CSCI-UA.0201 – 007 Computer Systems Organization
FALL 2022

PROJECT 2

Due date: 10.17.2022, 11.59pm
Late submissions due date: 10.20.2022, 11.59pm

This is an individual work. No team work is allowed.
Similarity check will be applied to submitted codes.

DYNAMIC ARRAY IN C

In your second project, you are going to implement a Dynamic Array which is also known as Vector. A Dynamic Array

- allocates memory dynamically (i.e. adds and removes objects dynamically)
- has random access to stored elements

C++ and Java has such a data structure. In Java, ArrayList and Vector classes also show similar behavior. In C++, it is called Vector. In this project, you will implement this data structure from scratch. **Your DynamicArray should be generic and allow for different data types**. Now you should:

- create 3 files:
    - 1 header file to separately handle the interface
    - Implementation file of your header file
    - main file to keep your driver function

- create a Dynamic Array struct which holds
    - pointer to an array of generic data type
    - initial capacity of the array
    - actual count of currently existing items
    - function pointers to all necessary functions: isEmpty, pushBack, removeAt, clear.
        - Hint: for instance, for pushBack function, your function pointer should look like: void (*pushBack)(struct DynamicArray*, void*)
        - These pointers will help you to integrate your functions directly into your struct just like classes (i.e. call your functions from your struct objects, not by using external functions) instead of handling your system as ADT+functions

- create these functions (besides, any helper functions are welcome)
    - isEmpty
        - checks whether the array is empty or not
    - pushBack
        - checks the limit of your array and inserts an item at the end of the array (resize the array to 3*capacity if space is not available)
    - removeAt
        - removes an item at the given index
            - Hint: you can swap the last item of the array with the given index to be removed and assign NULL to the last item
    - clear
        - clears your array
    - init
        - initializes your Dynamic Array by initializing the capacity, allocating the initial memory dynamically and assigning the function pointers to corresponding functions

Sample main file:

```c
#include <stdio.h>
#include "dynamicarray.h"

int main (int argc, char** args)

    DynamicArray da;
    DynamicArrayInit(&da);

    printf("Initialization - count: %d, capacity: %d\n", da.count, da.capacity);


//Here you notice that our struct behaves like a class with the help of function
pointers

    da.push_back(&da, "Gizem");
    da.push_back(&da, "NYU");
    da.push_back(&da, "CSCI");
    da.push_back(&da, "201");
    da.push_back(&da, "Fall");
    da.push_back(&da, "2022");
    da.push_back(&da, "Project201");

    printf("Step 2 - count: %d, capacity: %d\n", da.count, da.capacity);

    for (int i = 0; i < da.count; i ++)
    {
        printf("%s\n", (char*)da.data[i]);
    }

    da.remove_at(&da, 2);

    printf("Step 3 - count: %d, capacity: %d\n", da.count, da.capacity);

    printf("\nDynamic Array data:\n\n");

    for (int i = 0; i < da.count; i ++)
    {
        printf("%s\n", (char*)da.data[i]);
    }

    da.clear(&da);

    if(da.isEmpty)
        printf("The array is empty now\n");


    return 0;
}
```

Output:

```
Initialization – count: 0, capacity: 4

Step 2 – count: 7, capacity: 12
Gizem
NYU
CSCI
201
Fall
2022
Project201

Step 3 – count: 6, capacity: 12

Dynamic Array data:

Gizem
NYU
Project201
201
Fall
2022

The array is empty now
```