Data Structures and Algorithms
(ES221)

Dynamic Array

Dr. Zubair Ahmad

- **Attendance?**

  - Active Attendance
  - **Dead Bodies.**
  - **Active Minds**
  - Mobiles in hands -> Mark as absent
  - 80% mandatory

The static declaration of an array requires the size of the array to be known in advance

What if the actual size of the list exceeds its expected size?

Solution?

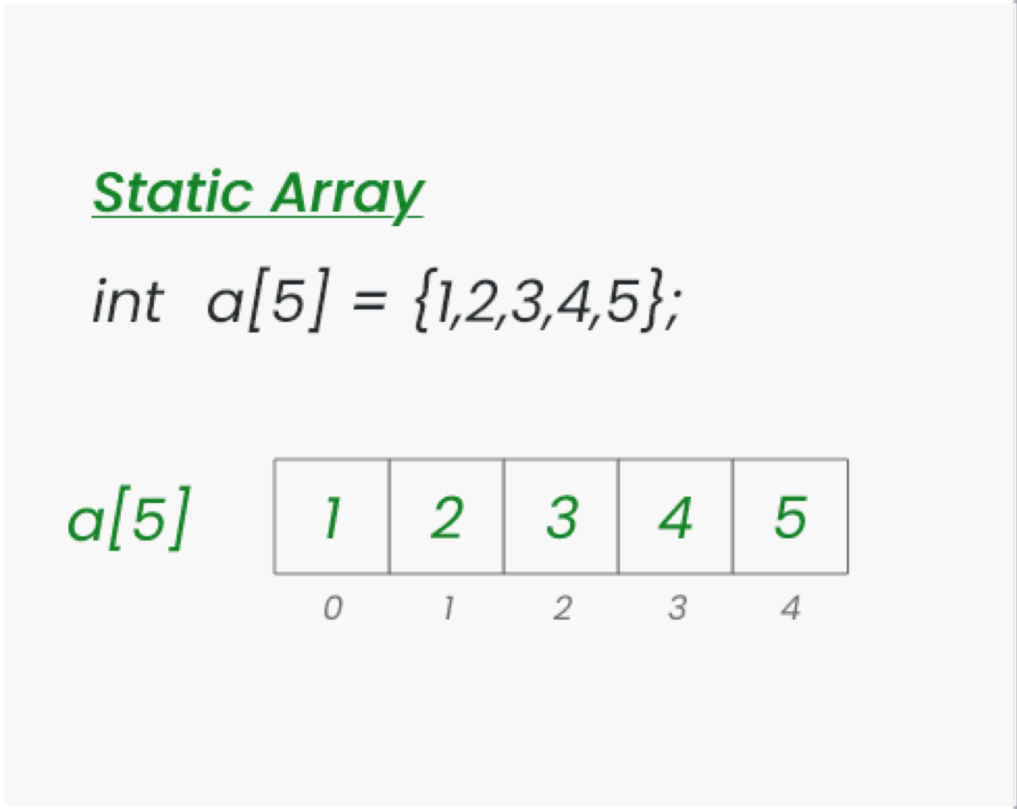
- Dynamic array
- Linked List

The size is determined automatically based on the number of values provided during initialization

The memory is allocated on the stack

The size of the array is fixed once it is defined.

```
int a[5] = {1, 2, 3, 4, 5};
```

