

I.

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

II.

1. It is okay to leave the first dimension of an array parameter unspecified but not the other dimensions, because the first dimension does not represent individual elements inside the array. The second dimension, third dimension, and so on is to be specified since these dimensions are representative of the individual elements inside the array and is used by the compiler to be able to locate these individual elements of the array.

2. A. `bool isPalindrome(char *string);`
B. `float computerAverage(float arr[20]);`
C. `void reverseSentence(void);`
D. `float squareRoot(int num);`

3.

- A. **You cannot have nested functions in C.**

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

- B. **I don't see any errors in the snippet aside from missing the return statement (which is negligible).**

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

- C. **You cannot redeclare an argument. There is also no need to put a semicolon after declaring the function.**

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

- D. **There is no need to put %s when printing a string. You can just directly print the string in the statement. Also, you cannot return a value when your return type of the function is void. Lastly, there are missing semicolons.**

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

4. A. `int numbers[SIZE] = { 1, 2, 3, 4, 5 };`
 B. `int *ptr;`
 C. `ptr = numbers;`
 D. `for(int i=0; i<5; i++){`
 `printf("%d", *(ptr+i));`
 `}`
 E. `for(int i=0; i<5; i++){`
 `printf("%d", *(numbers+i));`
 `}`
 F. 1. `numbers[1]`
 2. `*(numbers+1)`
 3. `ptr[1]`
 4. `*(ptr+1)`
 G. The address of `ptr+2` is the address of the third element in the `numbers` array. The value is 3.
5. A. **No error in this line.**
 B. **`num = *xp;`** instead of `num = xp;`
 C. **No need to put an asterisk sign to `xp` since you are using an array notation.**
 D. **You cannot use an increment operand before an array name.**

III.

```
1  #include <stdio.h>
2  #include <ctype.h>
3  #include <stdbool.h>
4
5  //Functions cannot return an array so I used a pointer here
6  int *scan_word(int occurrences[26]){
7      char c;
8      //Gets an input from the user until a space is detected
9      while((c = getchar()) != '\n'){
10         //checks if the input are letters
11         if(isalpha(c)){
12             //Increments the value of the index located by subtracting the position of the letter minus the position of 'A'
13             occurrences[toupper(c) - 'A']++;
14         }
15     }
16     //returns the address of the array occurrences
17     return occurrences;
18 }
19
20 //Function for checking if anagram
21 bool isAnagram(int occurrences1[26],int occurrences2[26]){
22     //same variable is used as an indicator
23     int same = 1;
24     //goes through every element of occurrences1 array and compares it with every element of occurrences2 array
25     for (int i = 0; i < 26; i++){
26         //if an element is not the same, the value of same is changed to 0
27         if (occurrences1[i] != occurrences2[i]){
28             same = 0;
29             break;
30         }
31     }
32     //returns a bool type value (0 or 1)
33     return same;
34 }
35
36 int main(void){
37     //declaration of variables and arrays
38     int *fWord, *sWord, letters[26] = {0};
39     int occurrences1[26], occurrences2[26];
40
41     //Asks the user the first word but the process of storing input is presented in the function scan_word
42     printf("Enter the first word: ");
43     //calls the function scan_word while passing the letters array as the argument and stores the returned value onto fWord
44     fWord = scan_word(letters);
45     printf("\n");
46     //every element of occurrences1 array is changed accordingly to the elements of the fWord array which was returned from the function
47     for (int i = 0; i < 26; i++){
48         occurrences1[i] = fWord[i];
49     }
50
51     //resets the values of the elements inside the letters array
52     for (int i = 0; i < 26; i++){
53         letters[i] = 0;
54     }
55
56     //Asks the user the second word but the process of storing input is presented in the function scan_word
57     printf("Enter the second word: ");
58     //calls the function scan_word while passing the letters array as the argument and stores the returned value onto sWord
59     sWord = scan_word(letters);
60     printf("\n");
61     //every element of occurrences2 array is changed accordingly to the elements of the fWord array which was returned from the function
62     for (int i = 0; i < 26; i++){
63         occurrences2[i] = sWord[i];
64     }
65
66     //calls the function isAnagram
67     if(isAnagram(occurrences1, occurrences2)){ //if returned value is 1, which is true, it is an anagram
68         printf("The words are anagrams\n");
69         return 0;
70     }
71     printf("The words are not anagrams\n"); //else, returned value is 0, which is false
72     return 0;
73 }
74
```

1.

2.

```

1  #include <stdio.h>
2  #include <ctype.h>
3  #include <stdbool.h>
4
5  //Functions cannot return an array so I used a pointer here
6  int *scan_word(int occurrences[26]){
7      char c;
8      //Gets an input from the user until a space is detected
9      while((c = getchar()) != '\n'){
10         //checks if the input are letters
11         if(isalpha(c)){
12             //Increments the value of the index located by subtracting the position of the letter minus the position of 'A'
13
14             occurrences[toupper(c) - 'A']++;
15         }
16     }
17     //returns the address of the array occurrences
18     return occurrences;
19 }
20
21 //Function for checking if anagram
22 bool isAnagram(int occurrences1[26],int occurrences2[26]){
23     //same variable is used as an indicator
24     int same = 1;
25     //goes through every element of occurrences1 array and compares it with every element of occurrences2 array
26     for (int i = 0; i < 26; i++){
27         //if an element is not the same, the value of same is changed to 0
28         if (*occurrences1 + i) != (*occurrences2 + i){
29             same = 0;
30             break;
31         }
32     }
33     //returns a bool type value (0 or 1)
34     return same;
35 }
36
37 int main(void){
38     //declaration of variables and arrays
39     int *fWord, *sWord, letters[26] = {0};
40     int occurrences1[26], occurrences2[26];
41
42     //Asks the user the first word but the process of storing input is presented in the function scan_word
43     printf("Enter the first word: ");
44     //calls the function scan_word while passing the letters array as the argument and stores the returned value onto fWord
45     fWord = scan_word(letters);
46     printf("\n");
47     //every element of occurrences1 array is changed accordingly to the elements of the fWord array which was returned from the function
48     for (int i = 0; i < 26; i++){
49         *(occurrences1 + i) = *(fWord + i);
50     }
51
52     //resets the values of the elements inside the letters array
53     for (int i = 0; i < 26; i++){
54         *(letters + i) = 0;
55     }
56
57     //Asks the user the second word but the process of storing input is presented in the function scan_word
58     printf("Enter the second word: ");
59     //calls the function scan_word while passing the letters array as the argument and stores the returned value onto sWord
60     sWord = scan_word(letters);
61     printf("\n");
62     //every element of occurrences2 array is changed accordingly to the elements of the fWord array which was returned from the function
63     for (int i = 0; i < 26; i++){
64         *(occurrences2 + i) = *(sWord + i);
65     }
66
67     //calls the function isAnagram
68     if(isAnagram(occurrences1, occurrences2)){ //if returned value is 1, which is true, it is an anagram
69         printf("The words are anagrams\n");
70         return 0;
71     }
72     printf("The words are not anagrams\n"); //else, returned value is 0, which is false
73     return 0;
74 }
75

```

Github Link: <https://github.com/ainzzcutie/CMSC21.git>