Лабораторная работа №5
Программирование с использованием функций.

Вся лабораторная работа обязана являться единым проектом разделённым на решения, где решение – это отдельный проект задания из лабораторной. В заданиях без звёздочки необходимо предусмотреть меню. Пункты меню: Условие задания, запуск функции самого задания, информация о студенте выполнившим задание. Предусмотреть возможность повторного запуска функции решения без перезапуска программы. Создать функции проверки на ввод, при этом использовать функции стандартных библиотек запрещено. Для заданий 3 - 5 подготовить минимум 3 разнотипных Unit теста для метода, который решает основную задачу.

Для задания 1 создать статическую библиотеку.
Для задания 2 создать динамическую библиотеку.
Библиотеки подключать в главный исполняемый файл. В библиотеках должна быть реализована функция, которая решает задание. При ознакомлении с принципами работы и подключения библиотек используйте рекомендованный методический материал. (В материале используется IDE Builder C++).

Выполнив только задания 1 - 5 студент может получить 4 балла при полностью отвеченной теории по лабораторной работе.

Задание 1. (Номер задания ваш номер по списку).
1. Составить программу для решения уравнения $a_i x + b_j = 0$, , где a_i и b_j – элементы динамических массивов, $i = 0, 1, \dots, 7$, $j = 0, 1, \dots, 7$. Массивы $A = a_0, a_1, \dots, a_7$ и $B = b_0, b_1, \dots, b_7$ ввести с клавиатуры. При $a_i \neq 0$ вывести на экран результат, а при $a_i = 0$ переменной *x* присвоить значение 0, которое также вывести на экран. Использовать функции, размерность массивов ввести с клавиатуры, исходные данные ввести с клавиатуры.

2. Сформировать два двумерных динамических массива-матрицы *A* и *B* размерностью $n \times n$. Значения элементов массивов a_{ij} и b_{ij} определить согласно выражениям $a_{ij} = 3\,i\,j - 3$; $b_{ij} = 2\,i\,j - 2$ при $i = 0, 1, \dots, \mathrm{n}$, $j = 0, 1, \dots, n$. Размерность массивов ввести с клавиатуры. Определить суммы элементов главных диагоналей данных массивов-матриц. Использовать функции. Вывести на экран полученные массивы в виде матриц и значения сумм.

3. Сформировать два двумерных динамических массива-матрицы *A* и *B* размерностью $n \times n$. Размерность массивов ввести с клавиатуры. Значения элементов a_{ij} и b_{ij} определить согласно выражениям:
$a_{ij} = \begin{cases} 3\,i\,j - 3, \text{если} & i \leq 5 \\ 2\,i\,j - 2, \text{если} & i > 5 \end{cases}$ $b_{ij} = \begin{cases} 4\,i\,j - 5, \text{если} & i > 7 \\ 5\,i\,j - 4, \text{если} & i \leq 7 \end{cases}$ при $i = 0,1, \dots, n$ $j = 0, 1, \dots, n$. Определить суммы элементов, расположенных по периметру, для данных массивов-матриц. Использовать функции. Вывести на экран полученные массивы в виде матриц и значения сумм.

4. Составить программу для вычисления математического ожидания *m* и дисперсии *D* по формулам:

$$m = \frac{1}{n}\sum_{i=1}^{n} a_i.$$

$$D = \frac{1}{n}\sum_{i=1}^{n}(a_i - m)^2.$$

Причем *n* чисел $a_1, a_2, \dots, a_n$ – элементы динамического массива, вычислить по формуле:

$$a_i = \begin{cases} \sin i, \text{если } i > 17; \\ \operatorname{ctg} i^2, \text{если } i \le 17, i = 1,2, \dots, n. \end{cases}$$

Размерность массива ввести с клавиатуры. Элементы массива, значения математического ожидания и дисперсии вывести на экран. Использовать функции.

5. Составить программу для нахождения наибольшего элемента двумерного динамического массива-матрицы *Z*. Каждый элемент массива-матрицы *Z* вычислить по формуле $Z = X_i Y_j$, где $i = 0,1, \dots , n; \quad j = 0, 1, \dots, m$ Одномерные динамическиемассивы $X = x_0, x_1, \dots, x_\text{n}$ и $Y = y_0, y_1, \dots, y_\text{m}$ ввести с клавиатуры. Использовать функции.

