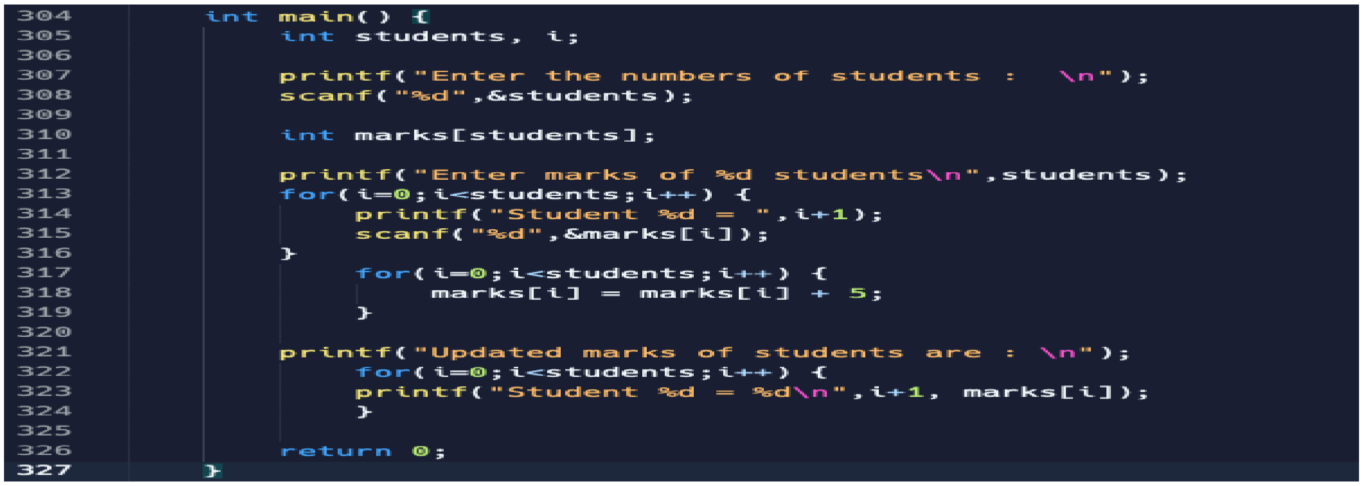
Assignment-2 (FOCP-1)

1. **WAP to increase every student mark by 5 & then print the updated array.**

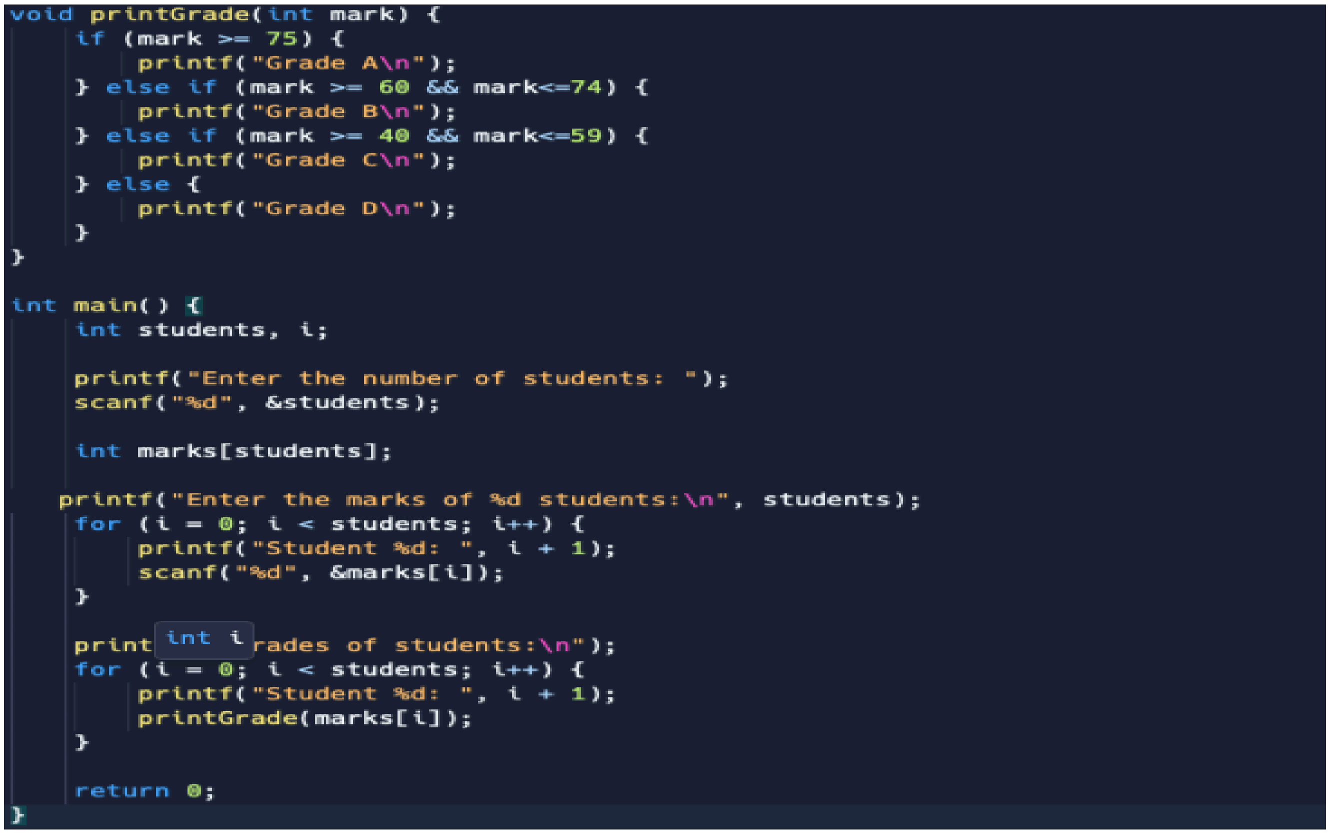
```
int main() {
      int students, i;

      printf("Enter the numbers of students :  \n");
      scanf("%d",&students);

      int marks[students];

      printf("Enter marks of %d students\n",students);
      for(i=0;i<students;i++) {
         printf("Student %d = ",i+1);
         scanf("%d",&marks[i]);
      }
         for(i=0;i<students;i++) {
            marks[i] = marks[i] + 5;
         }

      printf("Updated marks of students are : \n");
         for(i=0;i<students;i++) {
         printf("Student %d = %d\n",i+1, marks[i]);
         }

      return 0;
   }
```



```
int main() {
    int students, i;

    printf("Enter the numbers of students :   \n");
    scanf("%d",&students);

    int marks[students];

    printf("Enter marks of %d students\n",students);
    for(i=0;i<students;i++) {
        printf("Student %d = ",i+1);
        scanf("%d",&marks[i]);
    }
        for(i=0;i<students;i++) {
            marks[i] = marks[i] + 5;
        }

    printf("Updated marks of students are : \n");
        for(i=0;i<students;i++) {
        printf("Student %d = %d\n",i+1, marks[i]);
        }

    return 0;
}
```

