Pointer Basics

To tell our compiler to interpret the variable x as an address of an int we say

```
int *x;
```

To tell our compiler to assign to x the address of y we say

```
x = &y;
```

To tell our compiler to assign to y the value at x we say

```
y = *x;
```

Programming with Pointers

Write the following functions so that they perform according to the provided comment. Not all questions are guaranteed to be soluble.

1.
```
/*The first function you write in any language.
 *Prints the string "Hello World" to standard output.*/


void hello_world() {
        printf("Hello World\n");
}
```

2.
```
/*Swaps the value of two ints outside of this function.*/


void swap(int *x, int *y) {
        int temp = *x;
        *x = *y;
        *y = temp;
}
```

3.
```
/*Increments the value of an int outside of this function by one.*/


void plus_plus(int *x) {
        x[0]++;
}
```

4.
```
/*Returns a buffer for N ints.*/
```

```
//Insoluble using provided machinery. Can of course be done using malloc.
int* allocate_buffer(unsigned int size) {
        return malloc(sizeof(int) * size); //note that this is an unchecked malloc
}
```

5.
```
/*Returns the number of bytes in a string. Does not use strlen.*/
```

```
int mystrlen(char* str) {
    int count = 0;
    while(*str++) {
        count++;
    }
    return count;
}
```

6.
```
/*Returns the number of elements in an array ARR of ints.*/
```

insoluble

Problem?

The following code segments may contain either logic or syntax errors. Find them.

1.
```
/*Returns the sum of all the elements in SUMMANDS.*/
int sum(int* summands) { //int sum(int* summands, unsigned int n)
        int sum = 0;
        for (int i = 0; i < sizeof(summands); i++)         //i < n
                sum += *(summands + i);
        return sum;
}
```

2.
```
/*Increments all the letters in the string STRING, held in an array of length N.
 *Does not modify any other memory which has been previously allocated.*/
void increment(char* string, int n) {
        for (int i = 0; i < n; i++)        //for (i = 0; string[i] != 0; i++)
                *(string + i)++;           //string[i]++; or (*(string + i))++;

                //consider the corner case of incrementing 0xff
}
```

3.
```
/*Copies the string SRC to DST.*/
void copy(char* src, char* dst) {
        while (*dst++ = *src++);
}
```