6. Составить программу для вычисления произведения одномерных динамических массивов (векторов) $X = x_1, x_2, \dots, x_\text{n}$ и $Y = y_1, y_2, \dots, y_\text{n}$. Элементы вектора $X = x_1, x_2, \dots, x_\text{n}$ ввести с клавиатуры, а элементы вектора *Y* вычислить по формуле $y_i = 0{,}1\operatorname{tg}(0{,}1i)$, где $i = 0,1, \dots , n, \ n = 10$. Вывести на экран значение *произведения* и элементы вычисленного вектора *Y*. Использовать функции.

7. Сформировать два двумерных динамических массива-матрицы *C* и *D* размерностью $k \times k$. Размерность массивов ввести с клавиатуры. Значения элементов c_{ij} и d_{ij} определить согласно выражениям:

$$c_{ij} = \frac{i^2 + j^2}{i + j + 1}; \ d_{ij} = \begin{cases} i^2 + j^2, \text{если } j \le 8; \\ \frac{i^2 + j^2}{2}, \text{если } j > 8 \end{cases} \quad \text{при } i = 0,1, \dots , k \quad j = 0, 1, \dots , k.$$

Определить сумму элементов, расположенных на главной и побочной диагоналях, для каждой матрицы. На экран вывести полученные массивы и значения сумм. Использовать функции.

8. Составить программу для нахождения наименьшего элемента одномерного динамического массива $S = (S_1, S_2, \dots, S_m)$, где каждый элемент S_j вычислить по формуле: $S_j = \sum_{i=1}^{n} a_{ij}, \quad j = 1, 2, \dots, m$; Использовать функции. Размерность массивов ввести с клавиатуры. Значения элементов a_{ij} двумерного динамического массива-матрицы *A*, где $i = 1,2, \dots , n; \ j = 1, 2, \dots , m.$ ввести с клавиатуры: Значения элементов одномерного динамического массива *S* и его наименьшего элемента вывести на экран. Использовать функции.

9. Сформировать двумерный динамический массив-матрицу *A* размерностью $n \times n$, причем значения a_{ij} определить согласно выражению:

$$a_{ij} = \begin{cases} 2\ ij^2 - 2j, \text{если } i \le 5 \\ 3\ ij - \ 3, \text{если } i > 5 \end{cases} \quad \text{при } i = 1, 2, \dots , n \quad j = 1, 2, \dots , n.$$

Сформировать транспонированную матрицу $B = A^T$.
Определить сумму элементов четных строк и нечетных столбцов для массива-матрицы *A* и сумму четных столбцов и нечетных строк для массива-матрицы *B*. На

экран вывести массивы-матрицы *A* и *B* построчно и значения сумм. Использовать функции.

10. Составить программу вычисления значений элементов одномерного динамического массива (вектора) $X = x_0, x_1, \dots, x_n$ по формуле: $x_i = e^{-0.5 \cdot i - 2 \cdot \pi} - arctg(i + 0.1)$, где $i = 0,1,2, \dots, n$. Размерность массива ввести с клавиатуры. Вычисленные элементы массива вывести на экран. Преобразовать полученный массив по следующему правилу: все отрицательные элементы увеличить на 0.5, а все положительные заменить на 0.1. Преобразованный массив также вывести на экран. Использовать функции.

11. Составить программу для вычисления значений элементов одномерного динамического массива (вектора) $Z = z_1, z_2, \dots, z_n$ по формуле $z_k = x_k + my_k$, где x_k и y_k – компоненты одномерных динамических массивов $X = x_1, x_2, \dots, x_n$ и $Y = y_1, y_2, \dots, y_n$. Размерность массивов *n* ввести с клавиатуры; величину *m* вычислить по формулам:

$$m = \begin{cases} k, \text{если } |\sin k| \le 0{,}2; \\ \sqrt{k}, \text{если } 0{,}2 \le |\sin k| < 0{,}9; \\ \sqrt{\sqrt{k}}, \text{если } |\sin k| \ge 0{,}9, \end{cases} \text{ где } k = 0, 1, \dots, n.$$

Значения элементов массивов *X* и *Y* ввести с клавиатуры. Вывести на экран значения элементов массивов *X*, *Y* и *Z*. Использовать функции.

12. Сформировать два двумерных динамических массива-матрицы *A* и *B* размерностью $n \times k$. Размерность массивов ввести с клавиатуры. Значения элементов a_{ij} определить согласно выражениям:

$$a_{ij} = \begin{cases} 2\,i\,j^2 - 2j, \text{если } i \le 3; \\ 2\,i\,j\ -\ 2, \text{если } i > 3; \\ 2\,i\,j - \ 2, \text{если } j \le 9; \\ 3\,i\,j^2 - \ 3j, \text{если } j > 9, \end{cases} \quad \text{где } i = 0, 1, \dots, n \quad j = 0, 1, \dots, k.$$

Значения элементов b_{ij} определить путем возведения в квадрат соответствующих элементов массива-матрицы *A*. Определить сумму элементов четных столбцов для каждого массива-матрицы. На экран вывести массивы-матрицы *A*, *B* и значения сумм. Использовать функции.

