COP 3502 8/24/23

0) Acm

✓1) PO tips
- freq array
- fgets

✓2) Letter Grade Example

✓3) calloc

✓4) realloc (dynmemtest.c)

5) Example use of dyn mem
- array of strings (2 ways)

DONE - 1D array of a simple type

On Mac (to get to Eusti's)

> ssh aa123456@eustis.eecs.ucf.edu
> sftp aa123456@eustis.eecs.ucf.edu

] Use VPN if not on campus

free array

letter grades | line spaces, +'s, -'s
how many of each grade

```
fgets(strptr, max, stdin);
```

- strptr: ptr string mem already alloc.
- max: max # chars to read in
- stdin: usually file but for us it's standard input

+ + +
2 2 2
3 3

calloc

malloc - all purpose (can be used for arrays or single items)

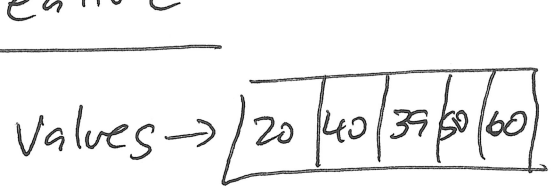
calloc - specifically for arrays

calloc(# items, size of item)

(1) all memory slot are set to 0.

```
int* scores = calloc(n, sizeof(int));
```

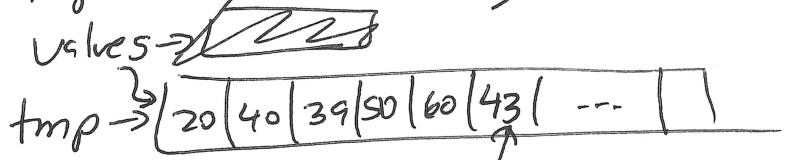
realloc



but now I have a 6th value to add...

What to do? (One solution)

```
int * tmp = malloc(sizeof(int) * 2 * n);
for (int i=0; i<n; i++) tmp[i] = values[i];
free(values);
values = tmp;
values[n] = item;
```



OR Sol 2

new size bytes

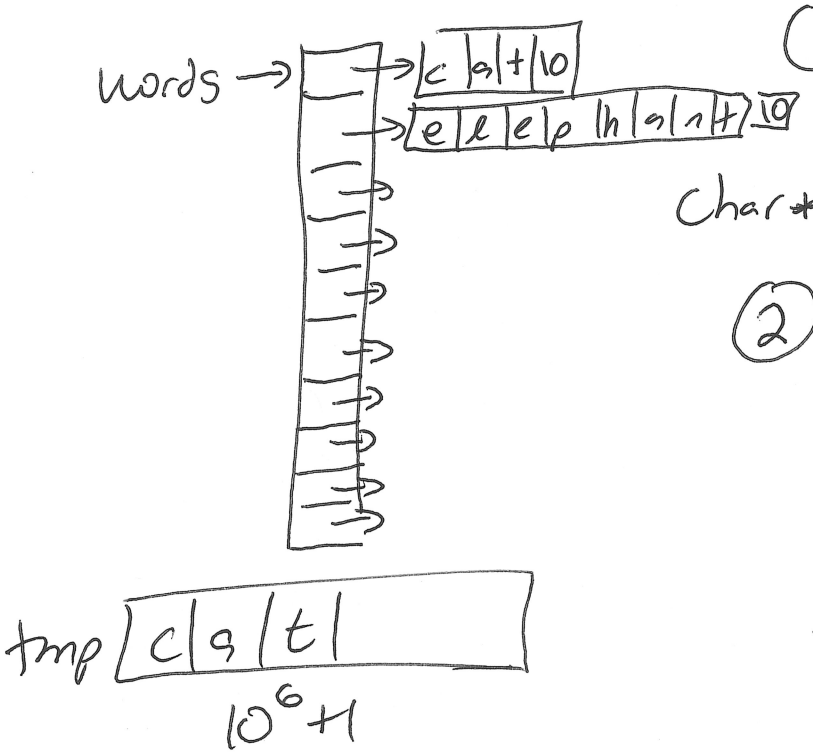
```
values = realloc(values, 2 * n * sizeof(int));
```

orig ptr

SIDE NOTE

realloc(ptr, 0) is identical to free(ptr);

Array of Strings



(1) 1st calloc is for array of pointers

```
char** words = calloc(n, sizeof(char*));
```

(2) allocate right memory for each string AFTER reading it into a tmp var that is "big enough"

```
for(int i=0; i<n; i++) {
    scanf("%s", tmp);
    int len = strlen(tmp);
    words[i] = calloc(len+1, sizeof(char));
    strcpy(words[i], tmp);
}
```

How to free

"each malloc/calloc equal + opposite free"

```
for(i=0; i<n; i++)
    free(words[i]);   ← free actual strings one by one
free(words) ⟶ frees array of pointers
```

Uses more memory but ok if most words are short.

SOLUTION #2

```
char** words = calloc(n, sizeof(char*));
for(int i=0; i<n; i++) {
    words[i] = calloc(MAX+1, sizeof(char));
    scanf("%s", words[i]);
}
```