Assignment 2 (1D Array)

Q1. WAP to increase every student mark by 5 & then print the updated

array.

#include <stdio.h>

int main()

{

int m[] = {40, 55, 60, 50, 45};

int n = 5;

for (int i = 0; i < n; i++) {

m[i] += 5;

}

for (int i = 0; i < n; i++)

{

printf("%d ", m[i]);

}

return 0;

}

Q2. WAP to print grade of students as per their marks given in an array.

(>=75—A grade, 74 to 60--B Grade, 59 to 40--C grade below 40--D grade).

#include <stdio.h>

int main() {

int m[] = {88, 66, 54, 83, 41};

int n = 5;

for (int i = 0; i < n; i++) {

if (m[i] >= 75) {

printf("A ");

} else if (m[i] >= 60) {

printf("B ");

} else if (m[i] >= 40) {

printf("C ");

} else {

printf("D ");

}

}

return 0;

}

Q3. WAP to find who scored first “99” in an array marks.

#include <stdio.h>

int main() {

int marks[] = {38, 59, 99, 69, 99};

int n = 5;

for (int i = 0; i < n; i++) {

if (marks[i] == 99) {

printf("First 99 found at index %d", i);

break;

}

}

return 0;

}

Q4. WAP to find Who & how many students have scored 99 in an array

Marks.

#include <stdio.h>

int main() {

int marks[] = {88, 99, 35, 99, 99};

int n = 5, count = 0;

for (int i = 0; i < n; i++) {

if (marks[i] == 99) {

printf("99 found at index %d\n", i);

count++;

}

}

printf("Total students with 99: %d", count);

return 0;

}

Q5. WAP to find sum of all scores in Marks array.

#include <stdio.h>

int main() {

int marks[] = {58, 79, 65, 99, 99};

int n = 5, sum = 0;

for (int i = 0; i < n; i++) {

sum += marks[i];

}

printf("Sum of all scores: %d", sum);

return 0;

}

Q6. WAP to find average score of the Marks array.

#include <stdio.h>

int main() {

int marks[] = {78, 99, 55, 99, 39};

int n = 5, sum = 0;

for (int i = 0; i < n; i++) {

sum += marks[i];

}

float average = sum / (float)n;

printf("Average score: %.2f", average);

return 0;

}

Q7. WAP to check whether score is even or odd in an array.

#include <stdio.h>

int main() {

int m[] = {17, 100, 98, 67, 49};

int n = 5;

for (int i = 0; i < n; i++) {

if (m[i] % 2 == 0) {

printf("%d is even\n", m[i]);

} else {

printf("%d is odd\n", m[i]);

}

}

return 0;

}

Q8. WAP to find maximum & minimum score in the Marks array.

#include <stdio.h>

int main() {

int marks[] = {78, 99, 55, 100, 39};

int n = 5;

int max = marks[0], min = marks[0];

for (int i = 1; i < n; i++) {

if (marks[i] > max) {

max = marks[i];

}

if (marks[i] < min) {

min = marks[i];

}

}

printf("Maximum score: %d\n", max);

printf("Minimum score: %d", min);

return 0;

}

Q9. WAP to find a peak element which is not smaller than its neighbors.

#include <stdio.h>

int main() {

int m[] = {55, 93, 65, 100, 49};

int n = 5;

for (int i = 0; i < n; i++) {

if ((i == 0 || m[i] >= m[i - 1]) && (i == n - 1 || m[i]

>= m[i + 1])) {

printf("Peak element: %d", m[i]);

break;

}

}

return 0;

}

Q10. WAP to count prime numbers in an array.

#include <stdio.h>

int main() {

int arr[] = {13, 17,11 , 14, 20};

int size = 5;

int count = 0;

for (int i = 0; i < size; i++) {

int n = arr[i], isPrime = 1;

if (n <= 1) isPrime = 0;

for (int j = 2; j < n; j++) {

if (n % j == 0) {

isPrime = 0;

break;

}

}

if (isPrime) count++;

}

printf("%d\n", count);

return 0;

}

Q11. WAP to implement Insert -Front, any position in between & end in

an array. Print the array before insert & after insert.

#include <stdio.h>

int main() {

int arr[100] = {10, 20, 30, 40, 50};

int size = 5;

int pos, val, i;

printf("Original Array: ");

for (i = 0; i < size; i++) {

printf("%d ", arr[i]);

}

printf("\n");

val = 5;

for (i = size; i > 0; i--) {

arr[i] = arr[i - 1];

}

arr[0] = val;

size++;

printf("After Insert at Front: ");

for (i = 0; i < size; i++) {

printf("%d ", arr[i]);

}

printf("\n");

pos = 3;

val = 25;

for (i = size; i > pos - 1; i--) {

arr[i] = arr[i - 1];

}

arr[pos - 1] = val;

size++;

printf("After Insert at Position %d: ", pos);

for (i = 0; i < size; i++) {

printf("%d ", arr[i]);

}

printf("\n");

val = 60;

arr[size] = val;

size++;

printf("After Insert at End: ");

for (i = 0; i < size; i++) {

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

Q12. WAP to implement delete-Front, any position in between & end in

an array. Print the array before delete & after delete.

#include <stdio.h>

int main() {

int arr[100] = {10, 20, 30, 40, 50};

int size = 5, pos, i;

printf("Original Array: ");

for (i = 0; i < size; i++) {

printf("%d ", arr[i]);

}

printf("\n");

for (i = 0; i < size - 1; i++) {

arr[i] = arr[i + 1];

}

size--;

printf("After Delete at Front: ");

for (i = 0; i < size; i++) {

printf("%d ", arr[i]);

}

printf("\n");

pos = 2;

for (i = pos - 1; i < size - 1; i++) {

arr[i] = arr[i + 1];

}

size--;

printf("After Delete at Position %d: ", pos);

for (i = 0; i < size; i++) {

printf("%d ", arr[i]);

}

printf("\n");

size--;

printf("After Delete at End: ");

for (i = 0; i < size; i++) {

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

Q13. Given an array, the task is to cyclically rotate the array clockwise

by one time.

Examples:

Input: arr[] = {1, 2, 3, 4, 5}

Output: arr[] = {5, 1, 2, 3, 4}

Input: arr[] = {2, 3, 4, 5, 1}

Output: {1, 2, 3, 4, 5}

#include <stdio.h>

int main() {

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

int size = 5, last, i;

last = arr[size - 1];

for (i = size - 1; i > 0; i--) {

arr[i] = arr[i - 1];

}

arr[0] = last;

for (i = 0; i < size; i++) {

printf("%d ", arr[i]);

}

printf("\n");

return 0;

}

Q14. Given an array of n integers. The task is to print the duplicates in

the given array.

If there are no duplicates then print -1.

Examples:

Input: {2, 10,10, 100, 2, 10, 11,2,11,2}

Output: 2 10 11

Input: {5, 40, 1, 40, 100000, 1, 5, 1}

Output: 5 40 1

#include <stdio.h>

int main() {

int arr[] = {2, 10, 10, 100, 2, 10, 11, 2, 11, 2};

int size = 10, i, j, flag = 0;

for (i = 0; i < size; i++) {

for (j = i + 1; j < size; j++) {

if (arr[i] == arr[j]) {

printf("%d ", arr[i]);

flag = 1;

break;

}

}

for (j = 0; j < i; j++) {

if (arr[i] == arr[j]) break;

}

}

if (flag == 0) printf("-1\n");

return 0;

}