2. WAP to print grades of students as per their marks given in a n array.

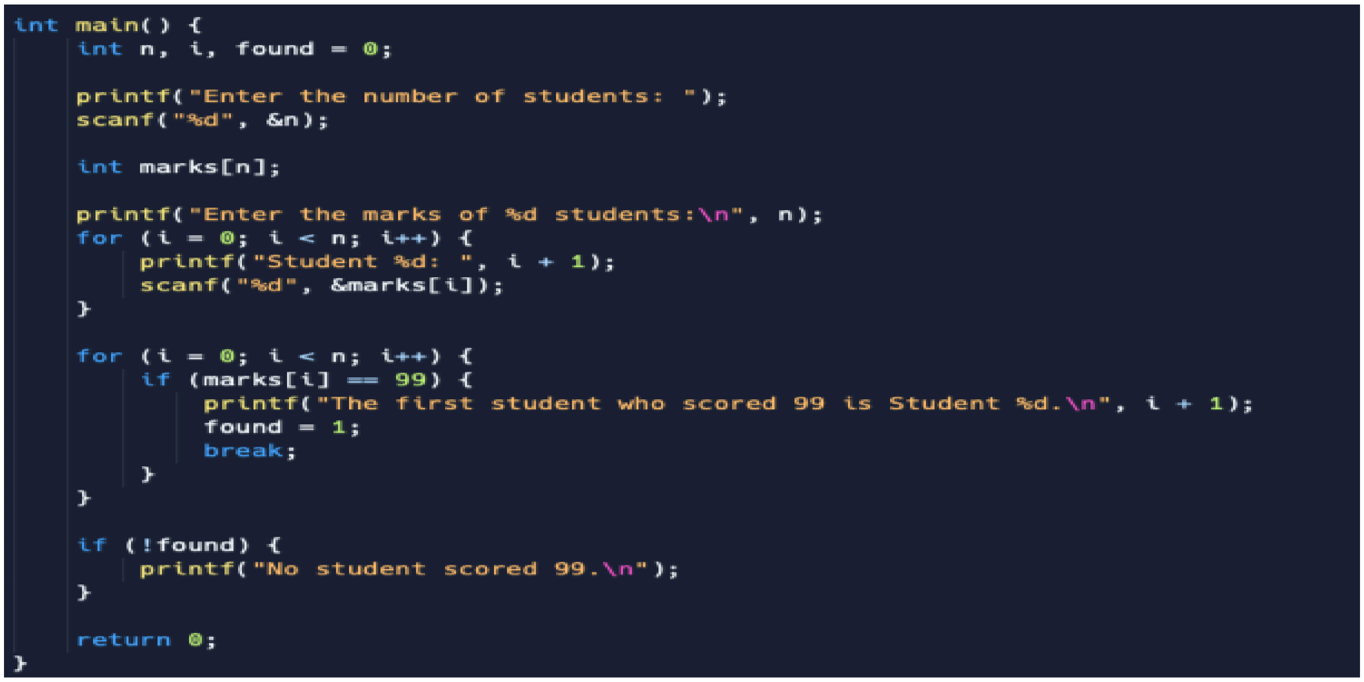
```
void printGrade(int mark) {
    if (mark >= 75) {
        printf("Grade A\n");
    } else if (mark >= 60 && mark<=74) {
        printf("Grade B\n");
    } else if (mark >= 40 && mark<=59) {
        printf("Grade C\n");
    } else {
        printf("Grade D\n");
    }
}

int main() {
    int students, i;

    printf("Enter the number of students: ");
    scanf("%d", &students);

    int marks[students];

   printf("Enter the marks of %d students:\n", students);
    for (i = 0; i < students; i++) {
        printf("Student %d: ", i + 1);
        scanf("%d", &marks[i]);
    }

    printf("\nGrades of students:\n");
    for (i = 0; i < students; i++) {
        printf("Student %d: ", i + 1);
        printGrade(marks[i]);
    }

    return 0;
}
```



```
void printGrade(int mark) {
    if (mark >= 75) {
        printf("Grade A\n");
    } else if (mark >= 60 && mark<=74) {
        printf("Grade B\n");
    } else if (mark >= 40 && mark<=59) {
        printf("Grade C\n");
    } else {
        printf("Grade D\n");
    }
}

int main() {
    int students, i;

    printf("Enter the number of students: ");
    scanf("%d", &students);

    int marks[students];

   printf("Enter the marks of %d students:\n", students);
    for (i = 0; i < students; i++) {
        printf("Student %d: ", i + 1);
        scanf("%d", &marks[i]);
    }

    print int i rades of students:\n");
    for (i = 0; i < students; i++) {
        printf("Student %d: ", i + 1);
        printGrade(marks[i]);
    }

    return 0;
}
```

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

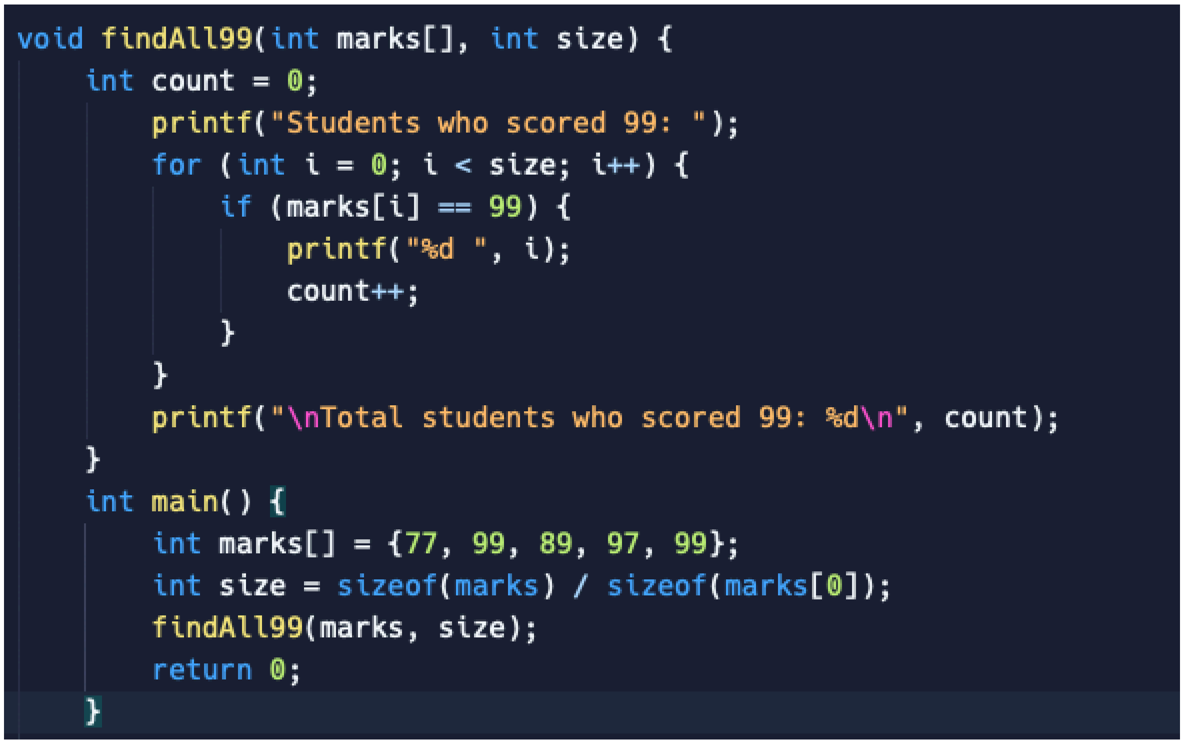
```
int main() {
    int n, i, found = 0;

    printf("Enter the number of students: ");
    scanf("%d", &n);

    int marks[n];

    printf("Enter the marks of %d students:\n", n);
    for (i = 0; i < n; i++) {
        printf("Student %d: ", i + 1);
        scanf("%d", &marks[i]);
    }

    for (i = 0; i < n; i++) {
        if (marks[i] == 99) {
            printf("The first student who scored 99 is Student %d.\n", i + 1);
            found = 1;
            break;
        }
    }

    if (!found) {
        printf("No student scored 99.\n");
    }

    return 0;
}
```



```
int main() {
    int n, i, found = 0;

    printf("Enter the number of students: ");
    scanf("%d", &n);

    int marks[n];

    printf("Enter the marks of %d students:\n", n);
    for (i = 0; i < n; i++) {
        printf("Student %d: ", i + 1);
        scanf("%d", &marks[i]);
    }

    for (i = 0; i < n; i++) {
        if (marks[i] == 99) {
            printf("The first student who scored 99 is Student %d.\n", i + 1);
            found = 1;
            break;
        }
    }

    if (!found) {
        printf("No student scored 99.\n");
    }

    return 0;
}
```

