

**CMSC 21 2<sup>nd</sup> Long Exam**

**I. True or False**

1. True
2. False
3. False
4. True
5. False
6. False
7. False
8. False
9. True
10. True
11. True

**II. Provide the answers to the following:**

1. With 2D arrays, the concept of rows and columns are only theoretical. The number of columns tells the computer how many elements are needed before proceeding to the next row. The number of elements in each row should be the same for every row. The number of rows can be left unspecified since new rows are just placed next to each other.

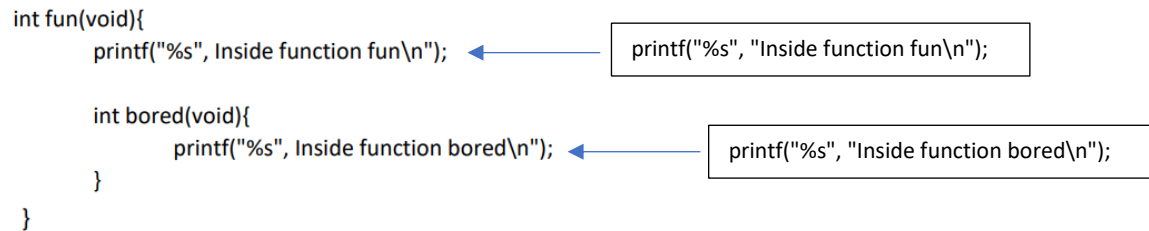
2.

- a. `bool isPalindrome(char *string);`
- b. `float computeAverage(float arr[20]);`
- c. `void reverseSentence(void);`
- d. `float squareRoot(int num);`

3.

```
int fun(void){
    printf("%s", Inside function fun\n");
}

int bored(void){
    printf("%s", Inside function bored\n");
}
```



The printf statement is lacking a “ which causes an error. This can be fixed by adding the missing “. It is also important to note that the code would work but the function bored can only be called inside the function fun. The code would encounter an error if bored was called in other functions like the main function. This could be fixed by moving the function bored outside function fun.

Fixed code:

```
int fun(void){
    printf("%s", "Inside function fun\n");
}

int bored (void){
    printf("%s", "Inside function bored\n");
}
```

b. 

```
int product (int a, int b){
    int result = a * b;
}
```

The code snippet is lacking a return statement.

Fixed code:

```
int product (int a, int b){
    int result = a * b;
    return result;
}
```

c. 

```
void fun (float a);
{
    float a;
    printf("%f", a);
}
```

To fix the code, the semicolon after (float a) should be removed. Also, the float a inside the function body should be removed since float a was already used as a formal parameter for the function.

Fixed code:

```
void fun (float a)
{
    printf("%f", a);
}
```

```
d. void sum(void){
    printf("%s", "Enter three integers: ")
    int a, b, c;
    scanf("%d%d%d", &a, &b, &c);
    int total = a + b + c;
    printf("Result is %d", total);
    return total;
}
```

The end of the printf statement is lacking a semicolon. Also, the data type of the function can't be void since it returns the variable total. Void will be changed to int.

Fixed code:

```
int sum(void){
    printf("%s", "Enter three integers: ");
    int a, b, c;
    scanf("%d%d%d", &a, &b, &c);
    int total = a + b + c;
    printf("Result is %d", total);
    return total;
}
```

4.

```
a. int arr[5] = {1,2,3,4,5};
```

```
b. int *ptr;
```

```
c. ptr = arr;
```

```
d. for (int i=0; i <=4 ; i++){
    printf("%d", *(ptr + i));
}
```

```
e. for (int i=0; i <=4 ; i++){
    printf("%d", *(arr + i));
}
```

```
f. arr[2]; //f.1
    *(arr + 2); //f.2
    ptr[2]; //f.3
    *(ptr + 2); //f.4
```

g. ptr + 2

address: 2508

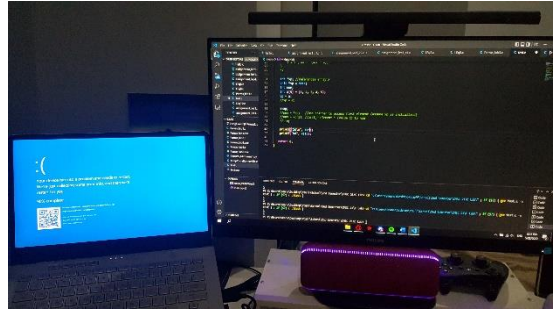
value: 3

5.

a. `++xp;`

Error in code: The pointer `xp` has not been initialized. Using pointers that have not yet been initialized can cause undefined behaviors.

Side note: I tried running the code without initializing the pointer and caused my screen to freeze, then my laptop crashed.



b. `num = xp; //use pointer to access first element (assume xp is initialized)`

Error in code: a `*` is needed to be placed before `xp` in order to use it as a pointer in accessing the first element.

Fix: `num = *xp;`

c. `num = *xp[1]; //assign element 1 (value 2) to num`

Error in code: When using the pointer index notation, the `*` before `xp[1]` should be removed.

Fix: `num = xp[1];`

d. `++x;`

Error in code: increments can only be applied on arrays in array index notation.

Fix: `++x[0];`

### **III. Application**

GitHub link: <https://github.com/Megunut/CMSC-21/tree/main/2nd%20Long%20Exam>