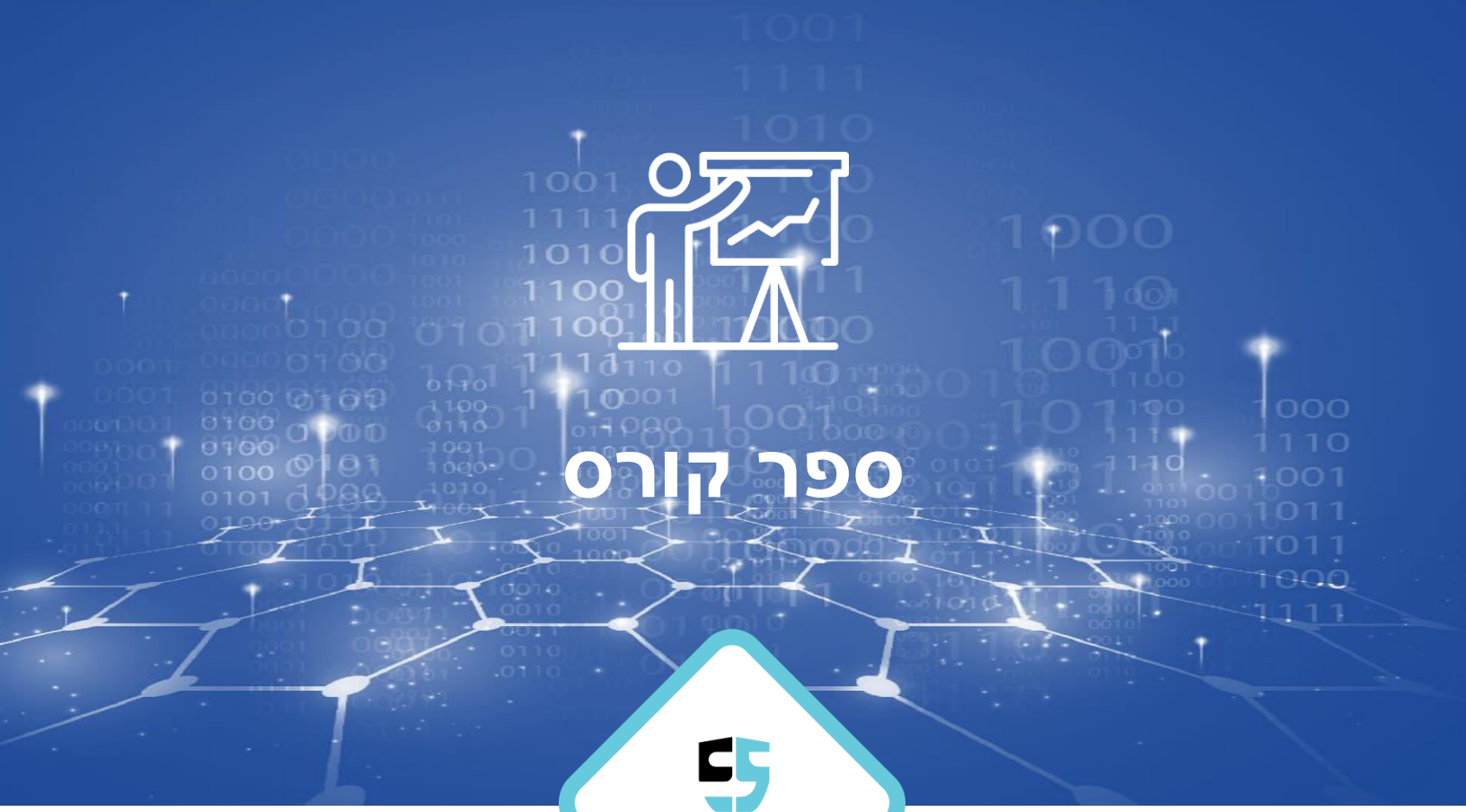

ספר קורס

CYBER SCHOOL

"Recursion (רקורסיה)"

CYBER SCHOOL

מידע על ספר לימוד זה

ספר לימוד זה נועד לסייע לכם ללמוד ולסקור את החומר הנלמד במהלך הקורס מבוא לפייתון. הוא מכיל מידע, הרלוונטי לשיעורים הנלמדים בכיתה, וכן הפניות לפעילויות מעבדה שכדאי לתרגל כדי להגביר את רמת הידע שלכם.

התוכן בספר לימוד זה כולל יסודות תכנות וכיצד לעבוד עם שפת התכנות פייתון.

מקרא

בלוקים של טקסט צבעוני עשויים להופיע לאורך ספר הלימוד כדי להפנות את הקוראים למקור ספציפי או להעשיר אותם במידע נוסף.

הבלוקים של הטקסט כוללים:

משימת מעבדה
זהו זמן טוב לבחון את קבצי המעבדה הרלוונטיים (על פי קוד המעבדה) ולתרגל את מה שנלמד עד כה.

טוב לדעת
מידע נוסף או תובנות לגבי הנושא. מידע זה נועד לצורך העשרה בלבד ואינו חלק מהחומרים עליהם תיבחנו.

טיפ
מידע שימושי שיש בו כדי לסייע לתלמידים ללמוד את החומר או לעבוד עם כלים מסוימים.

מידע נוסף
קישורים או הפניות לחומר חיצוני שניתן להשתמש בהם כדי להרחיב את הידיעות שלכם לגבי הנושא.

