בית הספר להנדסה ומדעי המחשב ע"ש רחל וסלים בנין

מבוא למדעי המחשב 67101

תרגיל 5 - השוואת סלי מזון

להגשה בתאריך 14/12/2016 בשעה 22:00

יעדים

התנסות בעבודה עם מבני נתונים בסיסיים ומבני נתונים מורכבים.
כמו כן תתרגלו בתרגיל זה עבודה בזוגות. הוראות מפורטות לגבי הגשה בזוגות תוכלו למצוא בתחתית קובץ זה.

הקדמה

ב- 2015 שונה החוק בישראל. החל משנה זו, כל רשתות השיווק הגדולות מחויבות בפרסום מחירי המזון שלהן. הן מחויבות לפרסם מדי יום את מחירי המוצרים בכל אחת מחנויותיהן וכן את פירוט המבצעים. את קבצי המחירים ניתן למצוא דרך הקישור:
http://economy.gov.il/Trade/ConsumerProtection/Pages/PriceTransparencyRegulations.aspx

פרויקט זה מאפשר להשוות בין סלי המזון השונים ברשתות השונות ובחנויות השונות. בתרגיל זה נפתח כלי להשוואה יעילה.

בתרגיל זה תוכלו לעשות שימוש ב- GUI שספקנו לכם. את ה- GUI ניתן להוריד מאתר הקורס תחת השם ex5_gui.py. ה- GUI (ממשק משתמש גרפי בעברית) מורכב מסט של כפתורים שלחיצה עליהם תקרא לפונקציות שעליכם לממש. כמו כן הוא מכיל שני חלונות עיקריים. חלונות אלו יאפשרו לכם לצפות במוצרי החנויות השונות ובסלי הקניות. שימו לב כי ה- GUI עושה import לקובץ ex5.py - הקובץ בו תיישמו את הפתרון שלכם. כרגע הוא מכיל רק את הגדרות הפונקציות אותם תיישמו כפי שמפורט למטה. מכיוון שהיישום ריק כרגע ה- GUI עובד רק באופן חלקי, ככל שתתקדמו בתרגיל חלקים נוספים של ה GUI יהפכו לזמינים לעבודה עבורכם.

לבינתיים בכדי להכיר את הממשקים השונים של ה- GUI אתם יכולים להריץ את פתרון בית הספר.
הקלידו בטרמינל:

```
~intro2cs/bin/ex5/ex5_gui
```

תוכנית זו מריצה את קובץ ה- GUI עם פתרון שכתבנו אנחנו לתרגיל 5.
המסך השמאלי מייצג את החנויות השונות והמסך הימני את סלי הקניות בינהם תעשו השוואה.
לחצו על הכפתור Load File ובחרו בקובץ store_1.xml. כעת שנו את האינדקס ליד כפתור ה- Load file ל- 2 ואת שם הקובץ וטענו את הקובץ store_2.xml. כעת טענתם שתי חנויות לתוך התוכנית. משמאל למטה תוכלו לצפות במוצרים השונים של החנויות. באמצעות התפריט המספרי וכפתור ה show תוכלו לצפות בחנויות השונות שטענתם.
כעת, סמנו באמצעות העכבר מספר מוצרים ולחצו על הכפתור Add selected. המוצרים יתווספו לסל הקניות מימין. בסל תוכלו לראות את המוצרים שנבחרו ואת מחירם בחנויות השונות. סימון מוצרים נוספים יוסיף גם אותם לסל. תוכלו לרוקנו באמצעות Clear basket.

בית הספר להנדסה ומדעי המחשב ע"ש רחל וסלים בנין

כעת עליכם לפתור בעצמכם את תרגיל 5 על ידי יישום הפונקציות הנתונות ב ex5.py. התקדמו במשימות השונות על פי סדר הופעתן.

