# CS 211 - 3/9/18

Friday, March 9, 2018 9:59 AM

Exam 1 Next Friday: 3/16/18 at 10am during Lecture

Project 4 Due on Thursday 3/15/18

C Pointers is the main topic

0. MALLOC!!!!!!!!

    Dynamic array

        arrptr = ( type1*) malloc ( sizeof(type1) * size ) ;

    Linked List

        nodeptr = (type2*) malloc ( sizeof(type2) );

1. Dynamic Array
    - array pointer
    - amount of space currently allocated
    - amount of space in use
2. Linked List
    - data field(s)
    - next pointer
3. structures that define them
4. use of structure for variable declaration

Operations:

- initialize
- adding values (and growing)
- access

==========

Project 4
- linked list access is NOT a queue

==========
Dynamic Array

```
// structure
typedef struct dynarrStruct
{
   int*    arrptr;
   int     allocated;
   int     inUse;
} dynarr;

main()
{
  dynarr   arr1;

  init ( & arr1);

  addValue ( & arr1, val );
```

```
============================
void init (  dynarr* arr)
{  // initially alolocate space for 10 values
  arr->allocated = 10;
  arr->arrptr = ( int* )  malloc ( sizeof(int) * arr->allocated ) ;
  arr->inUse = 0;
}


void addValue ( dynarr* arr, int val)
{
  if ( arr->allocated <= arr->inUse )
    {
      int* tmp = ( int* )  malloc ( sizeof(int) * arr->allocated * 2 ) ;

      int a;
      for ( a = 0 ; a < arr->inUse ; a++)
          tmp[a] = arr->arrptr[a];

      free ( arr->arrptr );
      arr->arrptr = tmp;
       arr->allocated = arr->allocated * 2;
    }

 arr->arrptr[arr->inUse] = val;
arr->inUse = arr->inUse + 1;
}
```

==========

Linked List

```
typedef struct nodeStruct
{
   int elem;
   struct nodeStruct*  next;
} node;

main()
{
  node* head;

   init ( *head);

   addValue ( *head, val);
```

==========

```
void init ( node ** hd)
{
  *hd = NULL;
}
```

```
void addValue ( node** hd, int val)
{
  // see code from lab

}
```