Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Белгородский государственный технологический университет им. В.Г. Шухова»

(БГТУ им. В.Г. Шухова)

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №8

По дисциплине: «Основы программирования»

Тема: «Реализация функций для работы с одномерными массивами в стиле С»

Выполнил: студент группы ВТ-231

Борченко Александр Сергеевич

Проверили:

Черников Сергей Викторович

Новожен Никита Викторович

Белгород 2023

Цель работы: получение навыков написания функций при решении задач на одномерные массивы.

Задача 1: Ввод массива a размера n.

Код функции:

```c
#include <stdio.h>

// ввод массива a размера n
void inputArray(int * const a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

int main()
{
    int n = 3;
    int a[n];
    inputArray(a, n);

    return 0;
}
```

Задача 2: Вывод массива a размера n.

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) { //ввод массива a размером n
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void OutputArray(int *a, const size_t n) { //вывод массива a размером n
    for (size_t i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

int main()
{
    int n = 3;
    int a[n];
    InputArray(a, n);
    OutputArray(a, n);

    return 0;
}
```

Задача 3: Поиск позиции элемента со значением x с начала массива:

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) { //ввод массива a размером n
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

//поиск позиции элемента со значением x
unsigned getItemPosition(const int *a, const size_t n, const int x) {
    for (size_t i = 0; i < n; i++) {
        if (a[i] == x) {
            return i;
        }
    }

    return -1;
}

int main()
{
    int n = 6;
    int a[n];
    int x = 5;

    InputArray(a, n);
    printf("%d", getItemPosition(a, n, x));

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
1 5 7 6 10 3 X = 5	1

Задача 4: Поиск позиции первого отрицательного элемента.

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) { //ввод массива a размером n
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}
//получение первого элемента < 0
size_t getFirstNegativeIndex(const int *a, const size_t n) {
    for (size_t i = 0; i < n; i++)
        if (a[i] < 0)
            return i;
    return -1;
}

int main()
{
    int n = 6;
    int a[n];

    InputArray(a, n);

    printf("%d", getFirstNegativeIndex(a, n));

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
2 4 -6 8 10 12	2

Задача 5: **Поиск позиции элемента с начала массива (по функции-предикату).

Код функции:

```c
#include <stdio.h>
#include <math.h>

void InputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

int isEven(int x) {
    return x % 2 == 0;
}

int isNegative(int x) {
    return x < 0;
}

int isPrime(const int n) {
    if (n <= 1) {
        return 0;
    }

    int max_d = sqrt(n);

    for (int d = 2; d <= max_d; d++) {
        if (n % d == 0) {
            return 0;
        }
    }

    return 1;
}

size_t getFirstIndexPredicate(const int *a,const int n, int(*predicate)(int))
{
    for (int i = 0; i < n; i++) {
        if (predicate(a[i])) {
            return i;
        }
    }

    return -1;
}

//функцию писал как в лекции объяснялось, я чист душой, только с тетради
писал

int main() {
    int n = 3;
    int a[n];

    InputArray(a, n);

    printf("%d", getFirstIndexPredicate(a, n, isEven));

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
1 2 3 Predicate = isEven	1
-78 5 9 Predicate = isNegative	0
1 9 2 Predicate = isPrime	2

Задача 6: Поиск позиции последнего чётного элемента.

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) { //ввод массива a размером n
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}
//получение индекса последнего четного элемента
size_t getLastEvenIndex(const int *a, const size_t n) {
    for (size_t i = n - 1; i > 0; i--)
        if (a[i] % 2 == 0)
            return i;
    return -1;
}

int main() {
    int n = 6;
    int a[n];

    InputArray(a, n);

    printf("%d", getLastEvenIndex(a, n));

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
12 78 4 2 6 9	4

Задача 7: **Поиск позиции с конца массива (по функции-предикату).