משימות

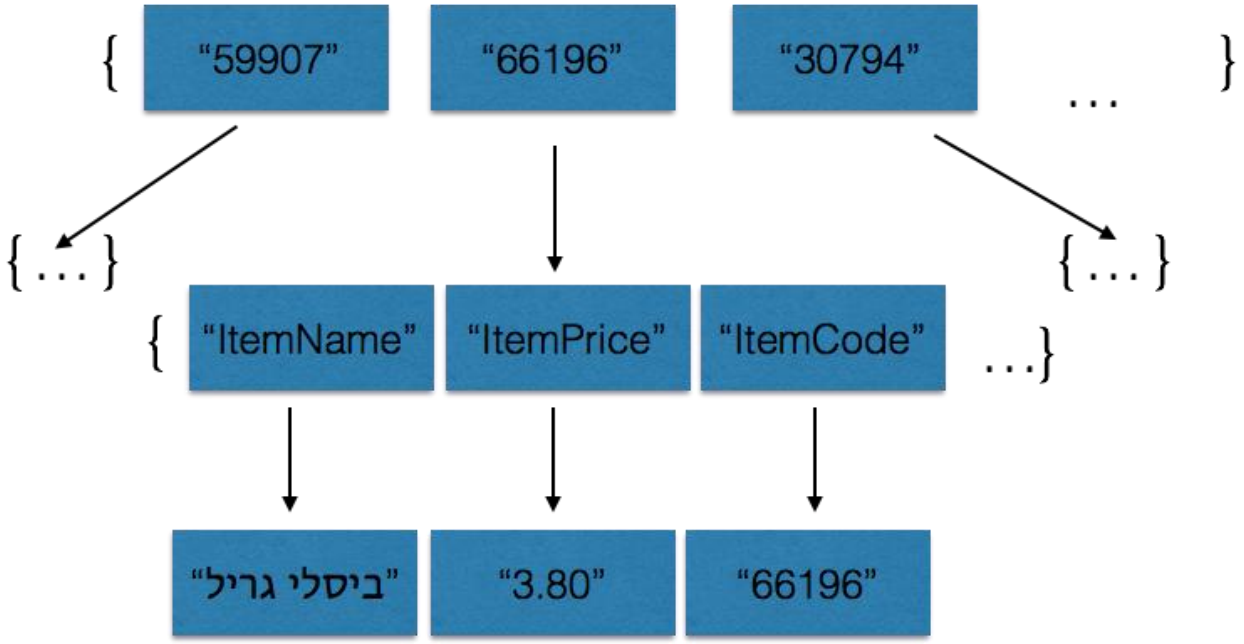
0. יישמנו עבורכם את הפונקציה

```
def get_demo_store():
```

פונקציה זו בונה מבנה נתונים שייצג עבורנו חנות בתכנית.

בהמשך התרגיל תצטרכו לבנות, אתם בעצמכם, חנות שכזו, מתוך קובץ של משרד הכלכלה. אך כעת פונקציה זו יושמה בכדי שנוכל להתחיל בתרגיל. הפונקציה מחזירה צמד: מספר (בטקסט) - המייצג את הID של החנות, וחנות - מבנה נתונים הבנוי ממילון של מילונים. ישום הפונקציה מופיע בקובץ ex5_gui.py.
כל מוצר בחנות ייוצג באמצעות מילון. עבור כל מוצר, הפונקציה מגדירה מילון חדש (מסומן באמצעות סוגריים מסולסלים). המילון הנ"ל יורכב מכל התכונות של המוצר הנ"ל. בין תכונות המוצר נמצא את שמו (ItemName), מחירו (ItemPrice) הקוד שלו (ItemCode) ועוד. המפתח במילון יהיה התגית, למשל 'ItemName' והערך הוא ספציפי עבור כל מוצר. למשל במקרה שלנו אחד המוצרים שמו הוא: "ביסלי גריל".

