Assignment 1

1. Dynamic Table
   Overview: Implement dynamic table with **and** without struct hacking
   Operations:
   a. void* make_new_dynamic_table(int)
      This function should initialize the dynamic table
      The argument passed to this function is the initial number of elements that the dynamic table should be able to hold without being resized
   b. void push_back(void**, int)
      This function should insert an element at the end of a dynamic table
      Increase the size of the table when the it reaches a particular threshold capacity (Reallocate-Threshold) on push
      The table size has to be increased by a factor of 1.5
   c. void pop_back(void**)
      This function should pop an element from the end of dynamic table
      Decrease the size of the table when the it reaches a particular threshold capacity (Deallocate-Threshold) on pop
      The table size has to be decreased by a factor of 0.25

   Deliverables:
   a. Fill the statistics-template provided, with the details of the number of copy operations and the time required by the dynamic table implementation, at varying threshold levels, in the case of *with* and *without* struct-hacking in separate tables (as specified in the template).
   b. 2 separate implementation files.
      i. A *.c* file that implements the dynamic table with struct hacking.
      ii. A separate *.c* file that implements the dynamic table without struct hacking.

2. Splay Tree
   Overview: Implement a splay tree
   Operations:
   a. void* make_new_dictionary()
      This function should initialize the dictionary
   b. void insert(void*, int, int)
      Inserts a new key-value pair into the dictionary
      Update the value in case of duplicate keys
      The first integer argument in the function call is the key and the second integer argument is the value to be inserted in the dictionary.
   c. int find(void*, int)
      Searches for the *key* provided in the argument
      Returns *value* stored at the *key* if found else returns -1

   Deliverables:
   a. Fill the statistics-template provided, with the details of the number of rotations and the time required by the splay tree implementation.
   b. A *.c* file that implements the Splay tree