Recursion

מה זה Recursion?

רקורסיה (Recursion) היא מושג שבו פונקציה קוראת לעצמה בתהליך הביצוע שלה כדי לבצע פעולה חוזרת.

רקורסיה (Recursion) מספקת דרך פשוטה ויעילה לביצוע לולאות נתונים, אך יש להשתמש בה בזהירות על מנת למנוע לולאות אינסופיות.

מידע נוסף

לקבלת מידע נוסף על פונקציה רקורסיבית (Recursion) יש לגשת לדף האינטרנט:
https://medium.com/better-programming/when-to-loop-when-to-recurse-b786ad8977de

מימוש Recurstion

מימוש Recursion כאשר פונקציה קוראת לעצמה מספר פעמים.

הדוגמה הבאה מראה איך מממשים Recursion:

```
count = 0

def recur(count):
   if count == 10:
      return

   print("*" * count)
   count += 1
   recur(count)

recur(count)
```

```
==========================================================================
*
**
***
****
```

```
*****
******
*******
********
*********
```

בדוגמה שלמעלה פעולת ה-Recursion מבוצעת על מנת להדפיס קבוצת כוכביות (*) עבור המשתמש ולסיים את הביצוע כאשר תנאי מתקיים.

משימת מעבדה: חיפוש רקורסיבי (Recursive)
יש ליישם את הרעיון של רקורסיה כדי להדפיס ערכים מרשימה מקוננת.

משימת מעבדה: הדפסת רשימת ספריות
יש ליישם רקורסיה ולהשתמש בפונקציות שונות של ספריית OS.

תכנות מונחה עצמים

מהו תכנות מונחה עצמים?

תכנות מונחה עצמים מתייחס ל"אובייקטים" ("objects") המכילים נתונים בצורה של פונקציות ותכונות.

השיטה מאפשרת גמישות בפיתוח תוכניות בשל השימוש שלה ביצירת סוגי נתונים מותאמים אישית הכוללים גם תכונות וגם מפרטי התנהגות.

טיפ
כאשר הקוד הוא מודולרי, הרבה יותר קל ליישם שינויים, מבלי לשנות את כל התוכנית.

מידע נוסף
לקבלת מידע נוסף על תכנות מונחה עצמים יש לגשת לדפי האינטרנט:
https://realpython.com/python3-object-oriented-programming/

Class

class היא המבנה המוגדר מראש של אובייקט, ונחשבת ל"שרטוט" ("blueprint") שלו. class מגדירה איך אובייקט ייראה על סמך המאפיינים והתכונות האופציונליים שלו.

כאשר אובייקט מוגדר ומופעלת תוכנית הכוללת את האובייקט, התוכנה משתמשת בclass כדי לבנות את האובייקט על סמך הערכים שהוקצו לו, ושומרת את המופע שלו.

לדוגמה, מחלקה יכולה להיות רכב, והתכונות שלו עשויות להיות משתנים, כמו מספר הגלגלים, מספר הדלתות, צבע וכו'. הפונקציונליות שלה עשויה לכלול "נהיגה", "עצירה", "האצה" וכו'.

הגדרת Class

ה-class מוגדרת על ידי פקודת **class** ומיד אחריה השם שלה. הטווח שלה מוגדר על ידי נקודותיים אחרי השם שלה.

על מנת לבצע אתחול של התכונות שלו בעת יצירת אובייקט מסוג זה, נעשה שימוש בפונקציה **__init__**. פונקציה זו מופעלת ברגע שהאובייקט נוצר והיא הראשונה שמתבצעת.

בתוך הפונקציה, תכונות ה class מוקצות בדרך כלל עם ערכים.

כדי לגשת לתכונות הclass מתוך הclass, יש להשתמש בפקודה **self**, המתייחסת למחלקה הנוכחית.

```
class Car:
    def __init__(self, color, window_number, price):
        self.color = color
        self.window_number = window_number
        self.price = price
```

בדוגמה למעלה,

- מחלקה מוכרזת כ "class", ושמה מתחיל באות גדולה.
- נעשה שימוש ב __init__ על מנת לאתחל את הקריאה לאובייקט.
- מוגדר משתנה חדש על ידי שימוש ב **self.**

יצירת אובייקט

כאשר מגדירים class באמצעות init ומשתנים, המשתנים חייבים לקבל ערכים בעת יצירת אובייקט. אם ניצור אובייקט של מחלקה באמצעות __init__, אנו חייבים לשלוח את כל הערכים הדרושים אל __init__, אחרת תתרחש שגיאה כאשר היא נקראת.

כאשר קוראים לתכונות אובייקט, יש להשתמש בתחביר הבא: **object name.attribute**

הדוגמה הבאה מראה איך ליצור אובייקט:

```
class Car:
   def __init__(self, color, window_number, price):
      self.color = color
      self.window_number = window_number
      self.price = price

car1 = Car("Blue",4,56000)
car2 = Car("Red",2,230000)

print(car1.color)
print(car2.price)

==========================================================================
=
Blue
230000
```

משימת מעבדה: יצירת מכוניות

הטמעת Object Oriented Programming על ידי יצירת מחלקה ואובייקטים שונים.

טיפ

בצד הלקוח, קבלת נתונים מקונפגת תחילה, כך שהשרת יצטרך לשלוח תחילה.

משימת מעבדה: Login

יש ליצור ממשק התחברות (login) באמצעות sockets.

CYBER SCHOOL