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

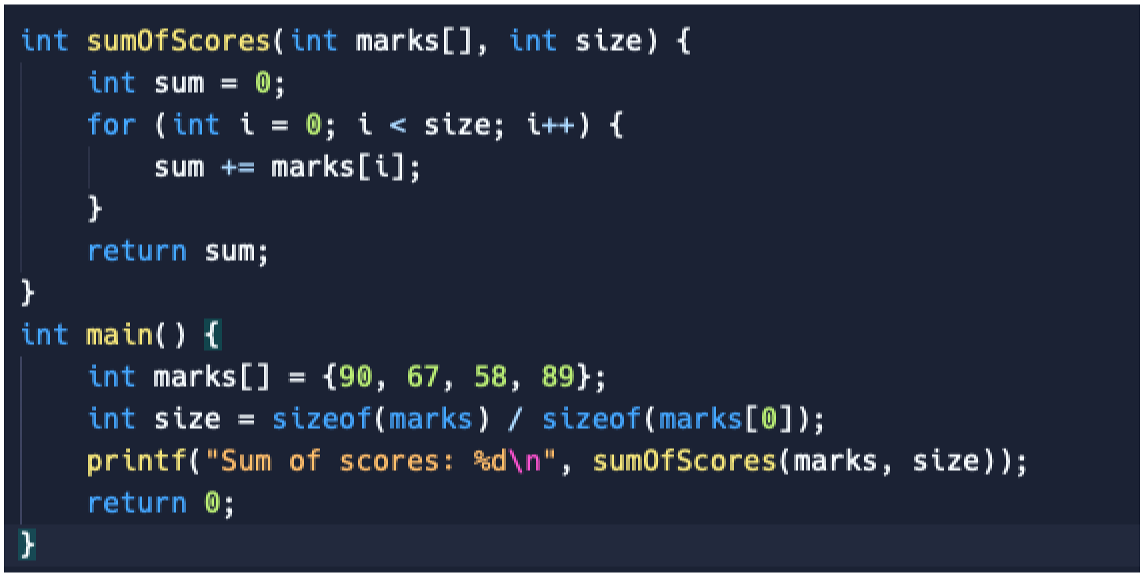
```
void findAll99(int marks[], int size) {
    int count = 0;
        printf("Students who scored 99: ");
        for (int i = 0; i < size; i++) {
            if (marks[i] == 99) {
                printf("%d ", i);
                count++;
            }
        }
        printf("\nTotal students who scored 99: %d\n", count);
    }
    int main() {
        int marks[] = {77, 99, 89, 97, 99};
        int size = sizeof(marks) / sizeof(marks[0]);
        findAll99(marks, size);
        return 0;
    }
```



```
void findAll99(int marks[], int size) {
    int count = 0;
        printf("Students who scored 99: ");
        for (int i = 0; i < size; i++) {
            if (marks[i] == 99) {
                printf("%d ", i);
                count++;
            }
        }
        printf("\nTotal students who scored 99: %d\n", count);
    }
    int main() {
        int marks[] = {77, 99, 89, 97, 99};
        int size = sizeof(marks) / sizeof(marks[0]);
        findAll99(marks, size);
        return 0;
    }
```

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

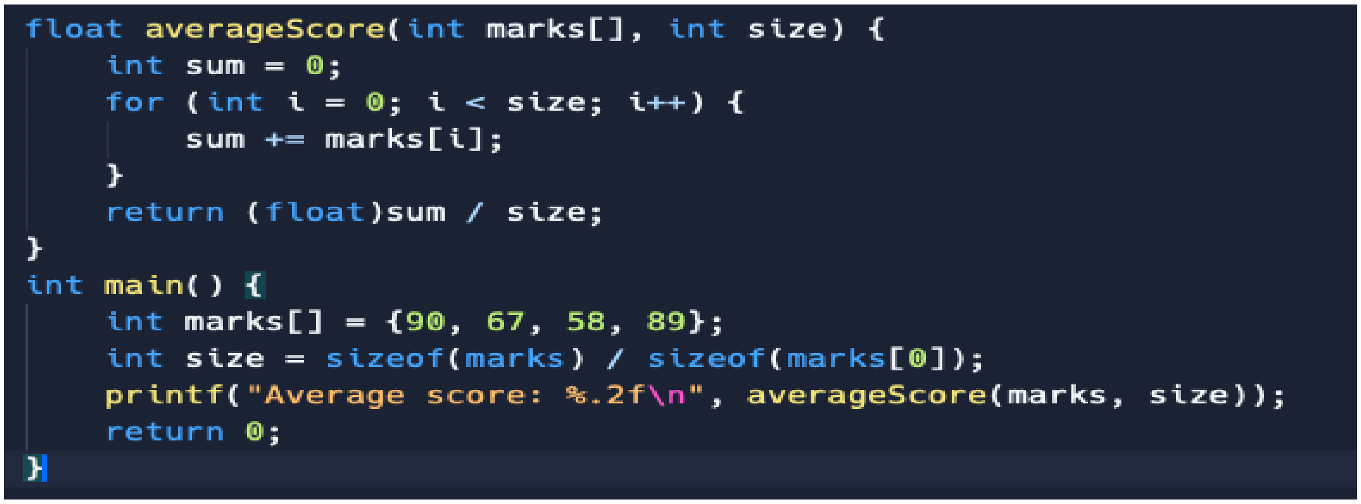
```
int sumOfScores(int marks[], int size) {
    int sum = 0;
    for (int i = 0; i < size; i++) {
        sum += marks[i];
    }
    return sum;
}
int main() {
    int marks[] = {90, 67, 58, 89};
    int size = sizeof(marks) / sizeof(marks[0]);
    printf("Sum of scores: %d\n", sumOfScores(marks, size));
    return 0;
}
```



```c
int sumOfScores(int marks[], int size) {
    int sum = 0;
    for (int i = 0; i < size; i++) {
        sum += marks[i];
    }
    return sum;
}
int main() {
    int marks[] = {90, 67, 58, 89};
    int size = sizeof(marks) / sizeof(marks[0]);
    printf("Sum of scores: %d\n", sumOfScores(marks, size));
    return 0;
}
```