עד כה, תארנו כל מוצר בחנות באמצעות מילון, אבל כיצד נתאר את החנות כולה (store_db)? נתארה באמצעות מבנה נתונים גדול יותר - מילון של מילונים, כאשר המילון החיצוני יתאר את רשימת המוצרים כולה בחנות. מפתחות המילון החיצוני יהיו ה- ItemCodes של המוצרים השונים, וכל מפתח יצביע על תת המילון הפנימי, כפי שהוגדר בפסקה הקודמת. חנות הדמו שלנו תיוצג על ידי מבנה הנתונים הבא:



כלומר בחנות ישנם מספר מוצרים עם הקודים המופיעים בשורה הראשונה („30794", „66196" וכן הלאה). כל אחד מהמפתחות הנ"ל מצביעים על מילון פנימי. למשל המפתח „66196" מצביע על מילון פנימי המתאר את כל תכונתיו של המוצר שהקוד שלו 66196. מעתה ועד תום התרגיל למבנה נתונים מסוג זה נקרא store_db.
הפונקציה הראשונה שתיישמו תאפשר גישה לפרטי המוצרים הנמצאים בחנות:

1. יישמו את הפונקציה

```
def get_attribute(store_db, ItemCode, tag):
```

המקבלת כאינפוט:
store_db - חנות, מבנה נתונים כפי שהוגדר לעיל.
ItemCode - קוד לאחד ממוצרי החנות (כלומר אחד המפתחות ב- store_db).
tag - אחת התכונות של המוצר (למשל ItemPrice).
ומחזירה את ערך התגית.

למשל עבור חנות הדמו שיצרנו במשימה 0 ,עליכם להחזיר לקריאה:

```
get_attribute(store_db, "59907", "ManufactureCountry")
```

את ארץ הייצור של המוצר שהקוד שלו הוא "59907". במקרה שלנו מוצר זה הוא פיצה משפחתית וארץ הייצור שלו היא ישראל ערך ההחזרה הנכון הוא 'IL'. שימו לב, יתכן כי בהמשך התכנית למוצרים יהיו תכונות נוספות מלבד אלו המופיעות בפונקצית הדמו. על הפונקציה לעבוד עם כל מבנה נתונים שצורתו דומה ל- store_db שהוגדר במשימה 0.

בתרגיל זה ברצוננו להשוות באופן ויזואלי בין החנויות השונות. לצורך זה עלינו להדפיס למסך מוצרים שונים. בכדי לעשות זאת עליכם לבצע המרה ממוצר ל- String. לצערנו ה- GUI שאיתו נעבוד - tkinter , תומך רק באופן חלקי בעברית. בכדי לעזור לו ניישם את הפונקציה הנ"ל:

2. יישמו את הפונקציה

```
def string_item(item):
```

הארגומנט של הפונקציה הוא מוצר - שכזכור מיוצג על ידי מילון, בו המפתחות הן שמות התגיות השונות (למשל ItemCode, ItemName וכו'), והמפתחות מצביעים על הערכים השונים.
ערך ההחזרה של הפונקציה הוא משתנה מסוג String.

עבור מוצר שקבלתם כארגומנט, עליכם להחזיר String בעל מבנה זה (כולל הסוגריים, עם סימן טאב בין הקוד והשם):

```
[ItemCode]\t{ItemName}
```

כאשר בתוך הסוגריים המרובעים יופיע הקוד של המוצר, ובתוך הסוגרים המסולסלות שם המוצר (בעברית).

למשל עבור הקליק מהדוגמא לעיל, ערך ההחזרה של:

```
string_item(store_db["84316"])
```

יהיה (משמאל לימין):

'[84316] {קוקה קולה בקבוק 1.5 ליטר}'

דוגמאות נוספות ניתן לראות בפתרון בית הספר.

3. יישמו את הפונקציה

```
def string_store_items(store_db):
```

היוצרת יצוג טקסטואלי של חנות שלמה.

פונקציה זו מקבלת כארגומנט חנות ומחזירה String (אחד!) המייצג את החנות כולה. הString יורכב מהייצוגים הטקסטואליים של כל אחד מהמוצרים בנפרד (כמו שהוגדרו במשימה 2), כאשר בין כל אחד מהם יהיה סימן ה- \n - break line (ללא תווים נוספים). אתם רשאים להניח שקבלתם כארגומנט מילון אך יתכן שמילון זה יהיה ריק - במידה וכך החזירו String ריק.