13. Составить программу вычисления элементов двумерного динамического массива-матрицы *A* размерностью $n \times n.$ Значения элементов a_{ij}определить согласно выражениям:

$$a_{ij} = \begin{cases} b_{ij}, \text{если } |b_{ij}| > |c_{ij}|; \\ c_{ij}, \text{если } |b_{ij}| \le |c_{ij}|, \end{cases} \text{ где } i = 0, 1, \dots, n, j = 0, 1, \dots, n.$$

Размерность массивов (*n* = 3) ввести с клавиатуры. Значения элементов двумерных массивов-матриц *B* и *C* ввести с клавиатуры. Результат – значения массива *A* вывести на экран. Использовать функции.

14. Составить программу определения координат седловой точки двумерного динамического массива *B* размерностью $m \times n.$ Размерность массива ввести с клавиатуры. Значения элементов массива *B* ввести с клавиатуры. *Примечание*. Элемент массива называется седловой точкой, если он является одновременно наименьшим в своей строке и наибольшим в своем столбце. Определение

координат седловой точки (номеров строки и столбца) оформить в виде функции для любых *m* и *n*. На экран вывести в виде матриц исходный массив, а также массив, размерностью *m* x *n*, в котором все элементы, кроме седловой точки, равны 0. Если седловая точка не обнаружена, вывести на экран текст СЕДЛОВОЙ ТОЧКИ НЕТ. Использовать функции.

15. Составить программу для определения значений элементов двумерного динамического массива *C*, являющегося суммой двумерных динамических массивов *A* и *B*. Все массивы имеют одинаковую размерность $n \times n$. Размерность массивов ввести с клавиатуры. Значения элементов c_{ij} определить согласно выражению: $c_{ij} = a_{ij} + b_{ij}$, где $i = 0, 1, \dots, n \quad j = 0, 1, \dots, n$. Значения элементов массива *A* ввести с клавиатуры. Значения элементов массива *B* определить по формуле:

$$b_{ij} = \begin{cases} a_{ij}, \text{если } a_{ij} \geq 0; \\ 1, \text{если } a_{ij} < 0. \end{cases}$$

Использовать функции. Вывести на экран значения элементов всех массивов в виде матриц.

Задание 2. (Номер задания ваш номер по списку)

1. Для заданного одномерного массива A из N элементов проверить, что существует, по крайней мере, один элемент A_i, для которого выполняется условие $\sin A_i > 0$. Рекурсивную функцию применять отдельно для каждой из половин массива. Рекурсивные вызовы заканчивать, когда останется только один элемент. Например, для N=8: $\bigcup_{i=1}^{8} \sin(A_i > 0) = \bigcup_{i=1}^{4} \sin(A_i > 0) \vee \bigcup_{i=5}^{8} \sin(A_i > 0)$, ($\vee$ −"или").

2. Для заданного одномерного массива X из N элементов проверить, что для всех элементов массива выполняется условие $-10 < X_i^3 < 20$. Рекурсивную функцию применять каждый раз отдельно для каждой из половин массива. Рекурсивные вызовы заканчивать, когда останется только один элемент. Например, для $N = 8$: $\bigcap_{i=1}^{8}(-10 < X_i^3 < 20) = \bigcap_{i=1}^{4}(-10 < X_i^3 < 20) \wedge \bigcap_{i=5}^{8}(-10 < X_i^3 < 20)$ (^-“и”).

3. Для заданного одномерного массива B из N элементов найти произведение множителей, вычисляемых по формуле $\text{B}_\text{i}^2 + \cos\text{B}_\text{i}$. Рекурсивную функцию применять каждый раз отдельно для каждой из половин массива. Рекурсивные вызовы заканчивать, когда останется только один элемент. Например, для N=8: $\prod_{\text{i}=1}^{8}(\text{B}_\text{i}^2 + \cos\text{B}_\text{i}) = \prod_{\text{i}=1}^{4}(\text{B}_\text{i}^2 + \cos\text{B}_\text{i}) \times \prod_{\text{i}=5}^{8}(\text{B}_\text{i}^2 + \cos\text{B}_\text{i})$.