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

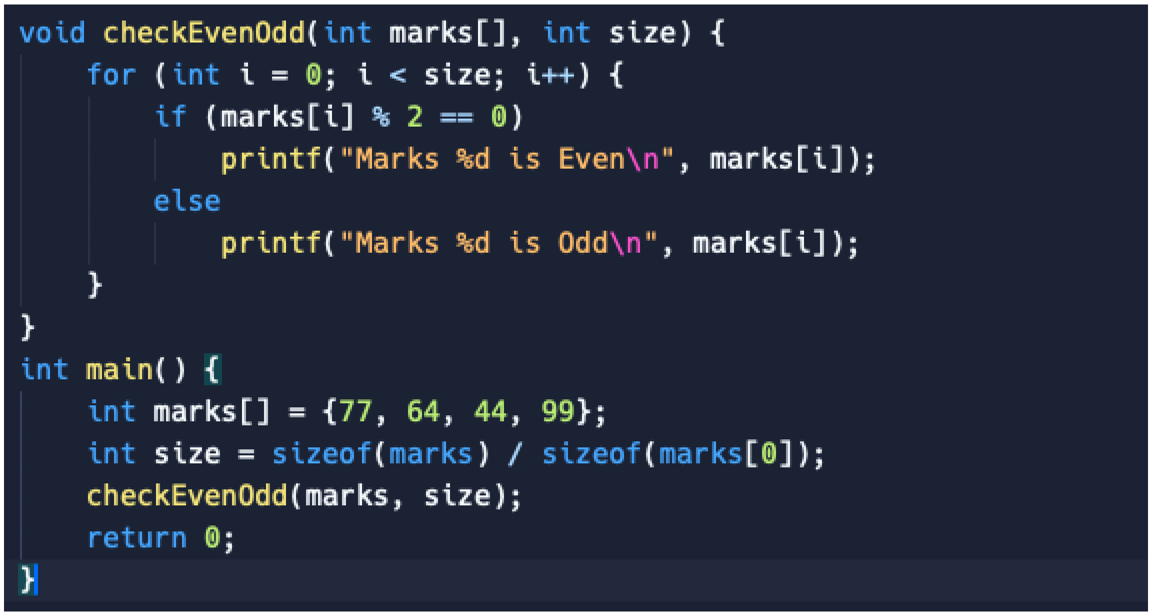
```c
float averageScore(int marks[], int size) {
    int sum = 0;
    for (int i = 0; i < size; i++) {
        sum += marks[i];
    }
    return (float)sum / size;
}
int main() {
    int marks[] = {90, 67, 58, 89};
    int size = sizeof(marks) / sizeof(marks[0]);
    printf("Average score: %.2f\n", averageScore(marks, size));
    return 0;
}
```



```c
float averageScore(int marks[], int size) {
    int sum = 0;
    for (int i = 0; i < size; i++) {
        sum += marks[i];
    }
    return (float)sum / size;
}
int main() {
    int marks[] = {90, 67, 58, 89};
    int size = sizeof(marks) / sizeof(marks[0]);
    printf("Average score: %.2f\n", averageScore(marks, size));
    return 0;
}
```

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

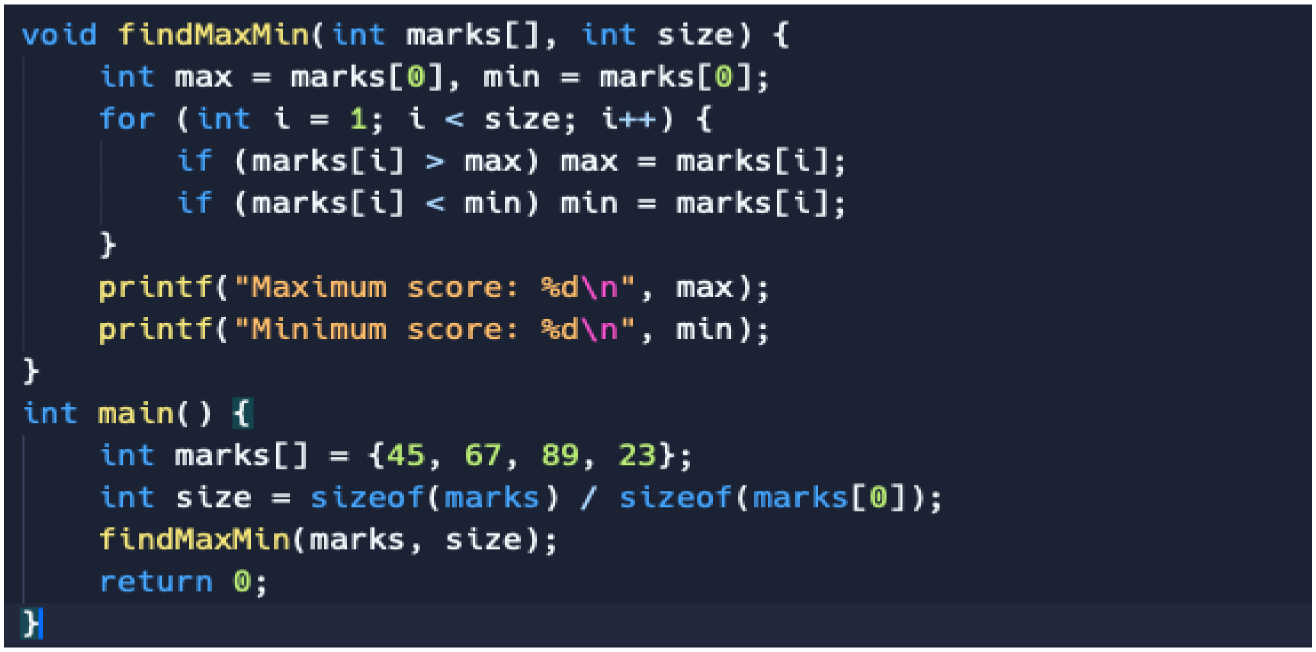
```
void checkEvenOdd(int marks[], int size) {
    for (int i = 0; i < size; i++) {
        if (marks[i] % 2 == 0)
            printf("Marks %d is Even\n", marks[i]);
        else
            printf("Marks %d is Odd\n", marks[i]);
    }
}
int main() {
    int marks[] = {77, 64, 44, 99};
    int size = sizeof(marks) / sizeof(marks[0]);
    checkEvenOdd(marks, size);
    return 0;
}
```



```
void checkEvenOdd(int marks[], int size) {
    for (int i = 0; i < size; i++) {
        if (marks[i] % 2 == 0)
            printf("Marks %d is Even\n", marks[i]);
        else
            printf("Marks %d is Odd\n", marks[i]);
    }
}
int main() {
    int marks[] = {77, 64, 44, 99};
    int size = sizeof(marks) / sizeof(marks[0]);
    checkEvenOdd(marks, size);
    return 0;
}
```

