241-15-331_assignment-1

```
 *** ARRAY ***

1.#include <stdio.h>

int main(){

    int numberOfTestCase;
    scanf("%d", &numberOfTestCase);
    int caseArray[numberOfTestCase];
    for(int i = 0; i < numberOfTestCase; i++){
        scanf("%d", &caseArray[i]);
    }
    for(int i = 0; i < numberOfTestCase; i++){
        printf("%d, ",caseArray[i]);
    }
    int largestNumber = caseArray[0];
    for(int i = 1; i<numberOfTestCase; i++){
        if(!(largestNumber > caseArray[i])){
            largestNumber = caseArray[i];
        }
    }
    //printf("The largest number in the array is %d", largestNumber);
    int secondLargestNumber = 0;
    for (int i =0; i< numberOfTestCase; i++){
        if(!(secondLargestNumber > caseArray[i]) && (caseArray[i] != largestNu
            secondLargestNumber = caseArray[i];
        }
    }
    //printf("The second largest number in the array is %d", secondLargestNumb
    printf("%d\n",secondLargestNumber);
}


2.#include<stdio.h>

int main(){
    int testCase;
    scanf("%d", &testCase);
    int testCases[testCase];
    for (int i = 0; i < testCase; i++){
        scanf("%d", &testCases[i]);
    }
    for(int i = (testCase-1); i >= 0; i--){
        printf("%d ", testCases[i]);
    }

    return 0;
}

3.#include <stdio.h>

int main(){
    int numberOfTestCases;
    scanf("%d", &numberOfTestCases);
    int testCases[numberOfTestCases];
    for (int i = 0; i < numberOfTestCases; i++){
         scanf("%d", &testCases[i]);
    }
    int rotation;
    printf("Rotate: ");
    scanf("%d", &rotation);
    int rotationPosition = (numberOfTestCases/rotation)-1;
    for (int i = rotationPosition+1; i < numberOfTestCases; i++){
        printf("%d ", testCases[i]);
    }
    for (int i = 0; i < rotationPosition+1; i++){
        printf("%d ", testCases[i]);
    }
    return 0;
}

4.#include <stdio.h>
#define MAX_NUMBER_IN_THE_ARRAY 10
int main(){

    int numberOfTestCases;
    scanf("%d", &numberOfTestCases);
    int testCases[numberOfTestCases+1];
    int containerPod[MAX_NUMBER_IN_THE_ARRAY] = {0};
    /*
        if(containerPod[0] == 0){
            printf("NULL");
        }*/
    int temp;
    for(int i = 0; i < numberOfTestCases; i++){
        scanf("%d", &temp);
        containerPod[temp]++;
    }

    for (int i = 0; i < MAX_NUMBER_IN_THE_ARRAY; i++){
        if (containerPod[i] > 0){
            printf("%d: %d times\n", i, containerPod[i]);
        }
    }
}

5.
#include <stdio.h>

int main(){

    int previousNumber = 0;
    int N;
    scanf("%d", &N);
    int temp;
    int missingNumber = 0;
    int counter = 0;
    for (int i = 0; i <= N; i++){
        scanf("%d", &temp);

        if (temp == (previousNumber+1)){
            previousNumber = temp;
        }else{
            missingNumber = previousNumber+1;
            N--;

        }
    }
    if(missingNumber != 0){
        printf("%d", missingNumber);
    }else{
        printf("There is no missing number");
    }
    return 0;
}

6.// Check duplicates Number;
#include <stdlib.h>
#include <stdio.h>

int main(){
    int testCaseNumber;
    scanf("%d", &testCaseNumber);
    int allTestCases[testCaseNumber];
    for(int i = 0; i < testCaseNumber; i++){
        scanf("%d", &allTestCases[i]);
    }
    // finding the largest to determine the max size of the new array

    int largest = allTestCases[0];
    for (int i = 0; i < testCaseNumber; i++){
        if(allTestCases[i] > largest){
            largest = allTestCases[i];
        }
    }
    // const int createLargest = largest;

    // int indexBasedTracking[createLargest+1] = {0};
    int *indexBasedTracking = malloc((largest+1 ) * sizeof(int));
    for(int i = 0; i < testCaseNumber; i++){
        indexBasedTracking[allTestCases[i]]++;
    }
    for (int i = 0; i <= largest; i++){
        // printf("%d", indexBasedTracking[i]);
        if(indexBasedTracking[i] > 1){

            printf("Yes, duplicate exists.");
            break;
        }else{
            continue;
        }
        printf("No, duplicate doesn't exists.");

    }

    return 0;
}
```

```
7.#include <stdio.h>

int main(){
    int numberOfTestCase;
    int targetSum;
    scanf("%d %d", &numberOfTestCase, &targetSum);
    int testCases[numberOfTestCase];
    for (int i = 0; i < numberOfTestCase; i++){
        scanf("%d", &testCases[i]);
    }
    int x;
    int y;
    for(int i = 0; i < numberOfTestCase; i++){
        x = testCases[i];
        for(int j=i+1; j < numberOfTestCase; j++){
            y = testCases[j];
            if((x+y) == targetSum){
                printf("(%d, %d)\n", x, y);
            }
        }
    }

    return 0;
}

8.#include <stdio.h>

int main(){
    int numberOfTestCase;
    scanf("%d", &numberOfTestCase);
    int testCases[numberOfTestCase];
    for(int i =0; i < numberOfTestCase; i++){
        scanf("%d", &testCases[i]);
    }
    int zeroCounter = 0;