כעת אתם יכולים להתחיל להשתמש ב- GUI. הריצו בטרמינל את הפקודה:

```
python3 ex5_gui.py
```

קובץ זה מניח כי לקובץ שלכם קוראים ex5.py. לחצו על הכפתור Load demo. לחיצה עליו תקרא לפונציה get_demo_store ותדפיס את החנות לתוך התיבה השמאלית התחתונה (ה- ID של החנות יודפס מעליה). מיד נלמד כיצד תוכלו בעצמכם ליצור מבנה נתונים המייצג חנות מתוך הקבצים במשרד הכלכלה המייצגים חנויות שונות. לאחר שתעשו זאת תוכלו לטעון חנויות מלאות לתוכנית באמצעות הכפתור Load file.

באתר משרד הכלכלה תמצאו קישור לאתרים שונים עבור רשתות שונות בישראל. בכל אתר כזה, יופיע קבצים המתארים את המחירים לכל חנות. קבצים אלא הינם בפורמט Gz אותו תאלצו לפתוח (למשל על

ידי תוכנת הקוד הפתוח 7-Zip) על מנת לקבל קבצים בפורמט XML עליו למדנו בכיתה. XML הוא פורמט נח בו מידע נתון באופן טקסטואלי ומבנהו דומה למבנה של עץ. הנתונים שמורים בין התגיות השונות. בדומה ל- HTML תגיות יופיעו מאחורי סוגריים משולשים ולאחריהם יופיע התוכן. בסיום תסגר התגית באמצעות סוגריים משולשים ולוכסן. למשל:

```
<first-name>Albert</first-name>
```

מכיוון שמדובר במבנה של עץ ייתכנו תת תגיות כמו:

```
<name>
    <first-name>Albert</first-name>
    <last-name>Einstein</last-name>
  </name>
```

עוד על XML ועל הפורמט של קבצי משרד הכלכלה ניתן לקרוא כאן .
בקצרה ניתן לתאר את קבצי משרד הכלכלה כך:
התגית העיקרית היא שורש העץ.
מתחתיה מופיעות תגיות המגדירות את החנות: ה- ID של החנות (StoreId), מספר המוצרים ופרטים נוספיה לגביה. התגית המשמעותית ביותר עבורנו תהיה תגית ה- Items. תגית זו תשמש שורש לכל מוצרי החנות. מתחת לשורש זה יופיעו כל המוצרים כך שכל מוצר מתחיל בתגית Item ומתחתיה מתואר המוצר. כל מוצר בפני עצמו גם כן ישמש כשורש לתת עץ - תת עץ המגדיר את תכונותיו של המוצר, כאשר גם תכונות אלו יתוארו באמצעות סט של תגיות. למשל בין תכונות המוצר (כלומר תיוגיו השוניו) ניתן למצוא את אותן תכונות שהגדרנו במשימה 0: שם המוצר (ItemName), קוד המוצר (ItemPrice), מחירו (ItemPrice) ועוד תגיות נוספות.
הנחת המוצא שלנו בתרגיל זה תהיה שלכל מוצר קוד מוצר (ItemCode) שונה. עובדה זו תאפשר לנו להשוות בין מוצרים בחנויות שונות. נשים לב כי מבנהו ההיררכי של ה- XML מכוון אותנו לעבודה עם מילונים.

4. יישמו את הפונקציה

```
def read_prices_file(filename):
```

פונקציה זו, מקבלת כארגומנט path לקובץ מסוג XML ועליה לקרוא אותו לתוך מבנה נתונים כפי שהוגדר לעיל במשימה 0.
עליכם להחזיר צמד המורכב מ- String שהוא ה- ID של החנות וממילון של מילונים שייצג את מבנה הנתונים של החנות, שוב בדיוק כמו במשימה 0.

כיצד תעשו זאת? עשו שימוש בספריית xml.etree.ElementTree של פייתון (דוקומנטציה).
שימו לב כי קובץ הטמפלייט שקבלתם כבר מכיל קריאת import לספריה זו.