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

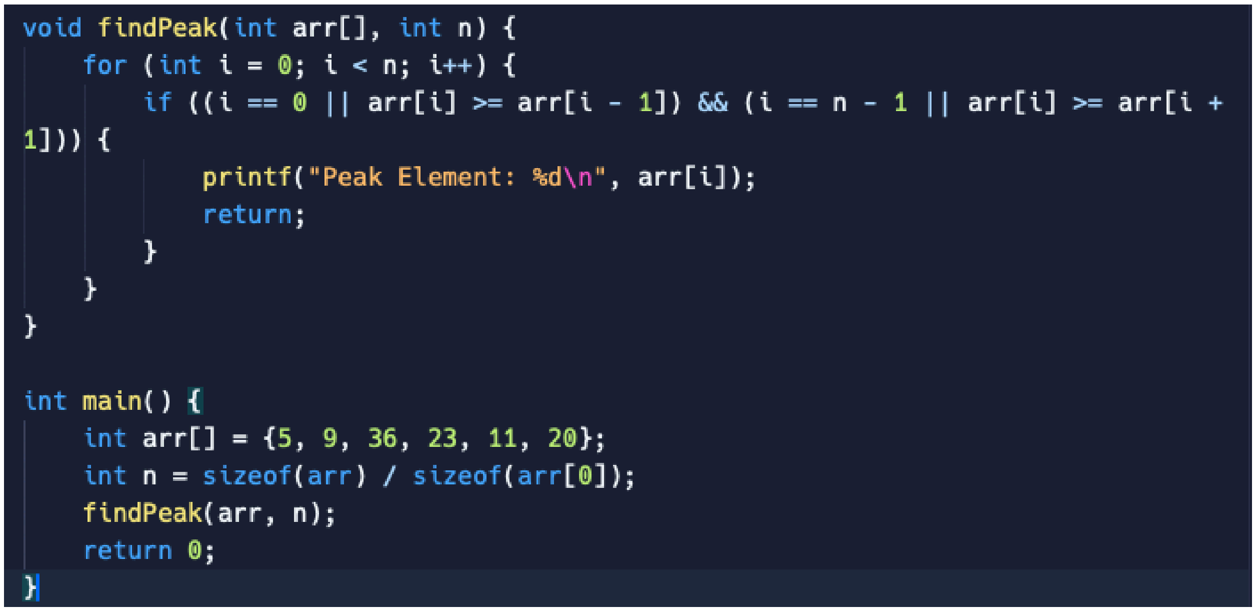
```
void findMaxMin(int marks[], int size) {
    int max = marks[0], min = marks[0];
    for (int i = 1; i < size; i++) {
        if (marks[i] > max) max = marks[i];
        if (marks[i] < min) min = marks[i];
    }
    printf("Maximum score: %d\n", max);
    printf("Minimum score: %d\n", min);
}
int main() {
    int marks[] = {45, 67, 89, 23};
    int size = sizeof(marks) / sizeof(marks[0]);
    findMaxMin(marks, size);
    return 0;
}
```



```
void findMaxMin(int marks[], int size) {
    int max = marks[0], min = marks[0];
    for (int i = 1; i < size; 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\n", min);
}
int main() {
    int marks[] = {45, 67, 89, 23};
    int size = sizeof(marks) / sizeof(marks[0]);
    findMaxMin(marks, size);
    return 0;
}
```

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

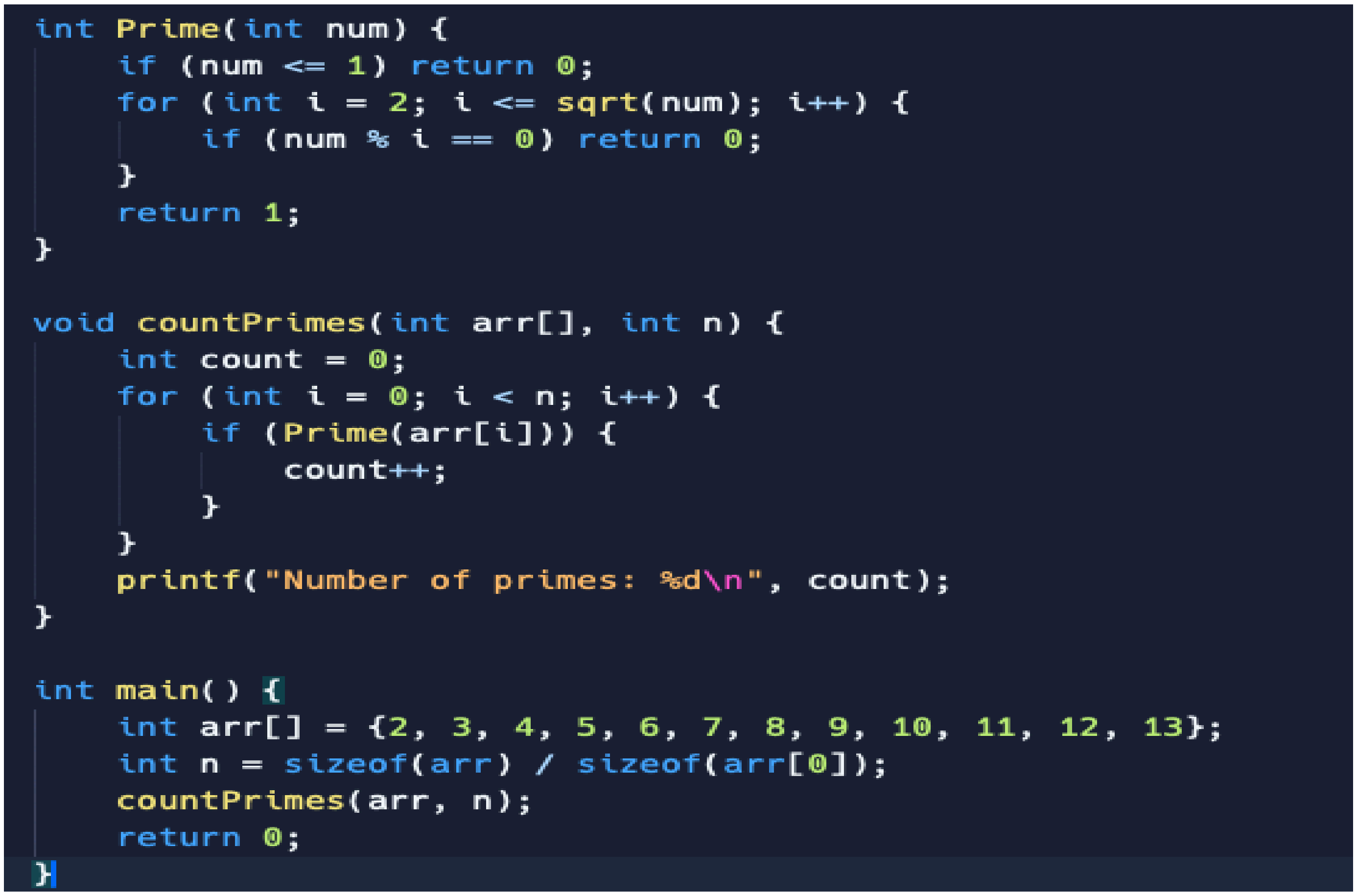
```
void findPeak(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        if ((i == 0 || arr[i] >= arr[i - 1]) && (i == n - 1 || arr[i] >= arr[i + 1])) {
            printf("Peak Element: %d\n", arr[i]);
            return;
        }
    }
}

int main() {
    int arr[] = {5, 9, 36, 23, 11, 20};
    int n = sizeof(arr) / sizeof(arr[0]);
    findPeak(arr, n);
    return 0;
}
```



```c
void findPeak(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        if ((i == 0 || arr[i] >= arr[i - 1]) && (i == n - 1 || arr[i] >= arr[i +
1])) {
            printf("Peak Element: %d\n", arr[i]);
            return;
        }
    }
}

int main() {
    int arr[] = {5, 9, 36, 23, 11, 20};
    int n = sizeof(arr) / sizeof(arr[0]);
    findPeak(arr, n);
    return 0;
}
```