4. Для заданного одномерного массива X из N элементов найти количество элементов массива, для которых выполняется условие$\sin\frac{\text{X}_\text{i}}{2} < 0$. Рекурсивную функцию применять каждый раз отдельно для каждой из половин массива. Рекурсивные вызовы заканчивать, когда останется только один элемент. Например, для N=8: $\mathbf{Count}_{\text{i}=1}^{8}\left(\sin\frac{\text{X}_\text{i}}{2} < 0\right) = \mathbf{Count}_{\text{i}=1}^{4}\left(\sin\frac{\text{X}_\text{i}}{2} < 0\right) + \mathbf{Count}_{\text{i}=5}^{8}\left(\sin\frac{\text{X}_\text{i}}{2} < 0\right)$.

5. Для заданного одномерного массива A из N элементов найти значение минимального элемента массива и его номер. Рекурсивную функцию применять каждый раз отдельно для каждой из половин массива. Рекурсивные вызовы заканчивать, когда останется только один элемент. Например, для N=12: $\mathbf{Min}_{i=1}^{12} A_i = \min(\mathbf{Min}_{i=1}^{6} A_i; \mathbf{Min}_{i=7}^{12} A_i)$.

6. Для заданного одномерного массива B из N элементов найти сумму выражений, вычисляемых по формуле $\sin B_i \cdot \cos B\_i$. Рекурсивную функцию применять каждый раз отдельно для каждой из половин массива. Рекурсивные вызовы заканчивать, когда останется только один элемент. Например, для N=8: $\sum_{i=1}^{8}(\sin B_i \cdot \cos B_i) = \sum_{i=1}^{4}(\sin B_i \cdot \cos B_i) + \sum_{i=5}^{8}(\sin B_i \cdot \cos B_i)$.

7. Для заданного одномерного массива A из N элементов проверить, что существует хотя бы один элемент A_i, для которого выполняется условие $\sqrt[3]{A_i^2 + 2} < 10$. Рекурсивную функцию применять каждый раз отдельно для первой трети массива и для остальной части (2/3) массива. Рекурсивные вызовы заканчивать, когда останется только один или два элемента. Например, для N=6: $\cup_{i=1}^{6}(\sqrt[3]{A_i^2 + 2} < 10) = \cup_{i=1}^{2}(\sqrt[3]{A_i^2 + 2} < 10) \vee \cup_{i=3}^{6}(\sqrt[3]{A_i^2 + 2} < 10)$(∨ −"или").

8. Для заданного одномерного массива X из N элементов проверить, что для всех элементов массива выполняется условие cos X_i > 0. Рекурсивную функцию применять каждый раз отдельно для первой трети массива и для остальной части (2/3) массива. Рекурсивные вызовы заканчивать, когда останется только один или два элемента. Например, для N=6: $\cap_{i=1}^{6}(\cos X_i > 0) = \cap_{i=1}^{2}(\cos X_i > 0) \wedge \cap_{i=3}^{6}(\cos X_i > 0)$ (^-“и”).

9. Для заданного одномерного массива C из N элементов найти произведение множителей, вычисляемых по формуле $\sin C_i - \cos C_i$ Рекурсивную функцию применять каждый раз отдельно для первой трети массива и для остальной части (2/3) массива. Рекурсивные вызовы заканчивать, когда останется только один или два элемента. Например, для N=12: $\prod_{i=1}^{12}(\sin C_i - \cos C_i) = \prod_{i=1}^{4}(B_i^2 + \cos B_i) \times \prod_{i=5}^{12}(\sin C_i - \cos C_i)$.

10. Для заданного одномерного массива B из N элементов найти количество элементов массива, для которых выполняется условие $(\cos B_i^2 > 0) \wedge (B_i < 0)$. Рекурсивную функцию применять каждый раз отдельно для первой трети массива и для остальной части (2/3) массива. Рекурсивные вызовы заканчивать, когда останется только один или два элемента. Например, для N=6: $\mathbf{Count}_{i=1}^{6}((\cos B_i^2 > 0) \wedge (B_i < 0)) = \mathbf{Count}_{i=1}^{2}((\cos B_i^2 > 0) \wedge (B_i < 0)) + \mathbf{Count}_{i=3}^{6}((\cos B_i^2 > 0) \wedge (B_i < 0))$.

11. Для заданного одномерного массива A из N элементов найти значение максимального элемента массива. Рекурсивную функцию применять каждый раз отдельно для первой трети массива и для остальной части (2/3) массива. Рекурсивные вызовы заканчивать, когда останется только один или два элемента. Например, для N=6: $\mathbf{Max}_{i=1}^{6} A_i = \max(\mathbf{Max}_{i=1}^{2} A_i; \mathbf{Max}_{i=3}^{6} A_i)$.