Код функции:

```c
#include <stdio.h>
#include <math.h>

void InputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

int isEven(int x) {
    return x % 2 == 0;
}

int isNegative(int x) {
    return x < 0;
}

int isPrime(const int n) {
    if (n <= 1) {
        return 0;
    }

    int max_d = sqrt(n);

    for (int d = 2; d <= max_d; d++) {
        if (n % d == 0) {
            return 0;
        }
    }

    return 1;
}

size_t getLastIndexPredicate(const int *a, const int n, int
(*predicate)(int)) {
    for (size_t i = n - 1; i > 0; i--) {
        if (predicate(a[i])) {
            return i;
        }
    }

    return -1;
}

int main() {
    int n = 3;
    int a[n];

    InputArray(a, n);

    printf("%d", getLastIndexPredicate(a, n, isEven));

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
1 2 3 Predicate = isEven	1
-5 -2 -9 Predicate = isNegative	2
1 2 19 Predicate = isPrime	2

Задача 8: Вычисление количества отрицательных элементов.

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) { //ввод массива a размером n
    for (size_t i = 0; i < n; i++)
        scanf("%d", &a[i]);
}
//получение количества отрицательных элемента
size_t getAmmountNegativeNumber(const int *a, const size_t n) {
    int count_negative_num = 0;
    for (size_t i = 0; i < n; i++) {
        if (a[i] < 0) {
            count_negative_num++;
        }
    }

    return count_negative_num;
}

int main() {
    int n = 3;
    int a[n];

    InputArray(a, n);

    printf("%d", getAmmountNegativeNumber(a, n));

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
-5 9 -16	2

Задача 9: * Вычисление количества элементов массива, удовлетворяющих функции-предикату.

Код функции:

```c
#include <stdio.h>
#include <math.h>

void InputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

int isEven(int x) {
    return x % 2 == 0;
}

int isNegative(int x) {
    return x < 0;
}

int isPrime(const int n) {
    if (n <= 1) {
        return 0;
    }

    int max_d = sqrt(n);

    for (int d = 2; d <= max_d; d++) {
        if (n % d == 0) {
            return 0;
        }
    }

    return 1;
}
size_t getAmmountElementPredicate(const int *a, const int n, int
(*predicate)(int)) {
    int counter = 0;
    for (size_t i = 0; i < n; i++) {
        if (predicate(a[i])) {
            counter++;
        }
    }
    return counter;
}

int main() {
    int n = 6;
    int a[n];

    InputArray(a, n);

    printf("%d", getAmmountElementPredicate(a, n, isPrime));

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
2 4 7 13 9 15 Predicate = isEven	2
-8 10 12 -9 -6 -3 Predicate = isNegative	4
1 2 3 44 17 19 Predicate = isPrime	4

Задача 10: Изменение порядка элементов массива на обратный.

Код функции:

```c
#include <stdio.h>

void swap(int * const a, int * const b) {
    const int temp = *a;
    *a = *b;
    *b = temp;
}

void InputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void reverseArray(int * a, const size_t n) {
    for (size_t i = 0, g = n - 1; i < g; i++, g--) {
        swap(&a[i], &a[g]);
    }
}

void OutputArray(int *a, const size_t n) { //вывод массива a размером n
    for (size_t i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}
int main()
{
    int n = 3;
    int a[n];

    InputArray(a, n);
    reverseArray(a, n);
    OutputArray(a, n);

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
7 8 9	9 8 7

Задача 11: Проверка, является ли последовательность палиндромом.

Код функции:

```
#include <stdio.h>
#include <stdbool.h>
#include <windows.h>

void InputArray(int *a, const int n) {
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}
//на лекции и на практике было, все записано :)
bool isPalindrome(const int * a, const size_t n) {
    for (size_t i = 0, g = n - 1; i < g; i++, g--) {
        if (a[i] != a[g]) {
            return false;
        }
    }
    return true;
}

int main()
{
    SetConsoleOutputCP(CP_UTF8);

    int n = 3;
    int a[n];

    InputArray(a, n);

    if (isPalindrome(a, n)) {
        printf("Последовательность является палиндромом");
    } else {
        printf("Последовательность не является палиндромом");
    }

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
1 2 1	Последовательность является палиндромом
1 2 3	Последовательность не является палиндромом

Задача 12: Сортировка массива выбором.