10. WAP to count prime numbers in an array.

```c
int Prime(int num) {
    if (num <= 1) return 0;
    for (int i = 2; i <= sqrt(num); i++) {
        if (num % i == 0) return 0;
    }
    return 1;
}

void countPrimes(int arr[], int n) {
    int count = 0;
    for (int i = 0; i < n; i++) {
        if (Prime(arr[i])) {
            count++;
        }
    }
    printf("Number of primes: %d\n", count);
}

int main() {
    int arr[] = {2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13};
    int n = sizeof(arr) / sizeof(arr[0]);
    countPrimes(arr, n);
    return 0;
}
```

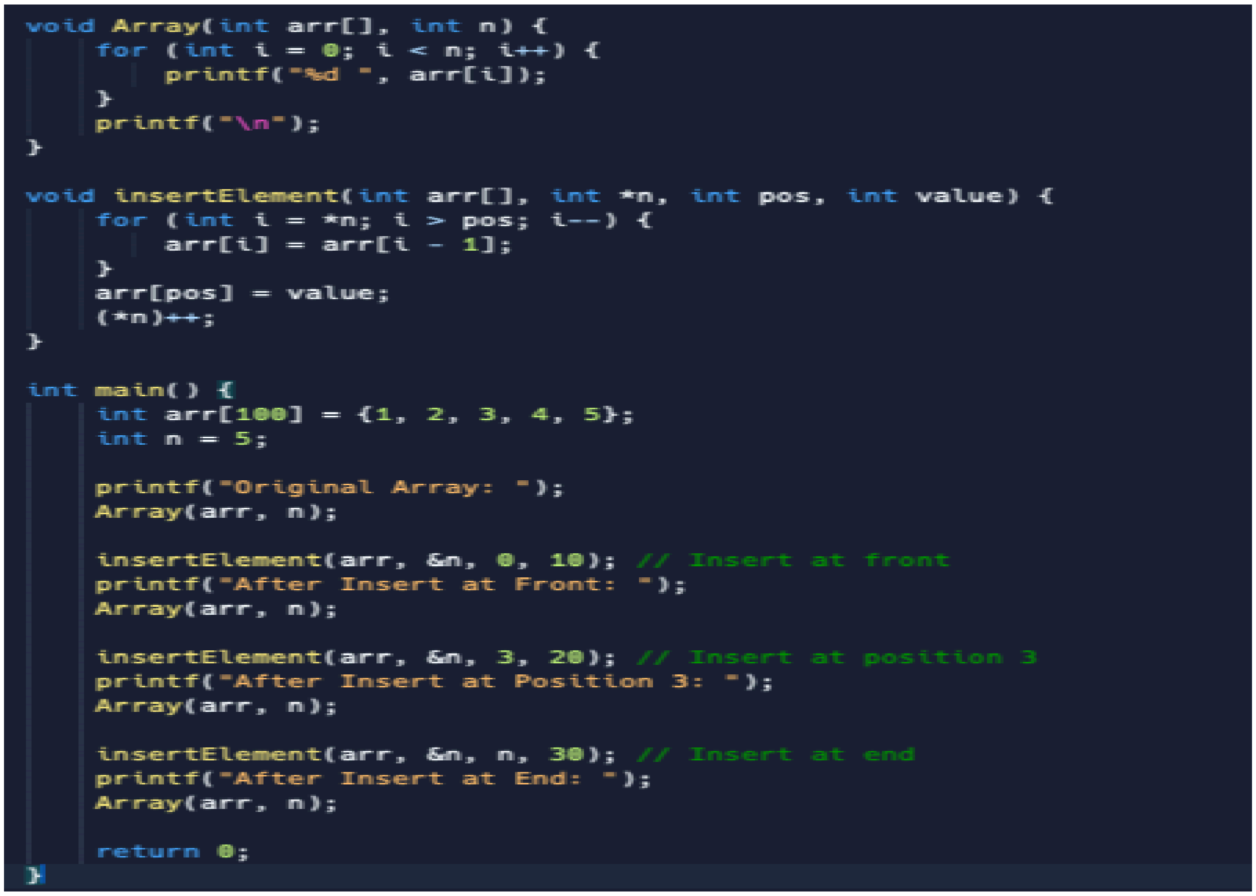


```
int Prime(int num) {
    if (num <= 1) return 0;
    for (int i = 2; i <= sqrt(num); i++) {
        if (num % i == 0) return 0;
    }
    return 1;
}

void countPrimes(int arr[], int n) {
    int count = 0;
    for (int i = 0; i < n; i++) {
        if (Prime(arr[i])) {
            count++;
        }
    }
    printf("Number of primes: %d\n", count);
}

int main() {
    int arr[] = {2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13};
    int n = sizeof(arr) / sizeof(arr[0]);
    countPrimes(arr, n);
    return 0;
}
```

11. WAP to implement Insert - Front, any position in between & end in an array. Print the array before insert & after insert.

```
void Array(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

void insertElement(int arr[], int *n, int pos, int value) {
    for (int i = *n; i > pos; i--) {
        arr[i] = arr[i - 1];
    }
    arr[pos] = value;
    (*n)++;
}

int main() {
```