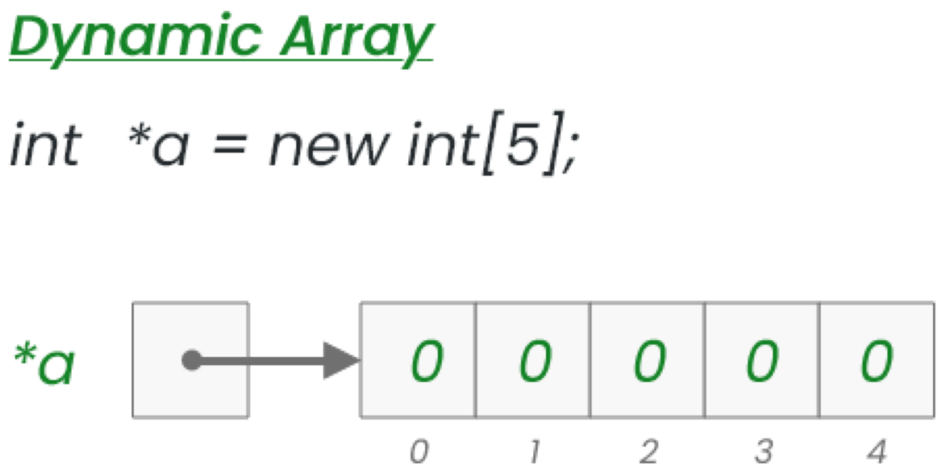


The memory is allocated during the run time using the ***new*** *keyword.*

The size is specified explicitly

The memory is allocated on the heap

```
int *a = new int[5];
```



Dynamic resizing

Managing large datasets

Dynamic Array

Dynamic data storage

Working with complex, potentially large, custom data structures

The new Keyword

In C++, we can use the new keyword to create a dynamic array. The number of elements to allocate is specified inside square brackets, and the type name comes before it. This ensures that the specified number of elements are allocated

Syntax

```
pointer_variable = new data_type;
```

Dynamic Array - Example

```
#include <iostream>
using namespace std;

int main() {
    int x, n;

    cout << "Enter the number of items:" << endl;
    cin >> n;

    int* arr = new int[n];

    cout << "Enter " << n << " items:" << endl;
    for (x = 0; x < n; x++) {
        cin >> arr[x];
    }

    cout << "You entered: ";
    for (x = 0; x < n; x++) {
        cout << arr[x] << " ";
    }

    // Free the allocated memory
    delete[] arr;

    return 0;
}
```

Dynamic Array growth strategy

Increase the size by a constant (tight strategy)

