Mandatory tasks

1. Implement a stack data structure in which "we will be able to stack boxes". To do this, create a structure called Box, which has an integer type weight value (weight) and a pointer to the Box structure (next).

2. Create a global pointer variable from the Box structure (top). Write the initialize function, which sets the top variable to NULL. Call initialize in the main function!

3. Write the isEmpty function, which waits for a pointer to Box and returns whether the stack is empty or not. (The stack is empty if the top variable is a NULL pointer.)

4. Write the peek function that returns the weight of the current top Box. (Remember the stack is empty!)

5. Write the Push function, which gets an integer as a parameter, this will be the weight of the next box. If we don't exceed the capacity of our stack, create a new item on the stack and adjust its weight. The next data member of the new item should point to the previous data member! Point the top variable to the new item.

6. Write the Pop function, which releases the item at the top of the stack and sets the top variable to the one below it. (Don't forget the possibility of an empty stack either!)

7. Write the copyTop function, which copies the top of the stack and places the copy on top of the stack. Let's see what happens if we copy only the pointer to the top element, or if we allocate new memory space to the copy and make a copy per data member!

Optional tasks

1. Create a Person structure that contains (at least) a name and an age value. The name is stored in a char [30] and the lifetime in a memory area referenced by int *. For convenient use of the structure, use type aliasing.

2. Create a Person within the main function, whose data is set. Create another Person that gets the value of the first Person. (pl: p2 = p1;)

3. Write the values of the two Persons on the standard output. Change the age of the first Person. Let's write out the details of the two Persons again. Why does the age data for the second Person change? (shallow copy)

4. Improve your program by copying the deep values shown instead of assigning values between Persons (deep copy). Run the program again and demonstrate that changing the age data for the first Person will no longer change the age for the second Person.