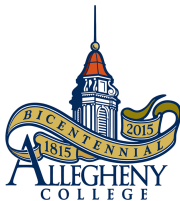# *CS200 - Computer Organization*
Dynamic Arrays & Structs

Aravind Mohan

Allegheny College

September 14, 2021

- How is the **Dynamic** array differ from a Static array?
- What is Structs? and how can we use them?

Variable Length Arrays

- Variable length arrays, use the space in the Stack and hence limited to store more data. It is often looked at an inefficient way of creating dynamic arrays.

Dynamic Arrays

- Malloc based dynamic arrays, uses the space in Heap and hence can store data of larger size. It is the most efficient method to create a Dynamic array.
- Malloc is simply memory allocation procedure, that allocates blocks of memory based on programmers request.

Dynamic Arrays

- A Pointer, can then be used to store, update, and read data elements from the memory. Malloc is part of the "stdlib" standard library.
- Look at "why.c" code in the src directory, to understand the difference.

Variable Length Array - Code

```c
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int size;
    printf("Enter the size of the array:");
    scanf("%d", &size);
    /* This code below is for variable length array!*/

    int array[size];
    printf("#created array\n");

    for (int i=0; i < size; i++)
        array[i] = 10*i;
    for (int i=0; i < size; i++)
        printf("%d\t", array[i]);
    printf("\n");
```

Variable Length Array - Code

```
Aravinds-MacBook-Pro:lecture12 amohan$ gcc why.c -o why.out
Aravinds-MacBook-Pro:lecture12 amohan$ ./why.out
Enter the size of the array:10000000
Segmentation fault: 11
Aravinds-MacBook-Pro:lecture12 amohan$
```

Aravind Mohan

Dynamic Array - Code

```c
/* creating the memory block to store the dynamic array! */
int *arr1 = (int *)malloc(size * sizeof(int));

/* assigning values to each element in the array */
for (int i = 0; i <  size; i++)
    *(arr1+i) =  10*i;

/* printing values inside each element in the array */
for (int i = 0; i <  size; i++)
    printf("%d\t", arr1[i]);

printf("\n");
free(arr1);
```

Dynamic Array - Code

2 Dimensional Dynamic Array

- How can we create a 2 Dimmensional Dynamic Array using a Malloc?
- **Approach 1:** A single pointer is used to allocate the memory block to hold the entire 2D array.
- **Approach 2:** An array of pointers is created for the individual rows. The element in the array [pointer], is used to allocate a memory block to hold all the columns in the individual rows.

Dynamic Array - Approach 1:

```c
void approach1(){
    int rs = 4, cs = 4;
    int *arr = (int *)malloc(rs * cs * sizeof(int));
    int i, j, count = 0;

    for (i = 0; i <  rs; i++) {
        for (j = 0; j < cs; j++) {
            *(arr + i*cs + j) = ++count;
            //printf("%d\t", arr2 + i*cs + j);
        }
        //printf("\n");
    }

    for (i = 0; i <  rs; i++) {
        for (j = 0; j < cs; j++) {
            printf("%d\t", *(arr + i*cs + j));
        }
        printf("\n");
    }
    free(arr);
```

Dynamic Array - Approach 2:

```c
int r = 3, c = 4, i, j, count;
int *arr[r];
for (i=0; i<r; i++)
  arr[i] = (int *)malloc(c * sizeof(int));
// Note that arr[i][j] is same as *(*(arr+i)+j)
count = 0;
for (i = 0; i <  r; i++)
   for (j = 0; j < c; j++)
     arr[i][j] = ++count; // Or *(*(arr+i)+j) = ++count

for (i = 0; i <  r; i++)
   for (j = 0; j < c; j++)
     printf("%d ", arr[i][j]);

printf("\n");
```

Structs in C

- A datatype declaration that defines a grouped list of variables to be placed under one name in a block of memory.
- Think of it as a class without functions, just variables.
- **Struct** can hold any type of members including arrays.
- **Possible to:** create a pointer to Struct.
- Let us do some examples.

KR - chapter 5 - [5.6 - 5.9]; chapter 06 - [6.1 - 6.4]

- Feel free to ask your questions.
- I welcome you to stop by after class time to clarify any confusion related to class topics.
- Also please stop by during my office hours so we can spend some time together.