|  |
| --- |
|  |
| **טיפול בסיפריות שהוצאו משימוש בפייתון** |
| עבודה מסכמת – תואר שני מדעי המחשב  שם: יוני שפונד  מנחה: פרופ׳ שמואל טישברוביץ |

List of Contents

1 הקדמה 4

2 רקע 4

2.1 פייתון – Python 4

2.1.1 תכנות דינאמי 4

2.2 ממשק תכנות יישומים - API 4

3 תכנות דינאמי מול סטטי 5

4 ביביליוגרפיה 6

List of Tables

**No table of figures entries found.**

List of Figures

**No table of figures entries found.**

# הקדמה

אחת השפות תכנות הכי פופולריות בשנים האחרונות היא Python [1], כאשר הסיפריות שלה משומשות עבור מכונות לומדות ומערכות לחישובים מדעיים. ממשקי תכנות יישומים (API) בסיפריות פייתון יוצאים מכלל שימוש בגלל שיפורי פיצ׳רים ותיקוני באגים בדיוק כמו בשפות אחרות.

שינוים אלה גורמות למפתחים לחוסר מוטיבציה להשתמש בפייתון עבור שימושים נוספים לפיתוחי תוכנה. בכל סיטואציה שבה מפתח צריך לזהות ולהחליף ממשקי תכנות יישומים (API), היא משימה מונוטונית וצורכת זמן יקר, כאשר בכל פרויקט קיים מספר רב של קריאות לממשקי תכנות יישומים (API). יתרה מזאת, התיעוד של ממשקים (במסמכים) גורם למשימות הללו להרבה יותר מסובכות ומאתגרות. הנושא הזה, לאורך שנים רבות היה בעיה רצינית עבור מפתחים.

בפרויקט זה, אני אציג כלי שבעזרתו יהיה ניתן להמנע מהבעיות הנ״ל וגם לחסוך המון שעות של עבודה של מפתחים

# רקע

## פייתון – Python

פייתון היא שפה דינאמית, קוד פתוח (open-source) חינמי, ומפוענח (interpreted) אשר בדרך כלל משומש לטובת בניית תוכנות, להפוך משימות לאוטומטיות, לבצע ניתוח נתונים, וכו׳. פייתון היא שפה לשימוש כללי, הכוונה היא לכך שניתן להשתמש בה לטובת מגוון רחב מאוד של תוכנות והיא לא ממוקדת לפתרון בעיות ספציפיות. הברסטיליות שלה והקלות שלה עבוד מפתחים צעירים, הפכו אותה לאחת השפות הכי פופולריות כיום [1].

פייתון תומכת בנוסף בתכנות מונחה עצמים וגם תכנות פרוצדורלי.

### תכנות דינאמי

תכנות דינאמי הוא ברובו אופטימיזציה מאשר רקורסיה.

בדומה לאלגוריתם ״הפרד ומשול״ (Divide and Conquer), תכנות דינאמי פותר בעיות על ידי שילוב של פתרונות עם תת פתרונות. אלגוריתמי הפרד ומשול מחלקים את הבעיה לתתי בעיות על מנת לפתור אותם בצורה רקורסיבית ורק לאחר מכן למזג בין הפתרונות שלהם ולפתור את הבעיה המקורית.

הרעיון הוא פשוט לשמור את התוצאות של תתי הבעיות בצורה כזו שאין צורך בחישוב שלהם מחדש לפעם הבעה שיהיה בהם צורך. אופטימיזציה כזו מפחיתה סיבוכיות זמן ריצה מזמן אקספוננציונלי לפולינומי.

## ממשק תכנות יישומים - API

ממשק תכנות יישומים הוא ערכה של ספריות קוד, פקודות, פונקציות, חוקים ופרוצדורות מוכנות, בהן יכולים המתכנתים לעשות שימוש פשוט, בלי להידרש לכתוב אותן בעצמם כדי שיוכלו להשתמש במידע של היישום שבו הם רוצים להשתמש לטובת היישום שלהם.

# תכנות דינאמי מול סטטי

* שפות תכנות סטטיות מבצעות בדיקות טיפוסי משתנים בזמן ריצת קומפילציה, כאשר שפות תכנות דינאמיות מבצעות בדיקות טיפוסי משתנים בזמן ריצה.
* שפות תכנות דינאמיות (כמו - Lisp, Perl, Python, Ruby) מתוכננות לייעל את עבודת המפתח ולבצע אופטימיזציות לקוד כך שאותה פונקציונליות תמומש בפחות קוד. שפות תכנות סטטיות (כמו - Java, C, C++) מתוכננות לייעל את עבודת החומרה על מנת שהקוד שרץ, ירוץ כמה שיותר מהר.
* שפות סטטיות דורשות הצהרה של טיפוסי המשתנים לפני שמשתמשים בהם, בזמן שבשפות דינאמיות אין את זה.

# ביביליוגרפיה

|  |  |
| --- | --- |
| [1] | "The top programming languages," GitHub, 2022. [Online]. Available: https://octoverse.github.com/2022/top-programming-languages. |