11.07.2020

BLG252E OBJECT ORIENTED PROGRAMMING FINAL EXAM

Spring 2020

Duration is 135 minutes between 12.00 – 14.15.

You have given a .zip file including necessary .h and .cpp files. Please fill out them considering given instructions and the main functions. Expected outputs are given in .txt format and your program output must exactly match with these. (Please be careful about the white spaces, punctuations and capitalizations.) Do not forget to print necessary messages.

- Plagiarism is strictly forbidden.
- The codes which could not be compiled will not be evaluated. Always backup your last working code version copy before moving on to implement the next functionality.
- You are not allowed to use STL. However, you may use <string> class.
- Please be careful about memory leaks.
- **Submission:** Please submit the given .zip file with updated versions of .cpp and .h files.

Question 1 (40 pts)

For this question you need to fill myDynamicArray.h and myDynamicArray.cpp and run q1.cpp successfully. You are asked to implement a generic dynamic array class. With this array class we want to be able to store different type of elements (int, char, other class objects etc.).

The solution must meet the following conditions:

- At creation time of an array, the **capacity** and **size** attributes must be initialized and required memory for the **dynamicArray** must be allocated.

- Implement reader methods for each attribute as **get_capacity(), get_size() and get_head()**. get_head() method must return a pointer to the beginning of a dynamicArray.

- Capacity is the maximum number of elements that an array may contain and must be initialized as 5.

- Size is the number of elements in an array and must be initialized as 0. Then it will be updated with addition and removal of elements.

- **Add** method must take an element as parameter and insert it to the end of the array.

- If the size of the array has reached the maximum capacity, then you need to double-size the capacity.

- **Remove** operation must remove the element from the end of the array and return its value. Trying to remove from an empty array would cause an error with a message “empty array\n” and should be handled **APPROPRIATELY** according to OOP styles.

- If the size of an array has decreased to the half of its capacity, then you need to update the capacity of array by dividing it by 2.

- Please do not forget to give back all the dynamically taken memory places and use private, public and **constant** properties where it is necessary.

11.07.2020

Question 2 (30pts)

For this question you need to fill animal.cpp, animal.h, bird.cpp, bird.h, cat.cpp, cat.h, dog.cpp, dog.h files. Also you need to update myDynamicArray.cpp and myDynamicArray.h from the question 1.

q2.cpp must run successfully.

In question 2, we want to create different types of animals as cat, dog and bird using the animal class as a base. Also we want to store their references in our dynamic array. Animals have common methods but implementation of them may differ for each animal type.

The solution must meet the following conditions:

- Please avoid code reduplication in all of your answers.
- Please keep in mind that, we should **never be** capable of creating a generic animal object. Instead, we should be able to create dogs, cats, birds etc.
- Each animal has a **name** attribute and an **outputvoice** method.
- Each animal type has a different output voice.
    - Dogs say “bark bark”
    - Cats say “meow meow”
    - Birds say “twit twit”

  The **outputvoice** method needs to print out the required voice for each class object.
- Please determine the methods which will not change the object's data (i.e. reader methods) and specify them appropriately.
- You need to add a **print** method into the myDynamicArray class. This method should automatically print out the elements of a dynamic array one by one using **cout <<** operation.

  (Hint: in case you need, C++ has a built-in pointer check function: std::is_pointer<T>::value to check if Type T is a pointer or not.)
- The **cout <<** operation should be capable of printing animal objects as well and call the **outputvoice** method.

Question 3 (30pts)

For this question, you need to update the animal.cpp, animal.h, myDynamicArray.cpp and myDynamicArray.h files from previous questions and fill out the leg.h and leg.cpp.

q3.cpp must run successfully.

In question 3, a leg class will be implemented. Each animal type has different number of legs and different leg length as in the table below.

Animal	Cat	Dog	Bird
# of legs	4	4	2
leg length	20 cm	30 cm	4 cm

The solution must meet the following conditions:

- Please avoid code reduplication in all of your answers.
- You need to implement a **leg class** which has an attribute as **length** which keeps the length of that leg object. Length value must be taken as a parameter at the creation of each leg object.
- Animal class has a **legLength** attribute which represents the length of the regarding animal's legs and a **legs** dynamic array (myDynamicArray object) which keeps the leg objects for that animal.
- Animal class objects could be initialized by one of the two ways:

    1. taking **name, legLength and legCount** as parameters or
    2. taking **legLength and legCount** as parameters.

- For each cat, dog and bird object, the objects must be initialized **without parameters** or only with the **name** parameter. Then related base initializer must be called appropriately by sending the leg number and leg length of the related animal.
- At initialization methods of animal class, attributes must be assigned accordingly, memory allocation for legs array must be done considering legCount, new leg objects must be created and their addresses must be added to the legs dynamic array.
- Copying an animal is also possible. But please do not forget that two animals cannot share the same legs.
- **get_legs()** method must return the legs array itself.