    for(int i = 0; i < numberOfTestCase; i++){
        if (testCases[i] > 0){
            printf("%d", testCases[i]);
            if((i == (numberOfTestCase-1) && zeroCounter == 0)){
                printf("\n");
            }else{
                printf(", ");
            }
        }else{
            zeroCounter++;
        }
        if(i == (numberOfTestCase-1)){
            for (int j = 0; j < zeroCounter; j++){
                printf("0");
                if (!(j == (zeroCounter-1))){
                    printf(", ");
                }
            }
        }

    }

    return 0;
}


9.#include <stdio.h>

int main(){

    int numberOfTestCase;
    scanf("%d", &numberOfTestCase);
    int testCases[numberOfTestCase];
    for(int i = 0; i < numberOfTestCase; i++){
        scanf("%d", &testCases[i]);
    }
    for (int i = 0; i < numberOfTestCase; i++){
        for (int j = i+1; j < numberOfTestCase; j++){
            if(testCases[i] > testCases[j]){
                if(j == (numberOfTestCase-1)){
                    printf("%d ", testCases[i]);

                }
                continue;
            }else{
                break;
            }

        }
    }
    printf("%d\n", testCases[numberOfTestCase-1]); // last number will always
    return 0;
}

10.#include <stdio.h>
#include <stdlib.h>
int main(){
    int numberOfTestCases;
    scanf("%d", &numberOfTestCases);
    int testCases[numberOfTestCases];
    for (int i = 0; i < numberOfTestCases; i++){
        scanf("%d", &testCases[i]);
    }
    int largestDifference = 0;
    int temp;
    for (int i = 0; i < numberOfTestCases; i++){
        for(int j = i+1; j < numberOfTestCases; j++){
            if(!(testCases[i] < testCases[j])){
                continue;
            }
            // printf("Passed for %d %d\n", testCases[i], testCases[j]);
            temp = (testCases[i] - testCases[j]);

            if(((temp) > largestDifference)*(-1)){
                largestDifference = (testCases[i] - testCases[j])*(-1);
            }
        }
    }
    printf("%d", largestDifference);
}
```

```
 *** STRING ***

1.#include <stdio.h>

int main(){
    char string[100];

    scanf("%s", string);
    int counter = 0;
    for(int i = 0;string[i] != '\0'; i++){
        // printf("%c", string[i]);
        counter++;
    }
    for(int i = counter; i >= 0; i--){
        printf("%c", string[i]);
    }
}

2.#include <stdio.h>
int main(){
    char string[100];
    scanf("%s", string);

    int lengthCounter = 0;
    for (int i = 0; string[i] != '\0'; i++){
        lengthCounter++;
    }
    for (int i = 0, j=lengthCounter-1; i < lengthCounter ;i++,j--){
        // printf("%d %d", i, j);
        if (string[i] == string[j]){
            if(i == (lengthCounter-1)){
                printf("Yes, it's a palindrome.");
            }
            continue;
        }else{
            printf("No, it's not a palindrome.");
            break;  }
    }
    return 0;
}

3.#include <stdio.h>

int main(){
    char word[100];
    scanf("%s", word);
    int vowelCounter = 0;
    int consonantCounter = 0;
    for(int i = 0; word[i]!='\0';i++){
        if(
            word[i] == 'a' || word[i] == 'e' || word[i] == 'i' || word[i] == '
        ){
            vowelCounter++;

        }else{
            consonantCounter++;
        }
    }
    printf("Vowels : %d\n", vowelCounter);
    printf("Consonants : %d", consonantCounter);


}

4.#include <stdio.h>

int main(){

    char string[100];
    scanf("%s", string);
    int length = 0;
    while(string[length] != '\0'){
        length++;
    }
    char storage[100]; // to solve the last comma
    int storageCounter = 0;
    for (int i = 0; i < length ; i++){
        for (int j = i+1; j < length; j++){
            if (string[i] == string[j]){
                storage[storageCounter] = string[i];
                storageCounter++;
                // printf("%c", string[i]);
                // if(i == (length-1)){

                // }else{
                //     printf(", ");

                // }
                break;
            }
        }
    }
    for(int i = 0; i < storageCounter; i++){
        printf("%c", storage[i]);
```

```
            if(i == (storageCounter - 1)){
            }else{
                printf(", ");

            }
        }
    }

    5.#include <stdio.h>

    int main(){
        char string[100];
        fgets(string, 100, stdin);
        for (int i = 0; string[i] != '\0'; i++){
            if(string[i] != ' '){
                printf("%c", string[i]);
            }
        }
    }

    6.//Find the longest word in sentence

    #include <stdio.h>

    int main(){
        // char sentence[100];
        // fgets(sentence, 100, stdin);
        // char sentence[100] = "I love coding";
        char sentence[100] = "I love coding";
        int lengthCounter =0;