12. Для заданного одномерного массива X из N элементов найти сумму выражений, вычисляемых по формуле X_i^2. Рекурсивную функцию применять каждый раз отдельно для первой трети массива и для остальной части (2/3) массива. Рекурсивные вызовы заканчивать, когда останется только один или два элемента. Например, для N=9: $\sum_{i=1}^{9}(X_i^2) = \sum_{i=1}^{3}(X_i^2) + \sum_{i=4}^{9}(X_i^2)$.

13. Для заданного одномерного массива A из N элементов проверить, что существует по крайней мере один элемент A_i, для которого выполняется условие $A_i \le i^2$. Рекурсивную функцию применять каждый раз отдельно для каждой из половин массива. Рекурсивные вызовы заканчивать, когда останется только один элемент. Например, для N=4: $\cup_{i=1}^{4}(A_i \le i^2) = \cup_{i=1}^{2}(A_i \le i^2) \vee \cup_{i=3}^{4}(A_i \le i^2)(\vee -"или")$.

14. Для заданного одномерного массива Y из N элементов проверить, что для всех элементов массива выполняется условие $Y_i < 0$. Рекурсивную функцию применять каждый раз отдельно для каждой из половин массива. Рекурсивные вызовы заканчивать, когда останется только один элемент. Например, для N=4: $\cap_{i=1}^{4}(Y_i < 0) = \cap_{i=1}^{2}(Y_i < 0) \wedge \cap_{i=3}^{4}(Y_i < 0)$ (^-“и”).

15. Для заданного одномерного массива X из N элементов найти произведение множителей, вычисляемых по формуле $\frac{X_i}{1+i}$. Рекурсивную функцию применять каждый раз отдельно для каждой из половин массива. Рекурсивные вызовы заканчивать, когда останется только один элемент. Например, для N=6: $\prod_{i=1}^{6}\left(\frac{X_i}{1+i}\right) = \prod_{i=1}^{3}\left(\frac{X_i}{1+i}\right) \times \prod_{i=4}^{6}(\frac{X_i}{1+i})$.

Задание 3.
Дан двумерный динамический массив целых чисел A размерностью $n \times k$. Размерность массива ввести с клавиатуры. Значения элементов массива ввести с клавиатуры. Создать динамический массив из элементов, расположенных на главной диагонали матрицы и имеющих четное значение. Вычислить произведение элементов динамического массива. Созданный массив и результат произведения вывести на экран. Использовать функции.

Задание 4.
Создать двумерный динамический массив вещественных чисел. Определить, встречаются ли среди них элементы с нулевым значением. Если встречаются такие элементы, то определить их индексы и общее количество. Переставить элементы этого массива в обратном порядке и вывести на экран. Использовать функции.

Задание 5.
Дан двумерный динамический массив целых чисел. Значения элементов данного массива ввести с клавиатуры. Создать динамический массив из элементов, расположенных в четных столбцах данного массива и имеющих нечетное значение. Вычислить среднее арифметическое элементов динамического массива. Вывести результат на экран. Использовать функции.

Задание 6*.
Определим следующую рекурсивную функцию **F(n):**

$$F(n) = \begin{cases} n\%10 > 0, if(n\%10 > 0) \\ 0, if(n = 0) \\ F\left(\frac{n}{10}\right), Otherwise \end{cases}$$

Определим функцию **S (p, q)** следующим образом: $S(p,q) = \sum_{i=p}^{q} F(i)$

По заданным **p** и **q** необходимо вычислить **S (p, q)**.

Входные данные
Состоит из нескольких тестов. Каждая строка содержит два неотрицательных целых числа **p** и **q** (**p ≤ q**), разделенных пробелом. **p** и **q** являются **32** битовыми знаковыми целыми. Последняя строка содержит два отрицательных целых числа и не обрабатывается.
Выходные данные

Для каждой пары **p** и **q** в отдельной строке вывести значение **S (p, q)**.
Входные данные #1
1 10
10 20
30 40
-1 -1
Выходные данные #1
46
48
52

Задание 7*.
Рекурсивная функция задана следующим образом:

$$f(0,0) = 1$$
$$f(n,r) = \sum_{i=0}^{k-1} f(n-1, r-i)\ when\ [(n>0) and\ (0 \le r < n(k-1)+1)]$$
$$f(n,r) = 0\ othrwise$$

