**Name : *Mehwish Habib Jaffar***

**Department : *C*omputer *S*cience**

**Submission DL: *20/11/202*4**

**Subject : Data Structure and Algorithm *(DSA)***

**Submitted to: *Tajmir khan***

*A simplified version of the dynamic array implementation in C++:*

*Key Requirements:*

*Dynamic Array: Use pointers for dynamic memory allocation.*

*Resize Array: Automatically resize when capacity is exceeded.*

*Basic Operations: Add element, display elements, find size, get value by index.*

*Simplified C++ Code:*

*cpp*

*Copy code*

*#include <iostream>*

*#include <stdexcept>*

*class DynamicArray {*

*private:*

*int\* arr;*

*int capacity;*

*int size;*

*void resize() {*

*capacity \*= 2;*

*int\* newArr = new int[capacity];*

*for (int i = 0; i < size; ++i)*

*newArr[i] = arr[i];*

*delete[] arr;*

*arr = newArr;*

*}*

*public:*

*DynamicArray(int initialCapacity = 2) {*

*capacity = initialCapacity;*

*size = 0;*

*arr = new int[capacity];*

*}*

*~DynamicArray() {*

*delete[] arr;*

*}*

*void addElement(int value) {*

*if (size == capacity) resize();*

*arr[size++] = value;*

*}*

*void display() const {*

*if (size == 0) {*

*std::cout << "Array is empty!" << std::endl;*

*return;*

*}*

*for (int i = 0; i < size; ++i)*

*std::cout << arr[i] << " ";*

*std::cout << std::endl;*

*}*

*int getSize() const { return size; }*

*int getValueByIndex(int index) const {*

*if (index < 0 || index >= size)*

*throw std::out\_of\_range("Index out of range!");*

*return arr[index];*

*}*

*};*

*int main() {*

*DynamicArray arr;*

*int choice, value, index;*

*while (true) {*

*std::cout << "\n1. Add Element\n2. Display Array\n3. Find Array Size\n4. Get Value by Index\n5. Exit\n";*

*std::cout << "Enter choice: ";*

*std::cin >> choice;*

*switch (choice) {*

*case 1:*

*std::cout << "Enter value: ";*

*std::cin >> value;*

*arr.addElement(value);*

*break;*

*case 2:*

*arr.display();*

*break;*

*case 3:*

*std::cout << "Size: " << arr.getSize() << std::endl;*

*break;*

*case 4:*

*std::cout << "Enter index: ";*

*std::cin >> index;*

*try {*

*std::cout << "Value at index " << index << ": " << arr.getValueByIndex(index) << std::endl;*

*} catch (const std::out\_of\_range& e) {*

*std::cout << "Error: " << e.what() << std::endl;*

*}*

*break;*

*case 5:*

*std::cout << "Exiting...\n";*

*return 0;*

*default:*

*std::cout << "Invalid choice!\n";*

*}*

*}*

*}*

*Key Features:*

*Dynamic Memory Allocation: Uses new and delete[] to manage memory for the array.*

*Automatic Resizing: The array resizes automatically when capacity is exceeded (resize() doubles the size).*

*Core Operations:*

*addElement(): Adds a new element to the array.*

*display(): Displays the contents of the array.*

*getSize(): Returns the current number of elements.*

*getValueByIndex(): Retrieves the value at a given index with bounds checking.*

*Simplified User Menu:*

*Add an element.*

*Display array contents.*

*Find the array size.*

*Get a value by index.*

*Exit the program.*

*Example Usage:*

*plaintext*

*Copy code*

*1. Add Element*

*2. Display Array*

*3. Find Array Size*

*4. Get Value by Index*

*5. Exit*

*Enter choice: 1*

*Enter value: 10*

*1. Add Element*

*2. Display Array*

*3. Find Array Size*

*4. Get Value by Index*

*5. Exit*

*Enter choice: 2*

*10*

*1. Add Element*

*2. Display Array*

*3. Find Array Size*

*4. Get Value by Index*

*5. Exit*

*Enter choice: 4*

*Enter index: 0*

*Value at index 0: 10*

*This version maintains the core functionality of a dynamic array with automatic resizing and basic operations, and it’s more concise and easy to understand.*