החלו בשתי הפקודות הבאות:

```
tree = ET.parse(filename)
root = tree.getroot()
```

שימוש בפקודה parse המקבלת כארגומנט כתובת של קובץ תחזיר לכם עץ. קריאה לפונקציה getroot תחזיר לכם את שורש העץ. כעת השורש הוא container המכיל את כל התגיות מתחת לשורש העץ (למשל StoreId או Items). כל תגית שכזו גם היא container לכל התגיות שמתחתיה - כך Items מכילה את כל המוצרים שבתוכה וכן הלאה (כל מוצר מכיל את כל התגיות שלו). כזכור, קל לעבור על האיברים השונים של Containers בפייתון באמצעות לולאת for.

קראו את הקובץ שקבלתם כארגומנט, זהו את ה- ID של החנות וצרו מילון של מילונים בדומה למשימה 0 והחזירו צמד זה.

חזרה ל- GUI: כעת ביכולתכם לטעון חנות שלמה לתוך התכנית שלכם. הורידו קובץ מאתר משרד הלכלה או בחרו באחד הקבצים שזמינים לכם באתר הקורס, שמרו אותם בתיקיית העבודה שלכם. לחצו על הכפתור Load File, ובחרו באחד מקבצים אלו. שימו לב, מימין לכפתור זה מופיע תפריט מספרים. תפריט זה נועד לטעון מספר חנויות לתוך התוכנית. למשל, עברו מ- 1 ל- 2 והזינו חנות נוספת. כעת חנות זו תוצג למסך. כעת עברו בחזרה ל- 1 ולחצו על show - תוכלו לראות בחזרה את החנות הראשונה. באמצעות טעינה של מספר חנויות שונות נוכל להשוות בינהן בהמשך התרגיל. ה- GUI תומך בטעינה של עד 3 חנויות שונות. את שאר הכפתורים נכיר מיד.

5. יישמו את הפונקציה

```
def filter_store(store_db, filter_txt):
```

פונקציה זו תעזור למשתמש לסנן את כמות המוצרים העצומה שיש בחנויות השונות, ולהוסיף לסל הקניות שלו רק את המוצרים המבוקשים.

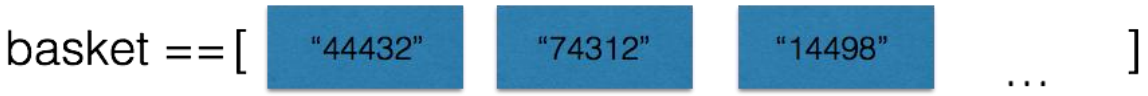
הפונקציה מקבלת כארגומנט חנות אחת (מבנה הנתונים שהוצג לעיל), ו- String.
ה- String יהיה תת מחרוזת של שמות מוצרים. למשל אם ברצוני למצוא את כל המוצרים שדוריטוס מופיע בשמם תקבלו כארגומנט filter_txt="דוריטוס".
עליכם ליצור חנות "חדשה", כלומר מבנה נתונים מסוג מילון של מילונים, בדומה למבנה הנתונים שקבלתם כארגומנט. אך הפעם החנות החדשה שתחזירו תהיה חנות מוקטנת - מבנה נתונים שיכיל את כל מוצרי החנות שבשם המוצר שלהם מופיע filter_txt ורק אותם. שימו לב כי יתכנו תווים ואותיות נוספות בשם המוצר. למשל אם חנות מכילה מוצר ששמו (כלומר ה- ItemName שלו) הוא "דוריטוס ירוק", ומוצר ששמו "50 גרם דוריטוס שחור" ומוצר נוסף שלישי ששמו "שחור דורי-טוס" עליכם להחזיר מילון חדש שיכיל רק את שני המוצרים הראשונים.