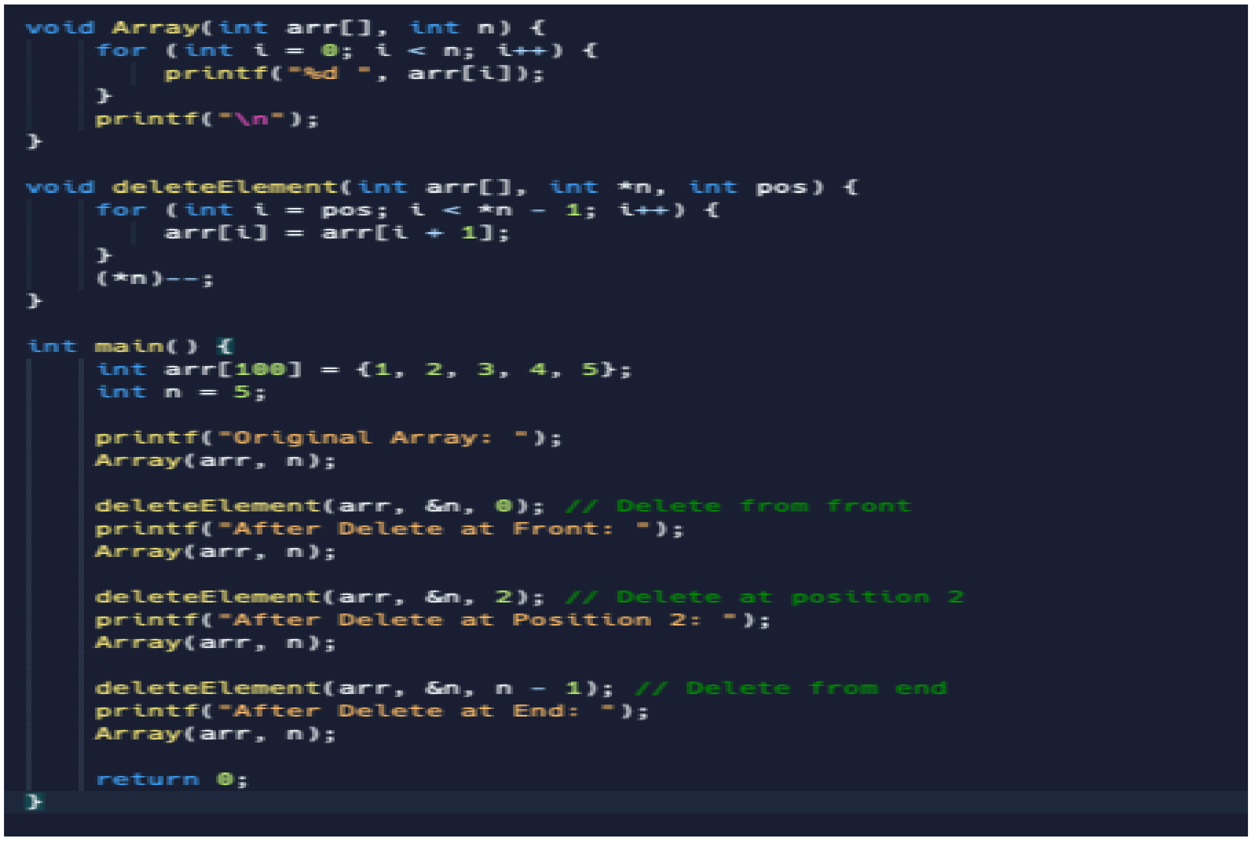
```c
    int arr[100] = {1, 2, 3, 4, 5};
    int n = 5;

    printf("Original Array: ");
    Array(arr, n);

    insertElement(arr, &n, 0, 10); // Insert at front
    printf("After Insert at Front: ");
    Array(arr, n);

    insertElement(arr, &n, 3, 20); // Insert at position 3
    printf("After Insert at Position 3: ");
    Array(arr, n);

    insertElement(arr, &n, n, 30); // Insert at end
    printf("After Insert at End: ");
    Array(arr, n);

    return 0;
}
```



```c
void Array(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

void insertElement(int arr[], int *n, int pos, int value) {
    for (int i = *n; i > pos; i--) {
        arr[i] = arr[i - 1];
    }
    arr[pos] = value;
    (*n)++;
}

int main() {
    int arr[100] = {1, 2, 3, 4, 5};
    int n = 5;

    printf("Original Array: ");
    Array(arr, n);

    insertElement(arr, &n, 0, 10); // Insert at front
    printf("After Insert at Front: ");
    Array(arr, n);

    insertElement(arr, &n, 3, 20); // Insert at position 3
    printf("After Insert at Position 3: ");
    Array(arr, n);

    insertElement(arr, &n, n, 30); // Insert at end
    printf("After Insert at End: ");
    Array(arr, n);

    return 0;
}
```

12. WAP to implement delete - Front, any position in between & end in an array. Print the array before delete & after delete.

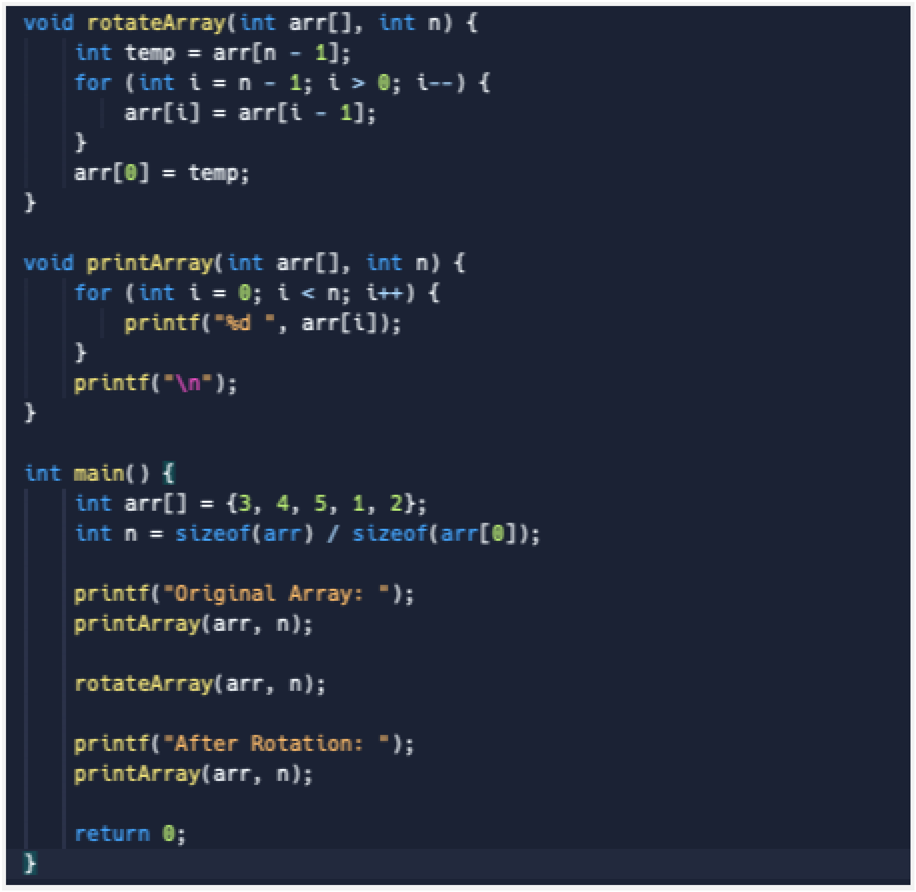
```
void Array(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

void deleteElement(int arr[], int *n, int pos) {
    for (int i = pos; i < *n - 1; i++) {
        arr[i] = arr[i + 1];
    }
    (*n)--;
}

int main() {
    int arr[100] = {1, 2, 3, 4, 5};
    int n = 5;

    printf("Original Array: ");
    Array(arr, n);

    deleteElement(arr, &n, 0); // Delete from front
    printf("After Delete at Front: ");
    Array(arr, n);

    deleteElement(arr, &n, 2); // Delete at position 2
    printf("After Delete at Position 2: ");
    Array(arr, n);

    deleteElement(arr, &n, n - 1); // Delete from end
    printf("After Delete at End: ");
    Array(arr, n);

    return 0;
}
```



```c
void Array(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

void deleteElement(int arr[], int *n, int pos) {
    for (int i = pos; i < *n - 1; i++) {
        arr[i] = arr[i + 1];
    }
    (*n)--;
}

int main() {
    int arr[100] = {1, 2, 3, 4, 5};
    int n = 5;

    printf("Original Array: ");
    Array(arr, n);

    deleteElement(arr, &n, 0); // Delete from front
    printf("After Delete at Front: ");
    Array(arr, n);

    deleteElement(arr, &n, 2); // Delete at position 2
    printf("After Delete at Position 2: ");
    Array(arr, n);

    deleteElement(arr, &n, n - 1); // Delete from end
    printf("After Delete at End: ");
    Array(arr, n);

    return 0;
}
```

