

תרגיל בית 5 בשפות תכנות (236319) - חלק יבש

שאלה 0

חלק א

.1

הנה הבניים התיאורתיים והמקבילים שמימשו להם.

- 1.Cartesian product -> Tuple
- 2.Disjoint union -> Union
- 3.Record -> Typedict
- 4.Mapping ->Dict
- 5.Unit, Null-> None
- 6.ANY-> any
- 7.BRANDING-> NEWTYPE
- 8.power set of T-> Set[Set[T]]
- 9.ENUM(is actually just a disjoint union of unit's, but anyway its also implemented directly)-> Enum
10. Exponentiation of T1^ T2 -> Callable[Callable[[T1], T2]

.2 דוגמאות:

```
28     from typing import Tuple
29
30     # Cartesian product of int, str, and float
31     def foo(x: Tuple[int, str, float]) -> None:
32         print(f"Integer: {x[0]}, String: {x[1]}, Float: {x[2]}")
33
34     foo((1, "hello", 3.14))
```

.1

```
40     from typing import Union
41
42     # Disjoint union of int and str
43     def bar(x: Union[int, str]) -> None:
44         if isinstance(x, int):
45             print(f"Integer: {x}")
46         else:
47             print(f"String: {x}")
48
49     bar(42)
50     bar("hello")
```

2.

```
1  from typing import TypedDict
2
3  class Person(TypedDict):
4      name: str
5      age: int
6      email: str
7
8
9  # Mapping of str to int
10 def count_fruits(fruit_counts: Dict[str, int]) -> None:
11     for fruit, count in fruit_counts.items():
12         print(f"{fruit}: {count}")
13
14 count_fruits({"apple": 10, "banana": 20, "cherry": 15})
15
```

5.

```
16  from typing import Any
17
18
19  # Function that can accept any type
20  def print_anything(x: Any) -> None:
21      print(f"Received: {x}")
22
23  print_anything(42)
24  print_anything("hello")
25  print_anything([1, 2, 3])
26
```

6.

```
27
28
29
30
31
32
33  from typing import NewType
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53  from typing import NewType
54
55
56  # Create a branded type for UserId
57  UserId = NewType('UserId', int)
58
59  def get_user_name(user_id: UserId) -> str:
60      print(user_id)
```

7.

```
62     from typing import Set  
63  
64     PowerSetInt = Set[Set[int]]  
8.
```

```
75  
76     from enum import Enum  
77     class MyEnum(Enum):  
78         OPTION1 = 1  
79         OPTION2 = 2  
80         OPTION3 = 3  
81
```

```
66  
67     from typing import Callable  
68  
69  
70     FuncType = Callable[[int], str]  
71  
72     def int_to_str(x: int) -> str:  
73         return str(x)  
74
```

פה בעשר אנחנו רואים אקספונצייאציה של int ל סטרינג, ודוגמה לפונקציה שישיכת

3. לא! חלק הוי מתבלבלים וחושבים שזה הנטיפר NONE אבל יש לו ערך אחד. הNONE
4. כי הנטיפר ANY מושם כTOP, כל ערך שקיים בשפה מתאים לANY

חלק ב

```
61 add_grades_to_set({1, 2, 3}, [4, 5, 'six'])
62 calculate_total_length(['hello', 'world'])
63 students[0].average_grade()
64 students[1].average_grade()
65 students[2].average_grade()
66 fibonacci(15)
67 factorial(0)
68 factorial(10)
69 add_grades_to_set(set(), [7, 8, 9])
70 calculate_total_length(['hello', 'world', 123])
71 calculate_total_length(['one', 'two', 'three'])
```

Failed (exit code: 1) (3128 ms)

```
main.py:61: error: List item 2 has incompatible type "str"; expected "int"  [list-item]
main.py:70: error: List item 2 has incompatible type "int"; expected "str"  [list-item]
Found 2 errors in 1 file (checked 1 source file)
```

1. השגיאה בשורה 61 קורرت בغالל שאנו מצפים לרשימה של ציונים- שכל אחד מהם הוא מספרשלם ומקבלים רשימה מעורבת שבה קיימות שתי מחרוזות.
2. השגיאה בשורה 70 קורرت בغالל שאנו מצפים לרשימה של מחרוזות אך מקבל רשימה מעורבת היota שבנוסף לשתי מחרוזות מקבלים גם מספרשלם.

פייתון היא שפה dynamically typed ולכון בדיקת הטיפוסים מתבצעת בזמן ריצה. לפני שהוספנו type annotations לפניה לא הוסיפה type constraints ולכון לא קיבלנו שגיאות.

זה מלמד אותנו למה הרעיון של gradual typing דרוש, כתיבת קוד שהוא dynamically typed בהתחלה מאפשר פיתוח מהיר ובבדיקה נוכנות כללית. כאשר אנו מוסיפים את ה-type annotations אפשרים זיהוי שגיאות מוקדמות יותר מאשר ב-type annotations. (כמובן שראינו בהרצאה כי החסרונות עולים על היתרונות בכל זאת).

מימין:

TYPE CHECKING IN PL'S



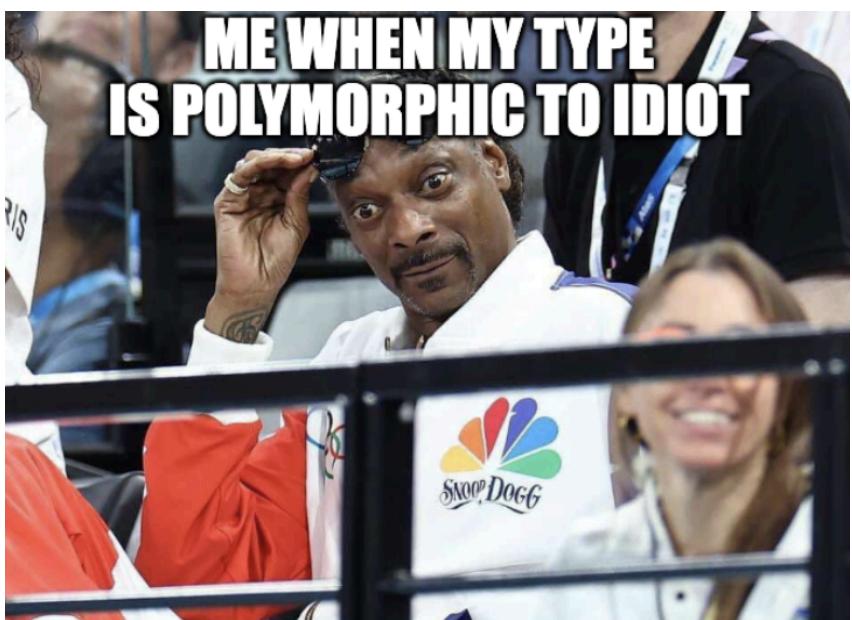
MY ONLY JOB IS MEMES



STATIC TYPING



ME WHEN MY TYPE IS POLYMORPHIC TO IDIOT



האלכזון של קנטור בשאלת 1 או משחו