Вычислить значение $x = \left(\sum_{i=0}^{n(k-1)} f(n,i)\right) \bmod m$, где $\boldsymbol{m = 10^t}$

n \ i	-2	-1	0	1	2	3	4	5	6	7	8	9	10
0	0	0	1	0	0	0	0	0	0	0	0	0	0
1	0	0	1	1	1	0	0	0	0	0	0	0	0
2	0	0	1	2	3	2	1	0	0	0	0	0	0
3	0	0	1	3	6	7	6	3	1	0	0	0	0
4	0	0	1	4	10	16	19	16	10	4	1	0	0
5	0	0	1	5	15	30	45	51	45	30	15	5	1

Часть поля таблицы при k=3

Входные данные
Каждая строка содержит три целых числа: $\boldsymbol{k(0 < k < 10^{19})}$, $\boldsymbol{n(0 < n < 10^{19})}$и $\boldsymbol{t(0 < t < 10)}$
Последняя строка содержит три нуля и не обрабатывается.
Выходные данные
Для каждого теста в отдельной строке вывести номер теста и значение x. Формат вывода приведен в примере.

Входные данные
1234 1234 4
2323 99999999999 8
4 99999 9
888 888 8
0 0 0
Выходные данные
Case #1: 736
Case #2: 39087387
Case #3: 494777344
Case #4: 91255296

Задание 8*.
Пусть **f(n)** - наибольший нечетный делитель натурального числа n. По заданному натуральному n необходимо вычислить значение суммы **f(1)** + **f(2)** + ... + **f(n)**.

Входные данные
Первым подаётся число k - количество тестов.
Следующие k строк содержат одно натуральное число n (n ≤ 10^9).

Выходные данные
Для каждого значения n в отдельной строке вывести значение суммы **f(1)** + **f(2)** + ... + **f(n)**.

Входные данные #1
7
1
777
Выходные данные #1
21
1
201537

Рекомендованный методический материал:

ПРОГРАММИРОВАНИЕ СТАТИЧЕСКИХ И ДИНАМИЧЕСКИХ БИБЛИОТЕК

ВОПРОС 1. РАЗРАБОТКА СТАТИЧЕСКИХ БИБЛИОТЕК

При разработке программ рекомендуется разбивать их на части, которые функционально ограничены и закончены. Например, некоторые функции можно расположить в отдельных ***.cpp** файлах. Такой подход обеспечивает ряд преимуществ:

• обычно сложная программа разбивается на несколько отдельных частей (моду лей), которые отлаживаются отдельно и зачастую разными людьми; поэтому в завершении остается лишь собрать готовые модули в единый проект;

• при исправлении в одном модуле не надо снова транслировать (переводить в машинные коды) все остальные (это могут быть десятки тысяч строк);

• при компоновке во многих системах можно подключать модули, написанные на других языках, например, на Паскале (в машинных кодах).

Библиотека объектных файлов − несколько объектных файлов, которые используются для хранения функций и ресурсов отдельно от исполняемого файла. Библиотека содержит символьный индекс, который состоит из названий функций и переменных и т.д., которые содержатся в библиотеке. Это позволяет ускорить процесс компоновки программы, так как поиск функций и переменных в объектных файлах библиотеки происходит намного быстрее, чем поиск в наборе указанных объектных файлов.

Поэтому использование библиотеки позволяет компактно хранить все требуемые объектные файлы в одном месте, и при этом значительно повысить скорость компиляции. Таким образом, можно создавать большие проекты, которые больше не будут отнимать много времени на компиляцию и поиск ошибок. Однако нужно помнить, что не стоит также чересчур разбивать программу, иначе получится несколько десятков файлов, в которых рано или поздно можно запутаться. Рекомендуется в отдельные файлы помещать те функции или классы, с которыми приходится больше всего работать при отладке. После того, как функция будет окончательно отлажена, ее вполне можно перенести в более крупный файл.

Объектные библиотеки по способу использования разделяются на два вида:

- *Статические библиотеки*
- *Динамические библиотеки*

Статическая библиотека – это коллекция объектных файлов, которые присоединяются к программе во время компоновки. Таким образом, статические библиотеки используются только при создании программы. Потом же, при выполнении программы, они участия не принимают, в отличие от динамических библиотек.

