## **EECE.2160: ECE Application Programming**

Summer 2018

Lecture 14: Key Questions June 27, 2018

## **QUESTIONS:**

- 1. Review the operation of bitwise (AND, OR, XOR, NOT) and bit shift operators
- 2. Describe how in general, you perform the operations below on a bit or range of bits:
- a. Setting bit(s) (desired bit(s) = 1, all others unchanged)
- b. Clearing bit(s) (desired bit(s) = 0, all others unchanged)
- c. Flipping bit(s) (desired bit(s) change from  $0 \rightarrow 1$  or  $1 \rightarrow 0$ , all others unchanged)
- 3. Describe how to extract a group of bits from a larger value.
- 4. Describe how to read and print hexadecimal values.
- 5. Explain the malloc() function.
- 6. Explain the use of type casting, and why it is necessary with the allocation functions.
- 7. Explain the calloc() function.
- 8. Explain the realloc() function.
- 9. Explain how free () is used to deallocate memory.
- 10. What are the common pitfalls of dynamic memory allocation?

## **EXAMPLES:**

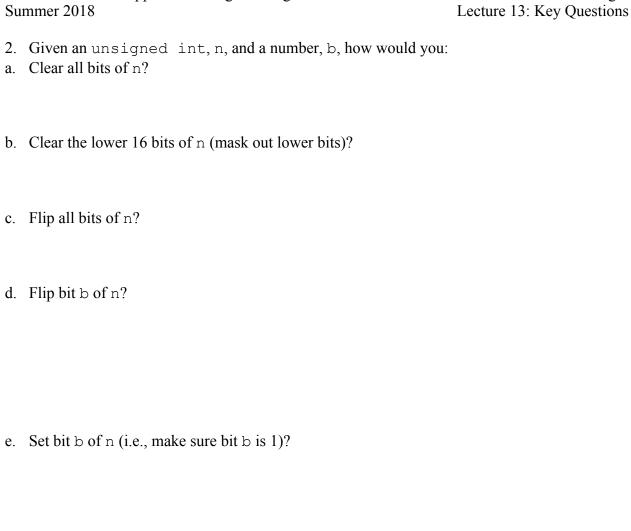
- 1. Evaluate each of the following expressions if you have the following unsigned int variables: A = 7, B = 10, and C = 0xFFFFFFFF
- a. A & B
- b. A | ~B

c.A ^ C

- d. A << 4
- e.B >> 5

f. A | (B << 2)

f. Clear bit b of n (i.e., make sure bit b is 0)?



## 3. 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);
}
```