Yonatan Shirman I.D 208870246
Reut Dayan I.D 318879210

Documentation of exercise 3

We were asked to write a code that displays how cache memory works.

In general, the purpose of cached memory is to improve the running time of codes by "remembering" frequent actions and using their outputs. As a result it eliminates the need for doing the same actions over and over thus saving time.

The code is composed of 4 files. The inheritance order specified in each file is the link between the files.
Each file has its own class which does certain action and passes the output to other classes to eventually finish with the desired output (hopefully).
In the following lines well elaborate about the files composing the code:

1. **Lru_cache:**

   This is the "base" file, meaning it doesn't inherit. It contains the class "LruCache" which builds the structure of the cache. We chose the dictionary structure as our cache structure (because we needed a structure that supports a "key" and a "value" for that key. Dictionary was the perfect fit). It only stores the data.
   The class builds a list that we used as a "frequency checker" for the keys in the dictionary. The first value is the most commonly used.
   It also contains two more functions that manipulate the cache: getting a value or inserting a value based on the frequency of each value in the current cache.

2. **Cache_calculator**:

   This file inherits from "LruCache". It contains the class "CacheCalculator" which returns the result of a given arithmetical expression. Before it proceeds to calculate, its checking if the value already appears in the cache (using get func'). If the value is missing in the cache, it calculates the expression and inserting to the cache (using put func'). Every calculation contains the information of whether the calculation was performed or the value was retrieved from the cache.

3. **Calculation**:

   It contains two classes: "Operator", "Calculation". "Operator" class is an enum class that represents the operator of the arithmetic expression.

"Calculation" class does the "dirty job" – calculates the expression and returns the value to "calculate" func'.

4. **Main**:

   Demonstrates how the code work with some examples.