Пример 1. Написать программу, вычисляющую площадь квадрата, прямоугольника, треугольника, круга и трапеции. Вид фигуры вводит пользователь с клавиатуры. Создать статическую библиотеку функций подсчета площадей каждой из фигур.

sl.cpp:

```
float kvadrat (float a)
{ return a*a;}
```

s2.cpp:

```
float pryamougolnik (float a, float b)
{ return a*b;}
```

s3.cpp:

```
float treugolnik (float a, float h)
{ return 0.5*a*h;}
```

s4.cpp:

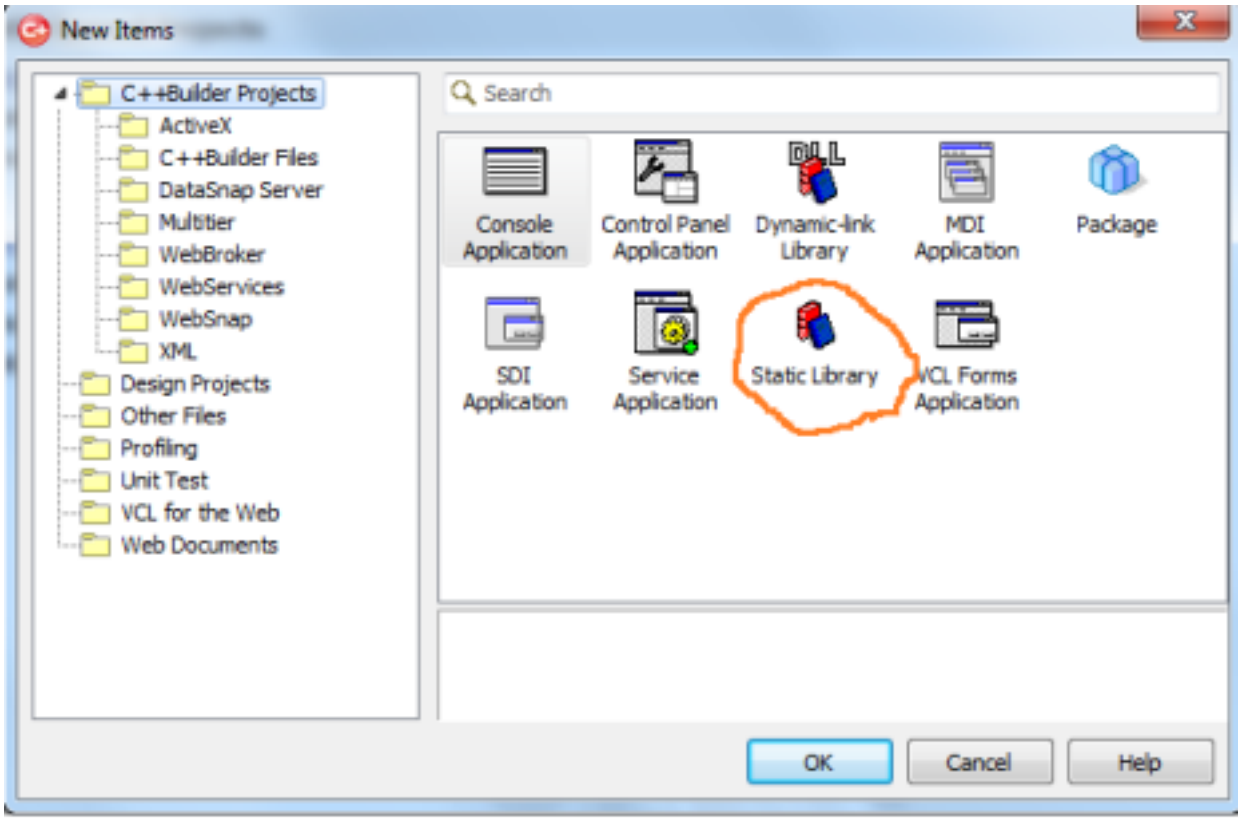
```
float krug (float r)
{ return 3.14*r*r;}
```

s5.cpp:

