipher ("bssgtf bu ommo")
```

האם התקבלה ההודעה המקורית? אם כן יש לנמק, אחרת יש להגדיר את השינוי הנדרש בפונקציה כדי שתפענח את הקוד המוצפן?

הצעות פתרון

שאלה ראשונה *(12 נק')*

א.	8192_{10}
ב.	$A9H7F_{16}$
ג.	$F0E1C2_{16}$
ד.	217_8
ה.	12_{12}
ו.	3210_4
ז.	49_5
ח.	101000_2

16_8	>	12_{10}	א.
27_8	<	27_{16}	ב.
32_8	=	$1A_{16}$	ג.
101000_2	=	50_8	ד.

שאלה שנייה *(16 נק')*

ה. $172_8, 174_8, 176_8,$ **200**$_8$, **202**$_8$

ו. $9B_{16},\ 9C_{16},\ 9D_{16},\ 9E_{16},$ **9F**$_{16}$, **A0**$_{16}$.

ז. $10000_2, 10011_2, 10110_2,$ **11001**$_2$, **11100**$_2$

ח. $0_4, 3_4, 12_4, 21_4,$ **30**$_4$, **33**$_4$

שאלה שלישית (12 נק')

בסיס הקסדצימלי (16)	בסיס אוקטלי (8)	בסיס בינארי (2)
34B	**1513**	**1101001011**
7AC	**3654**	**3654**
201	**1000000001**	**1001**

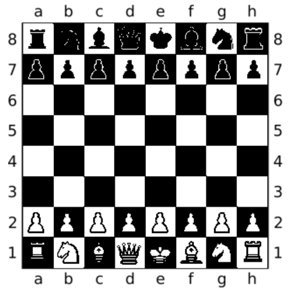
שאלה רביעית (8 נק')

תנאי	
num % 4 == 0	א.
num % 25 == 0	ב.
num % 100 == 0	ג.
num / 10 == 0 **and** num % 10 ==0	ד.
num / 10 == 0 **or** num % 10 ==0	ה.
num / 10 == 0 and num / 10 % 10 ==0	ו.

שאלה חמישית (9 נק')

א.	print len(x)	**13**
ב.	print x[3: 7]	**Now**
ג.	print x[10:]	**Hon**
ד.	print len(x[: 6])	**6**
ה.	print len(x[-5:])	**5**
ו.	print x[: 10: -1]	**no**

שאלה שישית (18 נק')



משחק שחמט מיוצג ע"י לוח 8x8 בו השורות מסומנות בספרות מ-1 ועד 8 והטורים מסומנים באותיות a עד h.

נתונה פונקציה **b_king**(x,y) המקבלת כפרמטר שתי קואורדינטות על הלוח, שורה ועמודה, x ו-y בהתאמה ומחזירה true אם בקואורדינטות אלו ניצב מלך שחור. בכל מצב אחר יוחזר false.

לפניך קטע תכנית חלקי אשר מטרתו לחפש ולהדפיס את מיקום המלך על לוח השחמט.

לדוגמה, עבור לוח השחמט שבתמונה יודפס e8.

בקטע יש שימוש בפונקציית **b_king**. יש להשלים את החלקים החסרים בקטע התכנית.

```
columns = 'abcdefgh'
x_count = 0
while x_count < 8 : # or len(columns)
  x_label = columns[x_count]
  x_count += 1
  y_count = 0
  while y_count  < < 8 : # or len(columns)
    y_count += 1
    print (x_label + str(y_count))* b_king (x_count,
y_count)
```

שאלה שביעית (13 נקי)

במתמטיקה, **עֲצֶרֶת** היא מכפלת כל המספרים הטבעיים מ-1 ועד למספר נתון.
למשל, "4 עצרת" היא המכפלה $1 \times 2 \times 3 \times 4 = 24$

מה ערך "5 עצרת" (ניתן לרשום גם !5)? **120**

יש לכתוב קטע תכנית הקולט מספר שלם n. על קטע התכנית לחשב ולהדפיס את n עצרת (!n).

לדוגמה, אם נקלט המספר 4 יש להדפיס

4! = 24

הנחה : הקלט תקין.

```
num = int(raw_input('Enter integer number'))
print str(num) + '! =',
factorial = 1
while num > 0:
     factorial = factorial * num
     num -= 1
print  factorial
```

שאלה שמינית (10 עד 15 נק')

להלן הפונקציה שתמר כתבה:

```
def cypher (txt):          #הפונקציה מקבלת  טקסט להצפנה בתוך מחרוזת
   abc = 'abcdefghijklmnopqrstuvwxyz'
   bca = 'bcaefdhigkljnomprsquvtxywz'
   count = 0
   new_txt=''
   while count < len(txt):
      char = txt[count]                  #הוצאת תו אחד מטקסט
      k = abc.index(char)                # מציאת מיקום תו
      coded_char = bca[k]                #הוצאת התו המקביל לו
      new_txt = new_txt + coded_char     #צירוף התו המקודד להודעה המוצפנת
      count = count + 1
   return new_txt         #החזרת מחרוזת מוצפנת
```

לבדיקת הפונקציה הגדירה וזימנה תמר את שורת הקוד הבאה:

```
print cypher ("arrive at noon")
```

התקבלה הודעת שגיאה. מה הסיבה? נימוק.

בטקסט יש תווי רווח, שהפונקציה לא מטפלת בהם.

איזה שינוי יש לבצע בקטע התכנית כדי שהפונקציה תחזיר מחרוזת מוצפנת תקינה? ניתן להסביר בקצרה או לרשום שורות קוד ולתעד אותן.

יש להוסיף את תו הרווח בשתי מחרוזות האלפבית, ובאותו מקום. למשל:

```
abc='abcdefghijklmnopqrstuvwxyz '
bca='bcaefdhigkljnomprsquvtxywz '
```

סעיף בונוס:

לאחר התיקון תמר הריצה את התכנית וקיבלה את הפלט הבא:

```
bssgtf bu ommo
```

לצורך פיענוח ובדיקת הקוד המוצפן שהתקבל, תמר שכפלה את הפונקציה ונתנה לעותק את השם החדש: `decipher`

תמר זימנה את הפונקציה באמצעות שורת הקוד הבאה:

```
print decipher ("bssgtf bu ommo")
```

האם התקבלה ההודעה המקורית? אם כן יש לנמק, אחרת יש להגדיר את השינוי הנדרש בפונקציה כדי שתפענח את הקוד המוצפן?

כדי לפענח יש להחליף את מחרוזות האלפבית:

```
bca='abcdefghijklmnopqrstuvwxyz'
abc='bcaefdhigkljnomprsquvtxywz'
```