

Assignment 3: Linked lists - 2, dynamic tables

1. Consider a dynamic table with the following properties.
 - a. Elements are stored in a dynamic array
 - b. Capacity is the size of the dynamic array
 - c. Size is defined as the number of elements stored in the array

Insert elements into dynamic table. Double capacity if size is equal to capacity before `push_back()`

Input: (n, elements)

9

6 7 8 12 4 10 11 1 15

Output:

capacity = 1; size = 1; elements = 6

capacity = 2; size = 2; elements = 6 7

capacity = 4; size = 3; elements = 6 7 8

capacity = 4; size = 4; elements = 6 7 8 12

capacity = 8; size = 5; elements = 6 7 8 12 4

capacity = 8; size = 6; elements = 6 7 8 12 4 10

capacity = 8; size = 7; elements = 6 7 8 12 4 10 11

capacity = 8; size = 8; elements = 6 7 8 12 4 10 11 1

capacity = 16; size = 9; elements = 6 7 8 12 4 10 11 1 15

Hint:

```
+---+
| 6 |
+---+
+---+---+
| 6 | 7 |
+---+---+
+---+---+---+---+
| 6 | 7 | 8 |   |   |
+---+---+---+---+
+---+---+---+---+
| 6 | 7 | 8 | 12 |
+---+---+---+---+
+---+---+---+---+---+---+---+
| 6 | 7 | 8 | 12 | 4 |   |   |   |
+---+---+---+---+---+---+---+
+---+---+---+---+---+---+---+
| 6 | 7 | 8 | 12 | 4 | 10 |   |   |
+---+---+---+---+---+---+---+
+---+---+---+---+---+---+---+
| 6 | 7 | 8 | 12 | 4 | 10 | 11 |   |
+---+---+---+---+---+---+---+
+---+---+---+---+---+---+---+
| 6 | 7 | 8 | 12 | 4 | 10 | 11 | 1 |
+---+---+---+---+---+---+---+
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
| 6 | 7 | 8 | 12 | 4 | 10 | 11 | 1 | 15 |   |   |   |   |   |   |
+---+---+---+---+---+---+---+---+---+---+---+---+---+---+---+
```

2. Implement `pop_back()` function that deletes the last element. If size is less than or equal to one fourth of the capacity then decrease the capacity by half.

Input: (capacity, size, elements, # of `pop_back()` calls).

16

9

6 7 8 12 4 10 11 1 15

5

Output:

```
capacity = 16; size = 8; elements = 6 7 8 12 4 10 11 1
capacity = 16; size = 7; elements = 6 7 8 12 4 10 11
capacity = 16; size = 6; elements = 6 7 8 12 4 10
capacity = 16; size = 5; elements = 6 7 8 12 4
capacity = 8; size = 4; element = 6 7 8 12
```

Hint:

Initial:

```
+-----+
| 6 | 7 | 8 | 12 | 4 | 10 | 11 | 1 | 15 | | | | | | | |
+-----+
4 pop_backs:
+-----+
| 6 | 7 | 8 | 12 | 4 | 10 | 11 | 1 | | | | | | | |
+-----+
+-----+
| 6 | 7 | 8 | 12 | 4 | 10 | 11 | | | | | | | |
+-----+
+-----+
| 6 | 7 | 8 | 12 | 4 | 10 | | | | | | | |
+-----+
+-----+
| 6 | 7 | 8 | 12 | 4 | | | | | | | |
+-----+
+-----+
| 6 | 7 | 8 | 12 | 4 | | | | | | |
+-----+
```

3. From a given dynamic table perform following operations:

- Delete an item by index
- Delete the first item by value
- Delete all items by value

Input: (capacity, size, elements, index, first_value, all_value)

```
16
10
4 2 3 4 3 5 3 4 4 3
2
3
```

Output:

```
capacity = 16; size = 9; elements = 4 2 4 3 5 3 4 4 3
capacity = 16; size = 8; elements = 4 2 4 5 3 4 4 3
capacity = 8; size = 4; elements = 2 5 3 3
```

4. Implement the following operations on a doubly linked list stored as a file.

- Insert
- Search
- Delete

5. Implement insertion sort in a linked list stored as a file.