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: «Структуры. Функции для работы со структурами»

Выполнил: студент группы ПВ-223
Мелехов Артём Дмитриевич

Проверили:
ст. преп. Притчин Иван Сергеевич
асс. Черников Сергей Викторович

Белгород 2022 г.

Лабораторная работа №11 «Структуры. Функции для работы со структурами»

Цель работы: получение навыков написания функций для решения задач со структурами.

Содержание отчета:

Тема лабораторной работы

Цель лабораторной работы

Функции для варианта 3. Для каждой функции указаны:

– Условие задачи.

– Спецификация.

– Исходный код функции.

Вывод.

Исходная структура:

```
typedef struct Fraction
{
	int numerator;		// числитель
	int denumerator;	// знаменатель
} Fraction;
```

Задача a.

Условие:

Реализуйте функцию inputFraction ввода структуры Fraction. Пример ввода, который должен обрабатываться программой: ‘5/7’, ‘2/17’.
Заголовок: void input_fraction(Fraction *f)

Код функции:

```
#include <assert.h>

// (a)
// Функция позволяет ввести дробь структуры Fraction
void input_fraction(Fraction *f)
{
	scanf("%d/%d", &f->numerator, &f->denumerator);

	assert(f->denumerator);
}
```

Задача b.

Условие:

Реализуйте функцию inputFractions ввода массива структур Fraction.
Заголовок: void input_fractions(Fraction* f, size_t n)

Код функции:

```
// (b)
// Функция позволяет ввести массив a структур
// Fraction размера n
void input_fractions(Fraction* f, size_t n)
{
	for (size_t i = 0; i < n; i++)
		input_fraction(&f[i]);
}
```

Задача c.

Условие:

Реализуйте функцию outputFraction вывода структуры Fraction в формате ‘5/7’.
Заголовок: void output_fraction(Fraction f)

Код функции:

```
// (c)
// Функция позволяет вывести дробь структуры Fraction
void output_fraction(Fraction f)
{
	printf("%d/%d\n", f.numerator, f.denumerator);
}
```

Задача d.

Условие:

Реализуйте функцию outputFractions вывода массива структур Fraction.
Заголовок: void output_fractions(Fraction *f, size_t n)

Код функции:

```
// (d)
// Функция возвращает массив a структур
// Fraction размера n
void output_fractions(Fraction *f, size_t n)
{
	for (size_t i = 0; i < n; i++)
		output_fraction(f[i]);
}
```

Задача e.

Условие:

Реализуйте функцию gcd возвращающую наибольший общий делитель.
Заголовок: int gcd(int a, int b)

Код функции:

```
// (e)
// Возвращает НОД чисел a и b
int gcd(int a, int b)
{
	while (a && b)
		if (a > b)
			a %= b;
		else
			b %= a;

	return a + b;
}
```

Задача f.

Условие:

Реализуйте функцию lcm возвращающую наименьшее общее кратное.
Заголовок: int lcm(int a, int b)

Код функции:

```
#include <assert.h>

// (f)
// Возвращает НОК чисел a и b
int lcm(int a, int b)
{
	assert(gcd(a, b));

	return a / gcd(a, b) * b;
}
```

Задача g.

Условие:

Реализуйте функцию simpleFraction для сокращения дроби a.
Заголовок: void simple_fraction(Fraction *f)

Код функции:
```c
#include <assert.h>

// (g)
// Сокращает дробь структуры Fraction
void simple_fraction(Fraction *f)
{
	int gcd_fraction = gcd(f->numerator, f->denumerator);

	assert(gcd_fraction);
	f->numerator /= gcd_fraction;
	f->denumerator /= gcd_fraction;
}
```

Задача h.

Условие:

Реализуйте функцию mulFractions умножения двух дробей a и b.
Заголовок: mul_fraction(Fraction f1, Fraction f2)

Код функции:
```c
// (h)
// Производит умножение двух дробей структуры
// Fraction
Fraction mul_fraction(Fraction f1, Fraction f2)
{
	Fraction f = (Fraction)
	{
		f1.numerator * f2.numerator,
		f1.denumerator * f2.denumerator
	};
	simple_fraction(&f);

	return f;
}
```

Задача i.

Условие:

Реализуйте функцию divFractions деления двух дробей a и b.
Заголовок: Fraction div_fraction(Fraction f1, Fraction f2)

Код функции:
```c
#include <assert.h>

// (i)
// Производит деление двух дробей структуры
// Fraction
Fraction div_fraction(Fraction f1, Fraction f2)
{
	assert(f2.numerator);
	swap(&f2.numerator, &f2.denumerator, sizeof(f2.numerator));

	return mul_fraction(f1, f2);
}
```

Задача j.

Условие:

Реализуйте функцию addFractions сложения двух дробей a и b.
Заголовок: add_fraction(Fraction f1, Fraction f2)

Код функции:

```
// (j)
// Производит сложение двух дробей структуры
// Fraction
Fraction add_fraction(Fraction f1, Fraction f2)
{
	Fraction f = (Fraction)
	{
		f1.numerator * f2.denumerator +
		 f2.numerator * f1.denumerator,

		f1.denumerator * f2.denumerator
	};
	simple_fraction(&f);

	return f;
}
```

Задача k.

Условие:

Реализуйте функцию subFractions вычитания двух дробей a и b.
Заголовок: Fraction sub_fraction(Fraction f1, Fraction f2)

Код функции:

```
// (k)
// Производит вычитание двух дробей структуры
// Fraction
Fraction sub_fraction(Fraction f1, Fraction f2)
{
	f2.numerator = -f2.numerator;

	return add_fraction(f1, f2);
}
```

Задача l.

Условие:

Реализуйте функцию для поиска суммы n дробей.
Заголовок: Fraction sum_fractions(Fraction* f, size_t n)

Код функции:

```
// (l)
// Производит сложение массива дробей структуры
// Fraction f размера n
Fraction sum_fractions(Fraction* f, size_t n)
{
	Fraction sum = f[0];

	for (size_t i = 1; i < n; i++)
		sum = add_fraction(sum, f[i]);

	return sum;
}
```

Вывод: в ходе выполнения работы были получены навыки написания функций для решения задач со структурами.