יישום נכון של הפונקציה תאפשר לכם להשתמש בכפתור ה- filter ב- GUI. הזינו טקסט וסננו את החנויות השונות. שימו לב ה- GUI מפעיל את פונקציית הפילטר על כל החנויות שנטענו לחנות ומציג אחת מהן. באמצעות התפריט לעיל וכפתור הshow תוכלו לבחור איזו חנות להציג.

6. יישמו את הפונקציה

```
def create_basket_from_txt(basket_txt):
```

כעת יהיה ברצוננו ליצור סלי קניות. סלים אלו יאפשרו לנו להשוות בין החנויות השונות. מבנה הנתונים **basket** אותו תיצרו יהיה מסוג רשימה (**list**) שאיבריה יהיו קודים (**ItemCode**) שונים (כמחרוזות) .



הפונקציה תקבל כארגומנט basket_txt משתנה מסוג String.

ה- String הוא מקטע טקסט המכיל ייצוג טקסטואלי של מספר מוצרים כפי שהוגדר במשימה 2. שימו לב כי מכיוון שהטקסט נבחר על ידי המשתמש יתכן כי הוא יסמן מוצר באופן חלקי. עליכם למצוא את כל המוצרים שקוד המוצר שלהם מתואר באופן מלא - כלומר את כל המוצרים שקוד המוצר שלהם מופיע בין סוגריים מרובעות.
את כל הקודים הללו עליכם להחזיר כרשימה של Strings.
למשל עבור הקלט:

```
basket_txt =' לחם) \n[324] {קמח}\n [123'
create_basket_from_txt(txt):
```

יהיה עליכם להחזיר:

```
["324"]
```

שכן המוצר היחיד שקוד המוצר שלו מתואר במלואו הוא קמח.

7. יישמו את הפונקציה

```
def get_basket_prices(store_db, basket):
```

פונקציה זו מקבלת כארוגמנט שני מבני נתונים:

1. store_db - חנות (מילון של מילונים) כפי שהוגדרה לעיל.
2. basket - סל קניות כפי שהוגדר במשימה 6 - כלומר רשימה של מחרוזות ItemCodes.

עליכם להחזיר את רשימת המחירים של המוצרים בסל עבור החנות הנ"ל. כלומר עבור כל מוצר בסל עליכם לבדוק מה מחיר המוצר בחנות ולצרף אותו לרשימת המחירים אותה תחזירו. שימו

לב כי אורכה של רשימת המחירים שתחזירו יהיה זהה לאורך הסל. כמו כן שימו לב כי בניגוד למילון, רשימה היא מבנה נתונים עם סדר. כלומר המחיר של המוצר השלישי ב- basket צריך להופיע במקום השלישי ברשימת המחירים שתחזירו. המחירים השונים יוזנו לרשימה כ- **floats**. יתכן כי חלק המוצרים שקבלתם בסל לא קיים בחנות. במידה וכך מחירו של המוצר יהיה None.

כעת ביכולתכם לעשות שימוש נוסף ב- GUI. לאחר שטענתם חנות (אחת או יותר) ביכולתכם לבחור באמצעות העכבר חלק מהטקסט המייצג את החנות. לחצו על Add selected. כעת צרפתם מוצרים לסל (באמצעות משימה 6) והדפסתם את מחירם בטבלה מימין (באמצעות משימה 7). סימון מוצרים נוספים ולחיצה נוספת על Add selected תגדיל את הסל הקיים שלכם. לחיצה על הכפתור Delete basket תמחק את הסל הנוכחי. נכון לכרגע המוצרים מופיעים בסל הקניות, באמצעות ה- ItemCode שלהם בלבד. במשימה 9 נשפר זאת.

8. יישמו את הפונקציה

```
def sum_basket(price_list):
```

פונקציה זו מחשבת מה סכום הסל הנתון בחנות מסויימת.

פונקציה זו מקבלת רשימה של floats & Nones כמו זו שיצרתם במשימה 7, כלומר רשימת מחירי המוצרים השונים בחנות כלשהי.
עליכם להחזיר tuple המורכב מצמד (sum_price_list, missing_items).