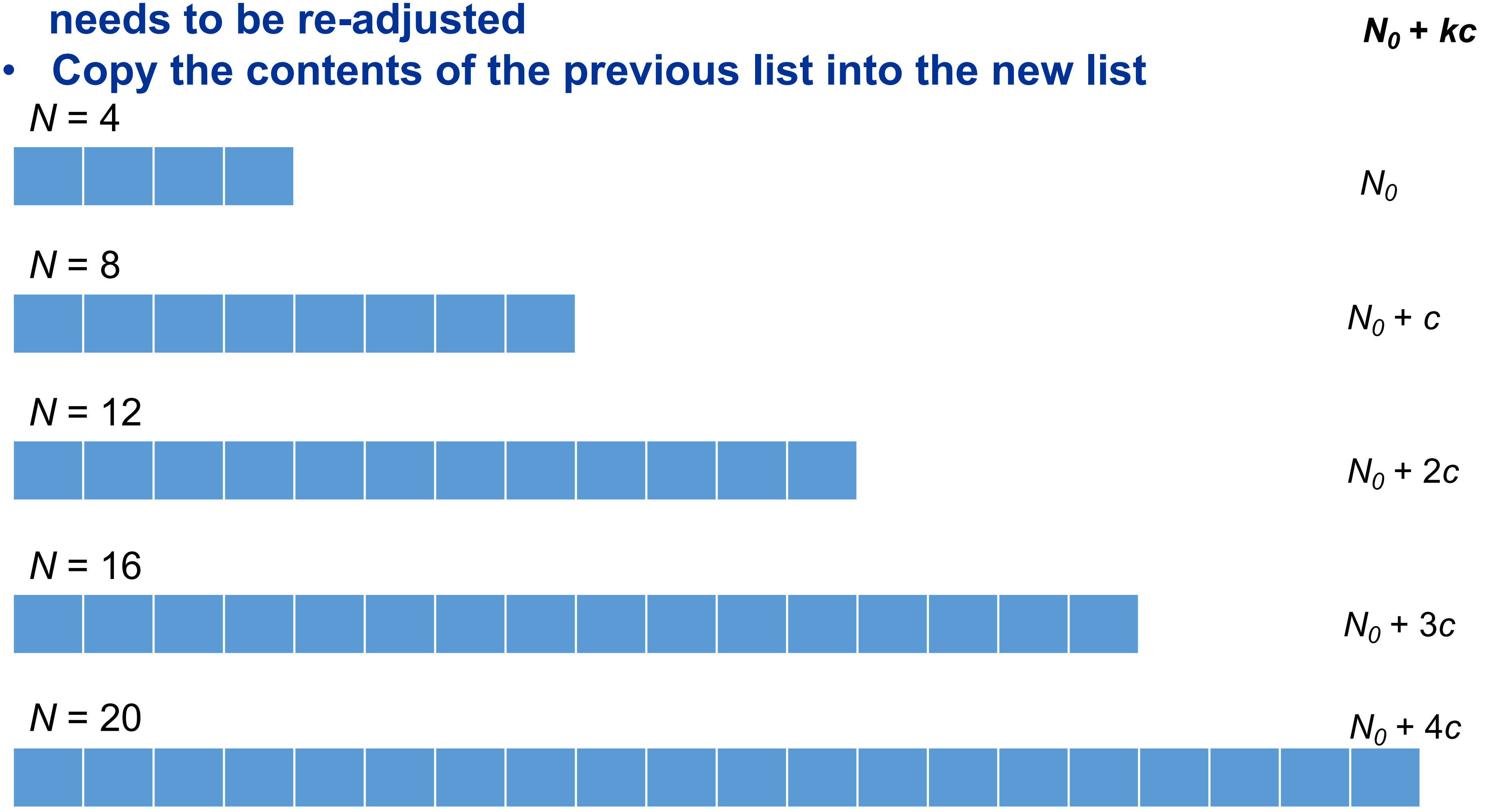
$N = N + c$

Double the size of the array

$N = 2 * N$

Increase the size by a constant

- **Increment the size by a constant number c, each time the size needs to be re-adjusted**
- **Copy the contents of the previous list into the new list**



Increase the size by a constant

$N = N_0 + kc$

$k = (N - N_0) / c$

Running Time: $N_0\,\mathrm{k} + c(1 + 2 + \ldots + \mathrm{k})$

$= N_0\,\mathrm{k} + ck\,(\mathrm{k}+1)/2$

$= \mathbf{O}(k^2)$

$= \mathbf{O}(N^2)$

Dynamic List: Add Element
Increase the size by a constant

Add at the end:

