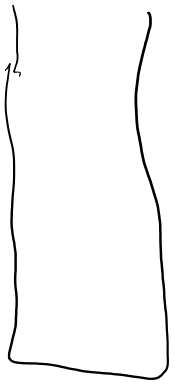
CS 211 - 2/2/18

Friday, February 2, 2018 9:38 AM

Last time:
dynamic array growing



This Time:
Doing the same (and more) with functions and structs

For project 2 assume a line of input is :

< { (( ) ( )) } > \n

Is the stack Empty? Yes ⇒ Balanced

< (a) \n

Is the stack empty? No ⇒ Not Balanced
missing >

> ( ) \n

If stack is Empty ⇒ Not Balanced
missing an opening <

(> \n

Top of stack does not correspond to current closing symbol
expecting)

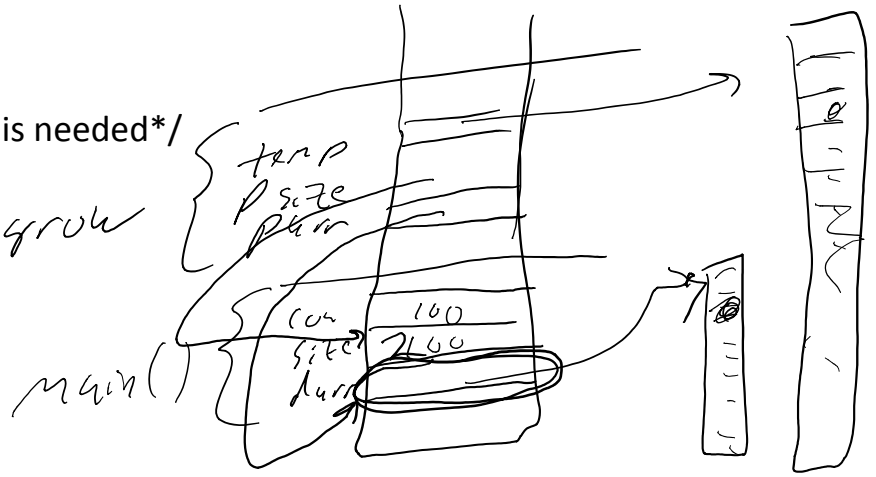
```
int *darr;
int size = 100;
darr = (int *) malloc (size* sizeof(int) );
int count = 0;
```

From <https://www.cs.uic.edu/pub/CS211/ProjectS18/proj1s18.pdf>

```
int val;
 /* loop until the user enters -999 */
 scanf ("%d", &val);
 while (val != -999)
```

```
/* loop until the user enters -999 */
 scanf ("%d", &val);
 while (val != -999)
  {
   /* CHECK IF ARRAY IS FULL  and Grow is needed*/
   if ( count >= size )
    grow (   &darr,   &size  )

   darr[count] = val;
   count++;
   scanf("%d", &val);
  }
```




```
void grow ( int **parr, int *psize)
 {
   int *temp = (int*) malloc ( sizeof(int) * *psize * 2) ;
   int* tempParr = *parr


      temp[i] = (*parr)[i];


   *parr = temp;
 *psize = *psize * 2;
```

code for the call: darr = grow (darr, ….);

```
int* grow ( int *parr, int *psize)
```