האיבר הראשון ברשימה הוא סכום כל המחירים ברשימת הקלט (סכום של None הוא 0), והאיבר השני הוא מספר המוצרים החסרים ברשימת הקלט (כלומר כמה מוצרים ברשימה הם None).

לאחר יישום הפונקציה תוכלו לראות את סכום ומספר המוצרים בסל של כל חנות בטבלה בימין ה- GUI. עד כה המוצרים הופיעו בסל הקניות באמצעות ה- ItemCode שלהם. אלמנט המקשה מאוד על המשתמש להבין מה מכיל סל הקניות שלו. על כן הפונקציה הבאה תמיר מ- ItemCode לשם מוצר.

9. יישמו את הפונקציה

```
def basket_item_name(stores_db_list, ItemCode):
```

פונקציה זו מקבלת איבר אחד בסל (ItemCode כלשהו) וצריכה לחזיר את שם המוצר (הItemName שלו).

הקושי במשימה זו הוא העובדה שכפי שראינו לא כל המוצרים קיימים בכל החנויות. לצורך זה הפונקציה מקבלת כארגומנט את רשימת החנויות של התכנית (stores_db_list). רשימת החנויות היא (כפי ששמה רומז) רשימה של מבני נתונים שכל אחד מהם מייצג חנות, כלומר

מילון של מילונים. סה"כ יש לנו כעת רשימה של מילונים של מילונים. עליכם למצוא את החנות בה המוצר קיים ולהחזיר את הייצוג הטקסטואלי של המוצר כפי שהוגדר ב- **string_item**. במידה והמוצר קיים ביותר מחנות אחת החזירו את שמו מהחנות הראשונה (לפי סדר האינדקסים ברשימת החנויות). במידה והמוצר לא קיים באף אחת מהחנויות החזירו רק את הקוד שלו בצורה הבאה:

```
return '['+ItemCode+']'
```

10. יישמו את הפונקציה

```
def save_basket(basket, filename):
```

פונקציה זו מאפשרת למשתמש לשמור את הסל שהרכיב.

הפונקציה מקבלת שני ארגומנטים: ,סל נוכחי, ושם קובץ.

עליכם ליצור קובץ טקסט מהסל הנוכחי, כך שכל איבר בסל יודפס בשורה נפרדת בקובץ. הדפסת האיברים תהיה בפורמט הבא:

```
[ItemCode1]
[ItemCode2]
…
[ItemCodeN]
```

11. יישמו את הפונקציה

```
def load_basket(filename):
```

פונקציה זו מקבלת כארגומנט שם של קובץ ומחזירה סל (כלומר רשימת קודים).

אתם רשאים להניח כי מבנה הקובץ דומה למבנה הקובץ אותו יצרתם במשימה 10.

12. יישמו את הפונקציה

```
def best_basket(list_of_price_list):
```

בפונקציה זו תעזרו למשתמש לבחור את החנות הטובה ביותר לרכוש בה את הסל שלו. לכאורה השאלה פשוטה - נרצה ללכת לחנות הזולה ביותר. אך לצערנו, כבר ראינו כי לא כל המוצרים נמצאים בכל רגע נתון בכל החנויות. כיצד נתמחר מוצרים חסרים?

בפונקציה זו תקבלו כארגומנט רשימה של רשימות. כל רשימה פנימית תייצג רשימת מחירים של אחת החנויות. עליכם להחזיר את החנות הזולה ביותר לפי החוקיות הבאה: מחיר מוצר יהיה

מחירו במידה והוא נמצא (כלומר אינו None), אך במידה ואינו נמצא נעניש את החנות במחיר גבוה במיוחד. מחיר המוצר החסר יהיה כמו מחירו המקסימלי בחנויות האחרות ועוד 25% עונש.
עליכם להחזיר את **האינדקס** של הרשימה הזולה ביותר לפי החוק שהמצאנו כאן.
למשל עבור האינפוט הבא:

```
L = [[5.0,2.0,8.0],[6.0,None,3.0],[None, 4.0,6.0]]
```

