**Dynamic Array:**

A dynamic array is a variable-size list data structure that allows elements to be added or removed. Dynamic arrays overcome a limit of static arrays, which have a fixed capacity that needs to be specified at allocation.

A simple dynamic array can be constructed by allocating an array of fixed-size, typically larger than the number of elements immediately required. The elements of the dynamic array are stored at the start of the underlying array, and remaining positions towards the end of the array are reserved or unused. Elements can be added at the end of the dynamic array by using reserved space until the space is completely consumed. The underlying fixed-size array needs to be increased in size when further elements have to be added after all the space is consumed. Typically resizing is expensive as it involves allocating a new array and copying each element from the original array (costs *O(n)* time).

A fixed-size array will suffice in scenarios where the maximum logical size is fixed. A dynamic array will be needed when the maximum logical size is unknown initially, or likely to change.

Dynamic Array in C:

**int main(int argc, char \*argv[])**

**{**

**int i;**

**double\* p;**

**p = calloc(10, sizeof(double) );**

**for ( i = 0; i < 10; i++ )**

**\*(p + i) = i;**

**for ( i = 0; i < 10; i++ )**

**printf("\*(p + %d) = %lf\n", i, \*(p+i) );**

**free(p);**

**putchar('\n');**

**p = calloc(4, sizeof(double) );**

**for ( i = 0; i < 4; i++ )**

**\*(p + i) = i\*i**

**for ( i = 0; i < 4; i++ )**

**printf("\*(p + %d) = %lf\n", i, \*(p+i) );**

**free(p);**

**}**