

EECE.2160: ECE Application Programming

Spring 2016

Lecture 29: Key Questions

April 13, 2016

1. (Review) Explain the dynamic allocation functions `malloc()`, `calloc()`, and `realloc()`.

2. Explain how `free()` is used to deallocate memory.

3. **Example:** What does the following program print?

```
void main() {
    int *arr;
    int n, i;

    n = 7;
    arr = (int *)calloc(n, sizeof(int));
    for (i = 0; i < n; i++)
        printf("%d ", arr[i]);
    printf("\n");

    n = 3;
    arr = (int *)realloc(arr, n * sizeof(int));
    for (i = 0; i < n; i++) {
        arr[i] = i * i;
        printf("%d ", arr[i]);
    }

    n = 6;
    arr = (int *)realloc(arr, n * sizeof(int));
    for (i = 0; i < n; i++) {
        arr[i] = 10 - i;
        printf("%d ", arr[i]);
    }

    free(arr);
}
```


7. **Example:** Write each of the following functions:
 - a. **char *readLine()** : Read a line of data from the standard input, store that data in a dynamically allocated string, and return the string (as a **char ***)
Hint: Read the data one character at a time and repeatedly reallocate space in string

- b. `int **make2DArray(int total, int nR)`: Given the total number of values and number of rows to be stored in a two-dimensional array, determine the appropriate number of columns, allocate the array, and return its starting address

Note: if `nR` does not divide evenly into `total`, round up. In other words, an array with 30 values and 4 rows should have 8 columns, even though $30 / 4 = 7.5$