התת רשימה הראשונה תייצג את החנות הראשונה. סכום הסל בה יהיה סך מחירי מוצריה כלומר 8+2+5=15. התת רשימה השנייה תייצג את החנות השנייה. ברשימה זו חסר המוצר השני. על כן נסתכל על שתי החנויות האחרות ונווכח כי מחיר המוצר בחנויות האחרות הוא 2 ו4, ועל כן המחיר המקסימלי של המוצר השני הוא 4. נתן קנס של 25% לחנות הזו ועל כן מחיר בחנות יהיה 4*1.25=5. סך החנות הוא 3+5+6=14.
בחנות השלישית חסר המוצר הראשון. ועל כן נסכום אותו על פי 7.5=1.25*max(5, 6). סך החשבון בחנות השלישית הוא 6+4+7.5=18.5. על כן החנות הזולה ביותר היא החנות השנייה, ועליכם להחזיר את האינדקס שלה (שכמובן בפייתון הוא 1).

לאחר יישום משימה 10, החנות הזולה ביותר עבור הסל תצבע באדום ב- GUI.

עליכם ליישם את כל המשימות בקובץ ex5.py. הורידו את קובץ הטמפלייט מהאתר, מחקו את ה- pass מהגדרת כל אחת מהפונקציות ויישמו את פתרונכם. ה- type של ערך ההחזרה בכל אחת מהפונקציות מוגדר במשימה ובדוקמנטציה של התרגיל. כרגיל, אנו מציעים לבדוק את יישומה של כל אחת מהפונקציות בפני עצמה. אל תחכו לסוף התרגיל בכדי לבדוק את עצמכם.
בכדי לעזור לכם תוכלו למצוא באתר מספר קבצים של משרד הכלכלה המתאימים לפורמט שהוגדר לעיל. אנו כמובן ממליצים לגשת לאתר ולהוריד קבצים נוספים בכדי להעשיר את הידע שלכם לגבי כיצד מומלץ לעשות את הקניות לסופ״ש הקרוב.

הערות כלליות

תוכלו להניח שהקלטים לפונקציות **תקינים** (כלומר, שהם בהתאם לתיאור בהגדרת הפונקציה).

כמו כן, אין צורך להתמודד עם מקרים שלא הוגדרו כגון ערכי החזרה עבור חנות\מוצר\ערך לא קיימים.

נהלי הגשה

את תרגיל זה תבצעו בזוגות. אנו ממליצים להתחיל לעבוד עליו מוקדם שכן התרגיל ארוך.

על כל זוג להגיש קובץ אחד בלבד!

שימו לב, בתרגיל זה מלבד הקבצים הסטנדרטיים עליכם להגיש גם קובץ נוסף בשם AUTHORS (ללא כל סיומת). קובץ זה יכיל שורה אחת ובו הלוגינים של שני הסטודנטים, מופרד ע"י פסיק. כך:

minniemouse,mickeymouse

שימו לב: ודאו כי הגשתם קובץ AUTHORS תקין! אי הגשה של קובץ AUTHORS תקין תגרור הורדה בציון.

הגשת התרגיל:

הגישו קובץ zip הנקרא ex5.zip, המכיל את הקובץ ex5.py את קובץ ה- README ואת קובץ ה- AUTHORS. אל תגישו אף קובץ נוסף (בפרט לא את קובץ ה- GUI או אחד מקבצי ה- XML)

סקריפט קדם-הגשה **(Pre-submit script)**: כרגיל, תוכלו להריץ בדיקה בסיסית של התרגיל שכתבתם באמצעות הקלדה בתרמינל של הפקודה

```
~intro2cs/bin/presubmit/ex5 ex5.zip
```

כרגיל גם כן, מעבר של הסקריפט בצורה תקינה לא מבטיח ציון גבוה בתרגיל. עליכם לבדוק את עצמכם.

נזכיר כאן שוב כי בתרגיל זה ספקנו לכם פתרון בית ספר. ניתן למצוא אותו כאן:

```
~intro2cs/bin/ex5/ex5_gui
```

בהצלחה! ☺