        for(int i = 0; sentence[i] != '\0'; i++){
            lengthCounter++;
        }
        printf("Length : %d\n", lengthCounter);
        int start = 0;
        int largestGapSize = 0;
        int gapSize = 0;
        int lastIndex = 0;
        int startIndex;
        for(int i = 0; i < lengthCounter; i++){
            if(i == (lengthCounter-1)){
                start = 1;
            }
            if((sentence[i] == ' ') && (start == 0)){
                startIndex = i;
                start = 1;

            }
            if(sentence[i] != ' '){
                // printf("gap letter : %c %d\n", sentence[i], gapSize);
                gapSize++;
                continue;

            }
            if(((sentence[i] == ' ') || (i == (lengthCounter-1))) && (start == 1 |
                if(gapSize > largestGapSize){
                    printf("Larger Gapsize found  %d > %d, startIndex : %d\n", gap
                    lastIndex = i;
                    largestGapSize = gapSize;
                    gapSize = 0;
                    start = 0;
                    startIndex = i;
                }else{
                    // printf("Gap Size : %d\n", gapSize);
                    gapSize = 0;
                    start = 0;
                    startIndex = i;

                }
                // printf("Gap Found : %d, Start : %d\n", gapSize, start);

            }
        }

        // printf("%d %d", largestGapSize, lastIndex);
        // printf("Start point %d \n", (lastIndex-largestGapSize));
        printf("Largest Gap Size : %d\n", largestGapSize);
        // printf("Last Index : %d\n", lastIndex);
        // printf("Start Index : %d\n", startIndex);
        for(int i = startIndex-largestGapSize; i < startIndex; i++){
            printf("%c", sentence[i]);
        }

    }

    7.#include <stdio.h>

    int main(){
        char str1[100], str2[100];
        scanf("%s %s", str1, str2);
        int weightOfStr1 = 0, weightOfStr2 = 0;

        // checking the length;
        int lenStr1 = 0;
        int lenStr2 = 0;
        for (int i = 0; str1[i] != '\0'; i++){
            lenStr1++;
        }
        for (int i = 0; str2[i] != '\0'; i++){
            lenStr2++;
        }
        if(lenStr1 != lenStr2){
            printf("No, they are not anagrams.");
        }
        for (int i = 0; i < lenStr1; i++){
            weightOfStr1+=(int) str1[i];
            weightOfStr2+=(int) str2[i];
        }
        if(weightOfStr1 == weightOfStr2){
            printf("Yes, they are anagrams.");
        }
    }

    8.#include <stdio.h>
    int main(){
        char sentence[100];
        fgets(sentence, 100, stdin);
        char letterToReplce, replaceWith;
        scanf("%c %c", &letterToReplce, &replaceWith);
        for(int i = 0; sentence[i]!='\0'; i++){
            if (sentence[i] == letterToReplce){
                printf("%c", replaceWith);
            }else{
                printf("%c", sentence[i]);
            }
        }
    }


    9.#include <stdio.h>

    int isExists(char word[100], char letter){
        for(int i = 0; word[i] != '\0'; i++){
            if(word[i] == letter){
                return 1;
            }
        }
        return 0;
    }
    int main(){
        char word[100];
        scanf("%s", word);
        int letterCounter = 0;
        char uniqueLetters[100];
        int uniqueCounter = 0;
        for(int i = 0; word[i] != '\0'; i++){
            if(!isExists(uniqueLetters, word[i])){
                uniqueLetters[uniqueCounter] = word[i];
                uniqueCounter++;
            }
        }
        for(int i = 0; uniqueLetters[i] != '\0'; i++){
            for(int j = 0; word[j] != '\0'; j++){
                if(uniqueLetters[i] == word[j]){
                    letterCounter++;
                }
            }
            printf("%c : %d\n", uniqueLetters[i], letterCounter);
            letterCounter=0;
        }
    }

    10.#include <stdio.h>

    int main() {
        char str_array[100];
        scanf("%s",str_array);

        int string_length = 0;
        while (1) {
            if (str_array[string_length]== '\0') {
                break;
            }
            string_length = string_length+ 1;
        }

        int is_first_print=1;

        for(int current_length = 1; current_length <= string_length; current_lengt
            for (int starting_pos = 0;starting_pos <= string_length - current_leng
                if (is_first_print== 1) {
                    is_first_print = 0;
                } else{
                    printf(",");
                }

                for (int char_pos = starting_pos; char_pos< starting_pos + current
                    char current_char = str_array[char_pos];
                    printf("%c",current_char);
                }
            }
        }
```

```
        return 0;
    }
```

PDF document made with CodePrint using Prism