Labreport

Write a program to find both the largest and smallest elements of an array using only one traversal (both in one loop).

```c
~

#include <stdio.h>

void findMinMax(int *a, int size){
    int max = *a, min = *a;
    for (int i = 0; i < size; i++) {
        if (a[i] > max){
            max = a[i];
        }
        if (a[i]<min){
            min = a[i];
        }
    }

    printf("Greatest %d, Minimum %d\n",max,min);
}
int main(){
    int arr[] = {2,3,4,5,1,6,8}, max = arr[0], min = arr[0];
    for(int i = 1; i < sizeof(arr)/sizeof(arr[0]);i++){
        arr[i] > max? (max = arr[i]):(arr[i] < min? (min = arr[i]): 0);
    }

}
```

```c
~

#include <stdio.h>

int main(){
    int arr[] = {2,3,4,5,1,6,8}, max = arr[0], min = arr[0];
    for(int i = 1; i < sizeof(arr)/sizeof(arr[0]);i++){
        arr[i] > max? (max = arr[i]):(arr[i] < min? (min = arr[i]): 0);
    }
    printf("Greatest %d, Minimum %d\n", max, min);
}
```

```
~

[wizard@archlinux w3]$ gcc main.c && ./a.out
Greatest 8, Minimum 1
[wizard@archlinux w3]$
```

Write a program to check whether two given strings are an anagram.

```c
~


#include <stdio.h>
#include <string.h>

int main(){
    char str1[] = "eleven plus two", str2[] = "twelve plus one";
    int sum;
    for (int i = 0; i < strlen(str1); i++) {
        sum ^= str1[i] ^ str2[i];
    }

    printf("sum: %d", sum);


}
```

```c
~


#include <stdio.h>
int main(){
    char str1[] = "eleven plus two", str2[] = "twelve plus one";
    int sum;
    for (int i = 0;str1[i] || str2[i]; i++) {
        sum ^= str1[i] ^ str2[i];
    }
    printf((sum == 0)? "Anagram": "Not an anagram");
}
```

```
~


[wizard@archlinux w3]$ gcc main.c
[wizard@archlinux w3]$ ./a.out
Anagram
[wizard@archlinux w3]$
```

Write a program to print all unique elements in an array. For example: a[] = {1,2,4,8,4,2,4,9,6} answer : 1,2,4,8,9,6.

```c
~


#include <stdio.h>

int main() {
    int arr[] = {1, 2, 4, 8, 4, 2, 4, 9, 6}, hash[10];
    for (int i = 0; i < sizeof(arr)/sizeof(arr[0]); i++) !hash[arr[i]] ? (printf("%d ", arr[i]), hash[arr[i]]
= 1) : 0;
    return 0;
}
```

```
~


[wizard@archlinux w3]$ gcc main.c
[wizard@archlinux w3]$ ./a.out
1 2 4 8 9
[wizard@archlinux w3]$
```

Write a program to sort an array of elements in ascending order.

```c
~

#include <stdio.h>

int main() {
    int arr[] = {10,8,6,2,1,3,4};
    int n = sizeof(arr) / sizeof(arr[0]);
    for (int i = 0; i < n - 1; i++) {
        int smallest = i;
        for (int j = i + 1; j < n; j++) {
            if (arr[j] < arr[smallest]) {
                smallest = j;
            }
        }
        arr[i] = (arr[i] ^= arr[smallest], arr[smallest] ^= arr[i], arr[i] ^ arr[smallest]);
    }
    for (int i = 0; i < n; i++) printf("%d ", arr[i]);
    return 0;
}
```

```
~

[wizard@archlinux w3]$ gcc main.c
[wizard@archlinux w3]$ ./a.out
1 2 3 4 6 8 10
[wizard@archlinux w3]$
```

Write a program to count and find the sum of all numbers in the array which are divisible by 5 but neither by 2 nor by 3. Also, print the indices of these numbers.

```c
~

#include <stdio.h>

int main() {
    int arr[] = {5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75}, sum=0;
    for (int i = 0; i < sizeof(arr)/sizeof(arr[0]); i++)
        if (arr[i] % 5 == 0 && arr[i] % 3 != 0 && arr[i] %2 != 0) { sum += arr[i]; printf("%d ",i); }
    printf("\nSum :%d ",sum);
    return 0;
}
```

WAP reads two 2-D arrays of user defined dimensions, adds the corresponding elements and displays the result on the screen. Include error handling for unequal dimensions. (For eg: a 2x2 array and 2x3 array cannot be added because of unequal dimensions.)

```c
~

#include <stdio.h>
#include <stdlib.h>

int main() {
    int r, c;
    printf("Enter rows and cols: ");
    scanf("%d-%d", &r, &c);
    int *arr1 = malloc(r * c * sizeof(int)), *arr2 = malloc(r * c * sizeof(int));

    printf("First matrix: \n");
    for (int i = 0; i < r * c; i++) scanf("%d", &arr1[i]);

    printf("second matrix: \n");
    for (int i = 0; i < r * c; i++) scanf("%d", &arr2[i]);

    for (int i = 0; i < r * c; i++)
        printf("%d%c", arr1[i] + arr2[i], (i + 1) % c == 0 ? '\n' : ' ');

    free(arr1), free(arr2);
    return 0;
}
```

```
~

Enter rows and cols: 2-2
First matrix:
1
2
3
4
second matrix:
5
6
7
8
6 8
10 12
[wizard@archlinux w3]$
```