Step 1 : Increment the size of the array by a constant

Step 2 : Copy the old list into the new list

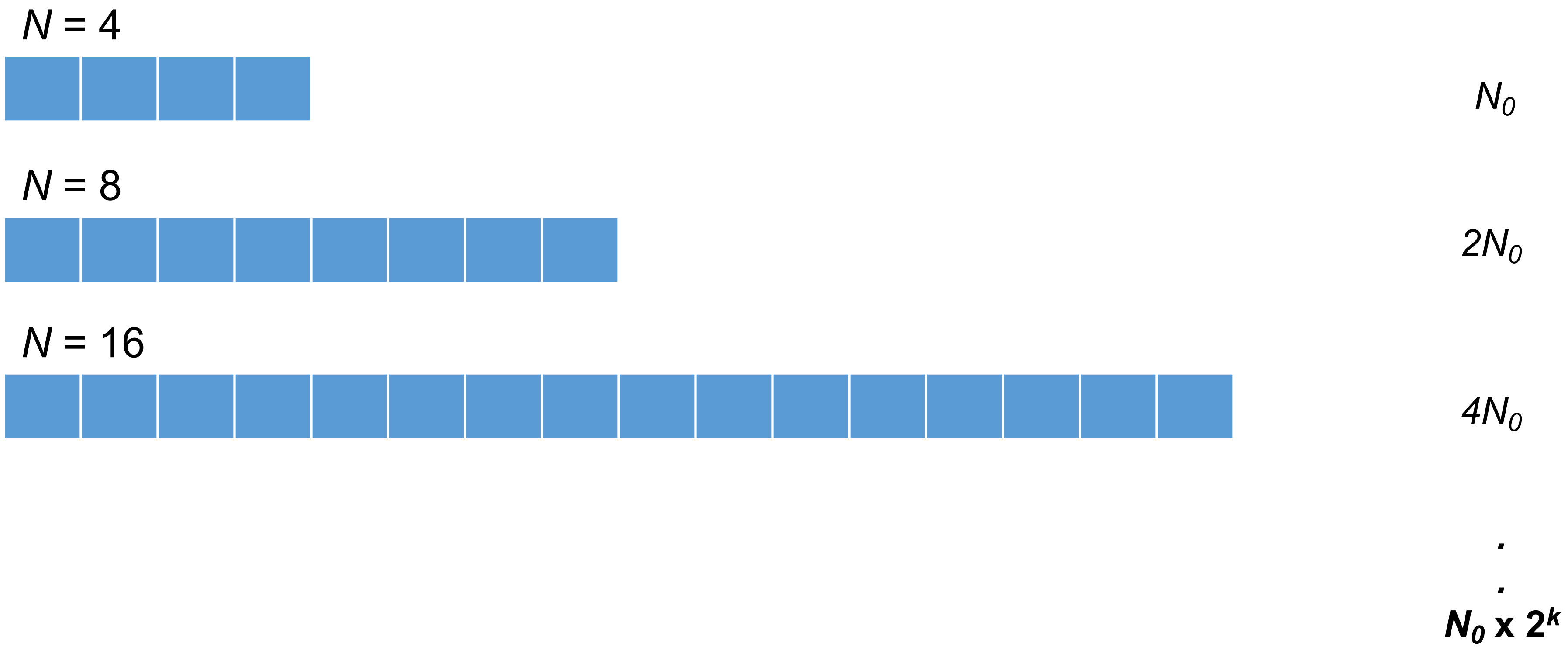
Step 3 : Rename the new list as the original list

Step 4 : Add the new element at end

```
if index == N  { // if the current index exceeds the size of the list
    N = N + c;       // increase the size of the list by c
    int *tempA =  new int [N];  // allocate memory for the new list
    for (int i = 0; i < N - c; i++)
        tempA[i] = A[i];  // copy the old list into the new list
    delete [] A;
    A = tempA;            // rename the new list as the original list
    }
    A[index]= NewElement;
    Index ++;
```

Double The Size of the Array

- **Increment the size twofold each time the size needs to be re-adjusted**
- **Copy the contents of the previous list into the new list**



Double The Size of the Array

$N = N_0 \text{ x } 2^k$

$k = \log_2(N/ N_0)$

Running Time: $= N_0(1 + 2 + 4 + 2^k)$

$= N_0 (2^{k+1} - 1)$

$= N_0(2^{\log_2(N/ N_0)+1} - 1) = 2N - N_0$

$= O(N)$

Dynamic List: Add Element
Double The Size of the Array

- Add at the end:
  - Step 1 : Double the size of the array
  - Step 2 : Copy the old list into the new list
  - Step 3 : Rename the new list as the original list
  - Step 4 : Add the new element at end

```
if index == N  {  // if the current index exceeds the size of the list
    N = N * 2;        // double the size of the list
    int *tempA =  new int [N];  // allocate memory for the new list
    for (int i = 0; i < N / 2; i++)
        tempA[i] = A[i];  // copy the old list into the new list
    A = tempA;             // rename the new list as the original list
    }
    A[index]= NewElement;
    Index ++;
```

Factors impacting performance of Dynamic Arrays

If the array starts small and grows slowly, it will need to resize often, which slows it down.

If the array starts big and grows quickly, it will waste a lot of memory, and resizing will take longer, reducing performance

Quiz!!

Quiz on next week Tuesday. Time and Venue will communicated later

Questions?

zahmaad.github.io