13. Given an array, the task is to cyclically rotate the array clockwise by one time.

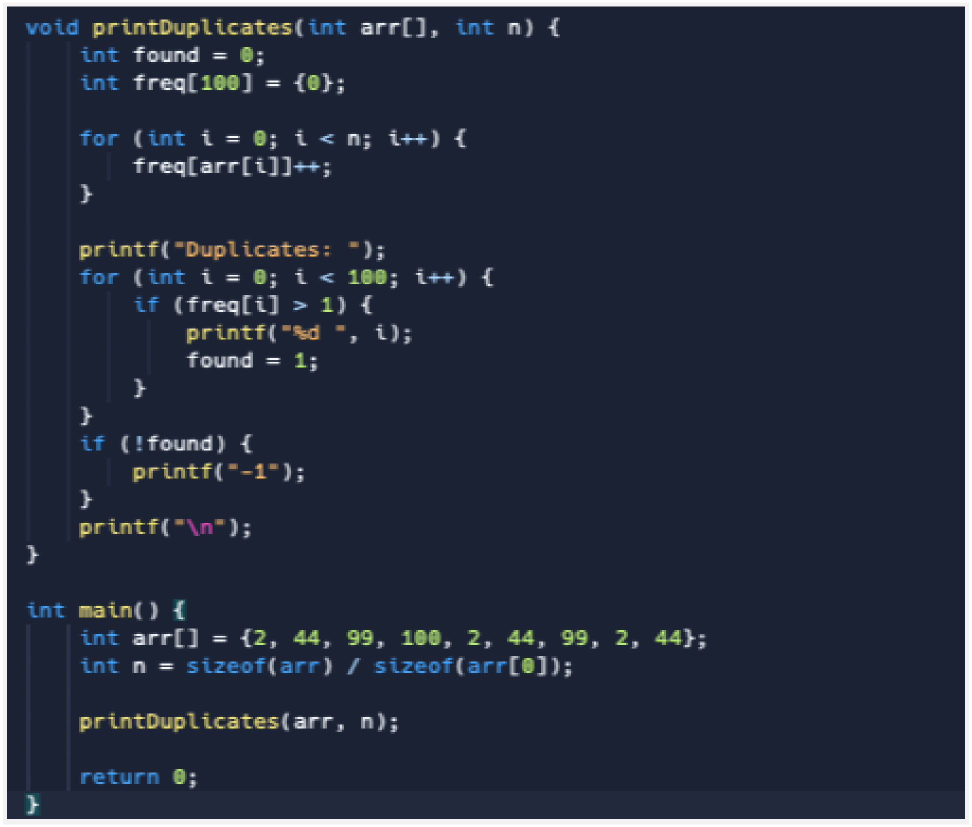
```c
void rotateArray(int arr[], int n) {
    int temp = arr[n - 1];
    for (int i = n - 1; i > 0; i--) {
        arr[i] = arr[i - 1];
    }
    arr[0] = temp;
}

void printArray(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

int main() {
    int arr[] = {3, 4, 5, 1, 2};
    int n = sizeof(arr) / sizeof(arr[0]);

    printf("Original Array: ");
    printArray(arr, n);

    rotateArray(arr, n);

    printf("After Rotation: ");
    printArray(arr, n);

    return 0;
}
```



```c
void rotateArray(int arr[], int n) {
    int temp = arr[n - 1];
    for (int i = n - 1; i > 0; i--) {
        arr[i] = arr[i - 1];
    }
    arr[0] = temp;
}

void printArray(int arr[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%d ", arr[i]);
    }
    printf("\n");
}

int main() {
    int arr[] = {3, 4, 5, 1, 2};
    int n = sizeof(arr) / sizeof(arr[0]);

    printf("Original Array: ");
    printArray(arr, n);

    rotateArray(arr, n);

    printf("After Rotation: ");
    printArray(arr, n);

    return 0;
}
```

14. 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.

```
void printDuplicates(int arr[], int n) {
    int found = 0;
    int freq[100] = {0};

    for (int i = 0; i < n; i++) {
        freq[arr[i]]++;
    }

    printf("Duplicates: ");
    for (int i = 0; i < 100; i++) {
        if (freq[i] > 1) {
            printf("%d ", i);
            found = 1;
        }
    }
    if (!found) {
        printf("-1");
    }
    printf("\n");
}

int main() {
    int arr[] = {2, 44, 99, 100, 2, 44, 99, 2, 44};
    int n = sizeof(arr) / sizeof(arr[0]);

    printDuplicates(arr, n);

    return 0;
}
```



```
void printDuplicates(int arr[], int n) {
    int found = 0;
    int freq[100] = {0};

    for (int i = 0; i < n; i++) {
        freq[arr[i]]++;
    }

    printf("Duplicates: ");
    for (int i = 0; i < 100; i++) {
        if (freq[i] > 1) {
            printf("%d ", i);
            found = 1;
        }
    }
    if (!found) {
        printf("-1");
    }
    printf("\n");
}

int main() {
    int arr[] = {2, 44, 99, 100, 2, 44, 99, 2, 44};
    int n = sizeof(arr) / sizeof(arr[0]);

    printDuplicates(arr, n);

    return 0;
}
```