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 №11

По дисциплине: «Основы программирования»

Тема: «Рекурсивные функции»

Выполнил: студент группы ВТ-231

Борченко Александр Сергеевич

Проверили:

Черников Сергей Викторович

Новожен Никита Викторович

Белгород 2023

Цель работы: получение навыков написания рекурсивных функций.

Содержание работы:

Задача 1: Определить количество цифр в тексте, вводимом с клавиатуры. Текст заканчивается символом перехода на новую строку “\n”.

Код задачи:

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

int getCountDigits() {
    char c = getchar();

    if (c == '\n') {
        return 0;
    } else {
        return isdigit(c) + getCountDigits();
    }
}

int main() {
    int count = getCountDigits();

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

    return 0;
}
```

Задача 2: Вывести данное натуральное число в восьмеричной системе счисления.

Код задачи:

```c
#include <stdio.h>

void getOctal(int x) {
    if (x) {
        getOctal(x / 8);
        printf("%d", x % 8);
    }
}

int main() {
    int x;
    scanf("%d", &x);

    getOctal(x);

    return 0;
}
```

Альтернативный код задачи (через побитовые операции):

```c
#include <stdio.h>

void getOctal(int x) {
    if (x == 0)
        return;
    else {
        int digit = x & 7;
        getOctal(x >> 3);
        printf("%d", digit);
    }
}

int main() {
    int x;
    scanf("%d", &x);

    getOctal(x);

    return 0;
}
```

Задача 3: Дан знаменатель и первый член геометрической прогрессии. Вычислить n-й член прогрессии.

Код задачи:

```c
#include <stdio.h>

//q - это знаменатель, а b1 - первый член прогрессии

int getNMemberGeomProg(int b1, int q, int n) {
    if (n > 2) {
        return getNMemberGeomProg(b1 * q, q, --n);
    } else {
        return b1 * q;
    }
}

int main()
{
    int b1, q, n;
    scanf("%d %d %d", &b1, &q, &n);

    printf("%d", getNMemberGeomProg(b1, q, n));
//пример: b1= 2, q = -2, n(5)-?
//Тогда: 2 * (-2)^5-1 = 2 * (-2)^4 = 32
    return 0;
}
```

Задача 4: Дана упорядоченная по убыванию последовательность целых чисел. Определить, есть ли среди членов данной последовательности число x, и если есть, найти номер этого члена.

Код задачи:

```c
#include <stdio.h>

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

int getIndexDigitX(int *a, int n, int x) {
    if (a[n - 1] != x && n) {
        return getIndexDigitX(a, --n, x);
    } else {
        return n - 1;
    }
}

int main()
{
    int n, x;
    scanf("%d %d", &n, &x);

    int a[n];
    inputArray(a, n);

    printf("%d", getIndexDigitX(a, n, x));

    return 0;
}
```

Задача 5: Дан массив a размера n ($n \geq 2$). Необходимо проверить, является ли он упорядоченным по неубыванию.

Код задачи:

```c
#include <stdio.h>

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

int isOrderedNonDecreasing(int *a, int n) {
    if (a[n - 2] < a[n - 1] && n >= 3) {
        return isOrderedNonDecreasing(a, --n);
    } else {
        return a[n - 2] < a[n - 1];
    }
}

int main()
{
    int n;
    scanf("%d", &n);

    int a[n];
    inputArray(a, n);

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

    return 0;
}
```

Задача 6: Найти номер первого вхождения минимального значения в последовательность длины n (линейный поиск).

Код задачи:

```c
#include <stdio.h>

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

int SearchFirstMin(int *a, int n, int temp_index, int min_index) {
    if (temp_index == n) {
        return min_index;
    }
    if (a[temp_index] < a[min_index]) {
        min_index = temp_index;
    }
    return SearchFirstMin(a, n, temp_index + 1, min_index);
}

int getFirstMinIndexNum(int *a, int n) {
    return SearchFirstMin(a, n, 1, 0);
}

int main() {
    int n;
    scanf("%d", &n);

    int a[n];
    inputArray(a, n);

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

    return 0;
}
```

Задача 7: Даны натуральные числа a и b. Определить, могут ли эти числа быть соседними членами последовательности Фибоначчи.

Код задачи:

```c
#include <stdio.h>

int isFibonacciNum(int sum_a_b, int a, int b) {
    if (b == sum_a_b) {
        return 1;
    } else if (b > sum_a_b) {
        return 0;
    } else {
        return isFibonacciNum(sum_a_b, b, a + b);
    }
}
int isNeighboringNum(int a, int b) {
    if (a < 1 || b < 1)
        return 0;
    int sum_a_b = a + b;
    return isFibonacciNum(sum_a_b, 1, 1);
}

int main()
{
    //пример a = 21 , b = 34
    int a, b;
    scanf("%d %d", &a, &b);
    //34 = 21 + 13 т.е является суммой предыдущих чисел => числа Фибоначчи
    printf("%d", isNeighboringNum(a, b));

    return 0;
}
```

Задача 8: Вывести в обратном порядке символы данного текста, вводимого с клавиатуры, которые не являются цифрами. Текст заканчивается символом перехода на новую строку \n.

Код задачи:

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

void ReversText() {
    char c = getchar();

    if (c != '\n') {
        ReversText();
    }
    if (isdigit(c) == 0) {
        printf("%c", c);
    }
}
//Пример: sasha228 -> ahsas (не трогает цифры)
int main()
{
    ReversText();

    return 0;
}
```

Задача 9: Дан n-й член арифметической прогрессии, ее разность и значение n. Вычислить первый член прогрессии.

Код задачи:

```c
#include <stdio.h>

int getFirstMemberArifProgression(int a_n, int d) {
    if (a_n > d) {
        return getFirstMemberArifProgression(a_n - d, d);
    } else {
        return a_n;
    }
}

int main()
{
    //пример: a_n = 11, b = 2
    int a_n, d;
    scanf("%d %d", &a_n, &d);

    printf("%d", getFirstMemberArifProgression(a_n, d));
    //Тогда a(1) = 1, т.к 11-2-2... = 1
    return 0;
}
```

Задача 10: С клавиатуры вводятся положительные вещественные числа $a1$, $a2$, . . . , an. Признак конца ввода – отрицательное число.

Код задачи:

```c
#include <stdio.h>

void getFormulaValue(float a1, float a2) {

    float answer = (a1 + a2) / 2;
    printf("%f ", answer);

    if (a2 >= 0) {
        float b;//новое значение a(2)
        scanf("%f", &b);

        getFormulaValue(a2, b);
    }
}

int main()
{
    float a1, a2;
    scanf("%f %f", &a1, &a2);

    getFormulaValue(a1, a2);

    return 0;
}
```

Задача 11: Реализовать функции any, которая возвращает значение ’истина’, если хотя бы один элемент удовлетворяет функции-предикату f, в противном случае – ложь. Реализовать функцию all, которая возвращает значение ’истина’, если все элементы удовлетворяет функции-предикату f, в противном случае – ложь.

Код задачи:

```
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;
}

int any(int *a, int n, int (*predicate)(int)) {
    if (n == 0) {
        return 0;
    } else {
        return predicate(a[0]) || any(a + 1, n - 1, predicate);
    }
}

int all(int *a, int n, int (*predicate)(int)) {
    if (n == 0) {
        return 1;
    } else {
        return predicate(a[0]) && all(a + 1, n - 1, predicate);
    }
}
```

Задача 12: Реализовать алгоритм бинарного поиска.

Код задачи:

```c
#include <stdio.h>
#include <windows.h>
//Начало функции
int binarySearch(int a[], int left, int right, int x)
{
    if (right >= left) {
        int middle = left + (right - left) / 2;
        if (a[middle] == x)
            return middle;
        if (a[middle] > x) {
            return binarySearch(a, left, middle - 1, x);
        }
        return binarySearch(a, middle + 1, right, x);
    }

    return -1;
}
//Конец функции
//Проверка работоспособности
int main()
{
    SetConsoleOutputCP(CP_UTF8);

    int a[] = { 5, 254, 567, 143, 4098 };
    int size = sizeof(a) / sizeof(a[0]);

    int x = 567;

    int index = binarySearch(a, 0, size - 1, x);

    if (index == -1) {
        printf("Элемента нет в массиве");
    }
    else {
        printf("Индекс элемента %d", index);//индекс элемента 2
    }

    return 0;
}
```

Задача 13: Реализовать сортировку выбором.

Код задачи:

```c
#include <stdio.h>

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

int SearchFirstMin(int *a, int n, int temp_index, int min_index) {
    if (temp_index == n) {
        return min_index;
    }
    if (a[temp_index] < a[min_index]) {
        min_index = temp_index;
    }
    return SearchFirstMin(a, n, temp_index + 1, min_index);
}
//Начало основной функции
void selectionSort(int a[], int n) {
    if (n <= 1) {
        return;
    }
    int min_index = SearchFirstMin(a, n, 1, 0);
    swap(&a[0], &a[min_index]);

    selectionSort(a+1, n-1);
}
//Конец основной функции
void OutputArray(int a[], int n) {
    for (int i = 0; i < n; i++) {
        printf("%d ", a[i]);
    }
}
//проверка работоспособности
int main()
{
    int a[] = { 8, 27, 18, 2, 1, -9, -12 };
    int n = sizeof(a) / sizeof(a[0]);

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

    return 0;
}
```

Вывод: в ходе выполнения лабораторной работы я получил навыки написания рекурсивных функций.