Код функции:

```c
#include <stdio.h>

void swap(int * const a, int * const b) { //обмен значениями
    const int temp = *a;
    *a = *b;
    *b = temp;
}

void InputArray(int *a, const int n) {//ввод массива a размером n
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void OutputArray(int *a, const size_t n) { //вывод массива a размером n
    for (size_t i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

void selectionSort(int *a, const int size) {
    for (int i = 0; i < size - 1; i++) {
        int minPos = i;
        for (int j = i + 1; j < size; j++)
            if (a[j] < a[minPos])
                minPos = j;
        swap(&a[i], &a[minPos]);
    }
}

int main() {
    int n = 6;
    int a[n];

    InputArray(a, n);
    selectionSort(a, n);
    OutputArray(a, n);

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
5 6 1 2 4 3	1 2 3 4 5 6

Задача 13: Удаление из массива всех нечетных элементов.

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) {//ввод массива a размером n
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}
//удаление нечетных элементов
void deleteOddElements(int * a, int * n) {
    int g = 0;
    for (int i = 0; i < *n; i++) {
        if (a[i] % 2 == 0) {
            a[g] = a[i];//в новый массив заносим четный элемент
            g++;
        }
    }
    *n = g;
}

void OutputArray(int *a, const int n) { //вывод массива a размером n
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

int main() {
    int n = 6;
    int a[n];

    InputArray(a, n);
    deleteOddElements(a, &n);
    OutputArray(a, n);

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
7 8 2 4 19 17	8 2 4

Задача 14: Вставка элемента в массив с сохранением относительного порядка других элементов.

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) {//ввод массива a размером n
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void InsertingValueIntoPosition(int *a, int *n, const int position, const int
value) {
    for (int i = *n - 1; i >= position; i--)
        a[i + 1] = a[i];
    a[position] = value;
    (*n)++;
}

void OutputArray(int *a, const int n) { //вывод массива a размером n
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

int main() {
    int n = 6;
    int a[n];
    int position = 4;
    int value = 144;

    InputArray(a, n);
    InsertingValueIntoPosition(a, &n, position, value);
    OutputArray(a, n);

    return 0;

}
```

Тестовые данные:

Входные данные	Выходные данные
1 2 3 4 5 6 Position = 4 Value = 144	1 2 3 4 144 5 6

Задача 15: Добавление элемента в конец массива.

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) {//ввод массива a размером n
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void append(int *a, int *n, const int value) {
    a[*n] = value;
    (*n)++;
}

void OutputArray(int *a, const int n) { //вывод массива a размером n
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

int main() {
    int n = 6;
    int a[n];
    int value = 12;

    InputArray(a, n);
    append(a, &n, value);
    OutputArray(a, n);

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
1 2 3 4 5 6 Value = 12	1 2 3 4 5 6 12

Задача 16: Удаление элемента с сохранением относительного порядка других элементов.

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) {//ввод массива a размером n
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void DeletingElementWhilePreservingRelativeorderOtherElements(int *a, int *n,
const size_t position) {
    for (size_t i = position; i < *n - 1; i++)
        a[i] = a[i + 1];
    (*n)--;
}

void OutputArray(int *a, const int n) { //вывод массива a размером n
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

int main()
{
    int n = 6;
    int a[n];
    int position = 3;

    InputArray(a, n);
    DeletingElementWhilePreservingRelativeorderOtherElements(a,&n,position);
    OutputArray(a, n);

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
9 8 7 6 5 4 Position = 3	9 8 7 5 4

Задача 17: Удаление элемента без сохранения относительного порядка других элементов.

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) {//ввод массива a размером n
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void DeletingElementWithoutPreservingRelativeorderOtherElements(int *a, int
*n, size_t pos) {
    a[pos] = a[*n - 1];
    (*n)--;
}

void OutputArray(int *a, const int n) { //вывод массива a размером n
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

int main()
{
    int n = 6;
    int a[n];
    int position = 3;

    InputArray(a, n);
   DeletingElementWithoutPreservingRelativeorderOtherElements(a,&n,position);
    OutputArray(a, n);

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
4 5 6 9 8 7 Position = 3	4 5 6 7 8

Задача 18: ** Реализуйте циклический сдвиг массива влево на k позиций.

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) {//ввод массива a размером n
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void CyclicShiftByKPositions(int *a, size_t n, int k) {
    int ammount_revolutions = k % n;
//пример с методички: k = 212, n = 5, тогда:
// 212 % 5 = 2 (42 круга пройдет, но сдвинет на 2 как остаток
    for(size_t i = 0; i < ammount_revolutions; i++) {
        int first_number_undone_array = a[0];
//первое число должно отправится в конец
        for(size_t g = 0; g < n - 1; g++) {
            a[g] = a[g + 1];
        }
        a[n - 1] = first_number_undone_array;
    }
}

void OutputArray(int *a, const int n) { //вывод массива a размером n
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

int main()
{
    int n = 5;
    int a[n];
    int k = 212;

    InputArray(a, n);
    CyclicShiftByKPositions(a, n, k);
    OutputArray(a, n);

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
1 2 3 4 5 K = 3	3 4 5 1 2

Задача 19: ** Реализуйте функцию $forEach$, которая применяет функцию f к элементам массива a размера $size$.

Код функции:

```c
#include <stdio.h>

void InputArray(int *a, const int n) {//ввод массива a размером n
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

int abs(int x) {
    if (x < 0) {
        return -x;
    } else {
        return x;
    }
}

int ExponentiationNumber(int b) {
    return b * b;
}

void forEach(int *a, size_t n, int (*predicate) (int)) {
    for(size_t i = 0; i < n; i++) {
        a[i] = predicate(a[i]);
    }
}

void OutputArray(int *a, const int n) { //вывод массива a размером n
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

int main()
{
    int n = 5;
    int a[n];

    InputArray(a, n);
    forEach(a, n,ExponentiationNumber);
    OutputArray(a, n);

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
-1 -2 3 -4 5 Predicate = abs	1 2 3 4 5
1 -2 3 -4 5 Predicate = ExponentiationNumber	1 4 9 16 25

Задача 20: ** Реализуйте функцию any, которая возвращает значение ’истина’, если хотя бы один элемент массива a размера $size$ удовлетворяют функции-предикату f, иначе – ’ложь’.

Код функции:

```c
#include <stdio.h>
#include <stdbool.h>

void InputArray(int *a, const int n) {//ввод массива a размером n
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

int isPositive(int a) {
    return a > 0;
}

bool any(int *a, size_t n, int (*predicate) (int)) {
    for(size_t i = 0; i < n; i++) {
        if (predicate(a[i])) {
            return 1;
        }
    }

    return 0;
}

int main()
{
    int n = 5;
    int a[n];

    InputArray(a, n);
    printf("%d", any(a, n, isPositive));

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
-1 -2 3 -4 -5 Predicate = isPositive	1
-1 -2 -3 -4 -5 Predicate = isPositive	0

Задача 21: ** Реализуйте функцию all, которая возвращает значение ’истина’, если все элементы массива a размера $size$ удовлетворяют функции-предикату f, иначе – ’ложь’.

Код функции:

```c
#include <stdio.h>
#include <stdbool.h>

void InputArray(int *a, const int n) {//ввод массива a размером n
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

int isPositive(int a) {
    return a > 0;
}

bool all(int *a, size_t n, int (*predicate) (int)) {
    int is_positive = true;
    for (size_t i = 0; i < n; i++) {
        if(!predicate(a[i])) {
            is_positive = false;
        }
    }

    return is_positive;
}

int main()
{
    int n = 5;
    int a[n];

    InputArray(a, n);
    printf("%d", all(a, n, isPositive));

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
1 -2 3 4 5 Predicate = isPositive	0
1 2 3 4 5 Predicate = isPositive	1

Задача 22: **Реализуйте функцию $arraySplit$, которая разделяет элементы массива a размера $size$ на элементы, удовлетворяющие функции-предикату f, сохраняя в массиве b, иначе – в массиве c.

Код функции:

```c
#include <stdio.h>
#include <math.h>

int isPositive(int a) {
    return a > 0;
}

int isPrime(const int n) {
    if (n <= 1) {
        return 0;
    }

    int max_d = sqrt(n);

    for (int d = 2; d <= max_d; d++) {
        if (n % d == 0) {
            return 0;
        }
    }

    return 1;
}
void InputArray(int *a, const int n) {//ввод массива a размером n
    for (int i = 0; i < n; i++)
        scanf("%d", &a[i]);
}

void OutputArray(int *a, const int n) { //вывод массива a размером n
    for (int i = 0; i < n; i++)
        printf("%d ", a[i]);
    printf("\n");
}

void arraySplit(int *a, size_t n, int (*predicate) (int)) {
    int b[n], c[n];
    int bn = 0, cn = 0;
    for(size_t i = 0; i < n; i++) {
        if(predicate(a[i])) {
            b[bn] = a[i];
            bn++;
        } else {
            c[cn] = a[i];
            cn++;
        }
    }
    OutputArray(b, bn);
    OutputArray(c, cn);
}

int main()
{
    int n = 5;
    int a[n];

    InputArray(a, n);
    arraySplit(a, n, isPrime);

    return 0;
}
```

Тестовые данные:

Входные данные	Выходные данные
-1 2 -3 -4 5 Predicate = isPositive	B[n] = 2 5 C[n] = -1 -3 -4
1 2 3 4 5 Predicate = isPrime	B[n] = 2 3 5 C[n] = 1 4

Вывод: в ходе выполнения лабораторной работы я смог реализовать функции для работы с одномерными массивами в стиле C, закрепил навыки написания отдельных функций.