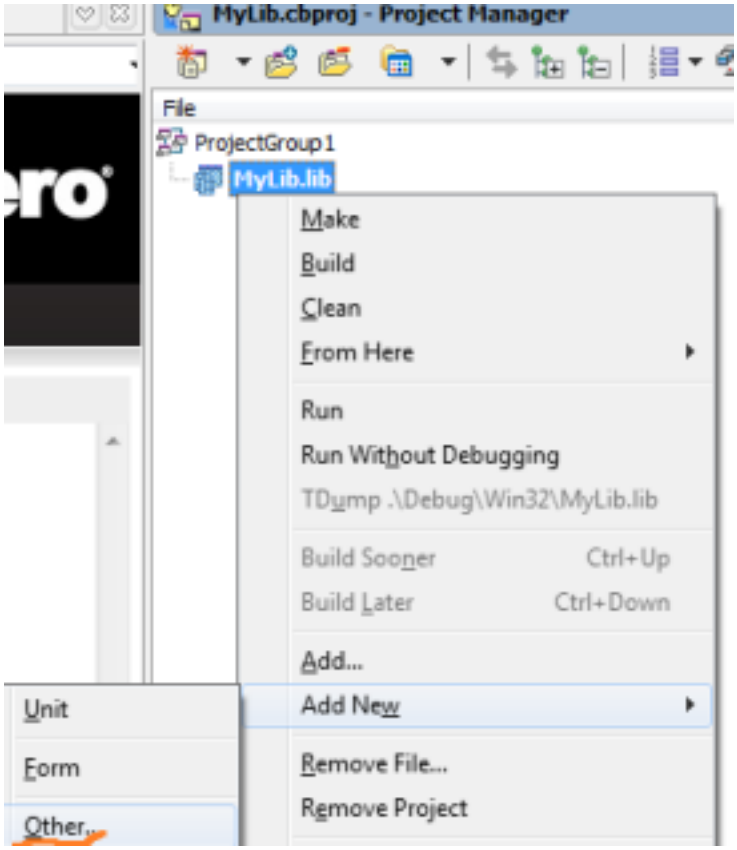
```
float trapeciya (float a, float b, float h)
{ return 0.5*(a+b)*h;}
```

Решение:

Создать проект для разработки статической библиотеки: File/ New/ Other…/Static Library



В окне **Project Manager** изменить имя созданного проекта на MyLib.lib. С помощью команды контекстного меню **Add New** добавить пять .cpp файлов (**sl.cpp, s2.cpp, s3.cpp, s4.cpp, s5.cpp**) и один .h файл (**squares.h**).



В ***.cpp** файлах разместить определения соответствующих функций. В заголовочном файле – прототипы этих функций:

```
float kvadrat (float a);
float pryamougolnik (float a, float b);
float treugolnik (float a, float h);
float krug (float r);
float trapeciya (float a, float b, float h);
```

Сохранить весь проект в папке **D:\412\GZ6**.

Создайте библиотеку с помощью команды: **Project/Build MyLib**. В папке проекта **D:\412\GZ6\Debug\Win32** должен появиться файл **MyLib.lib**. Это и есть статическая библиотека.

Пример *2.*

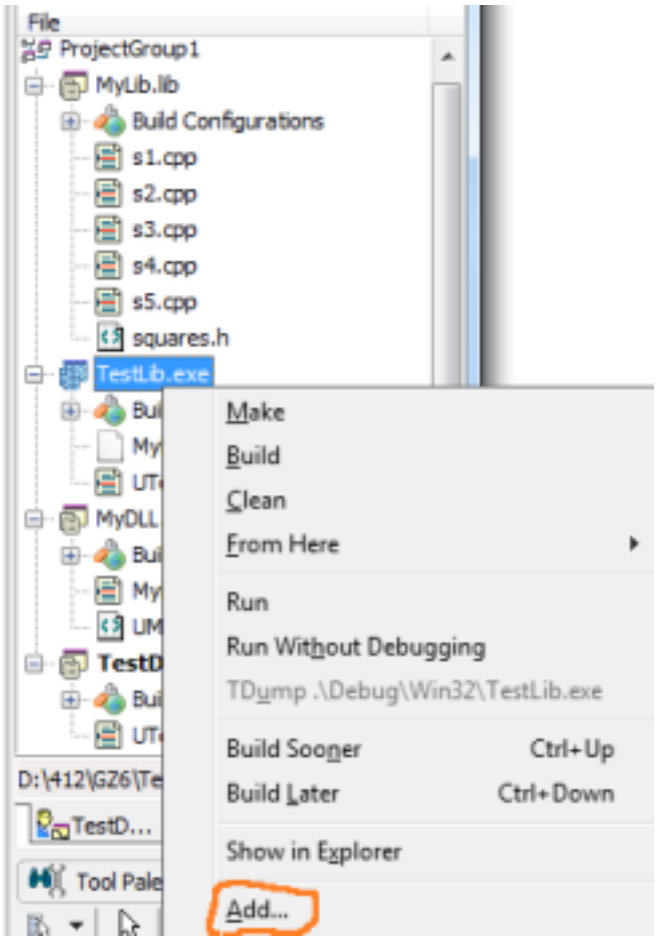
Использовать разработанную ранее статическую библиотеку **MyLib.lib** для вычисления площадей фигур.

Решение:

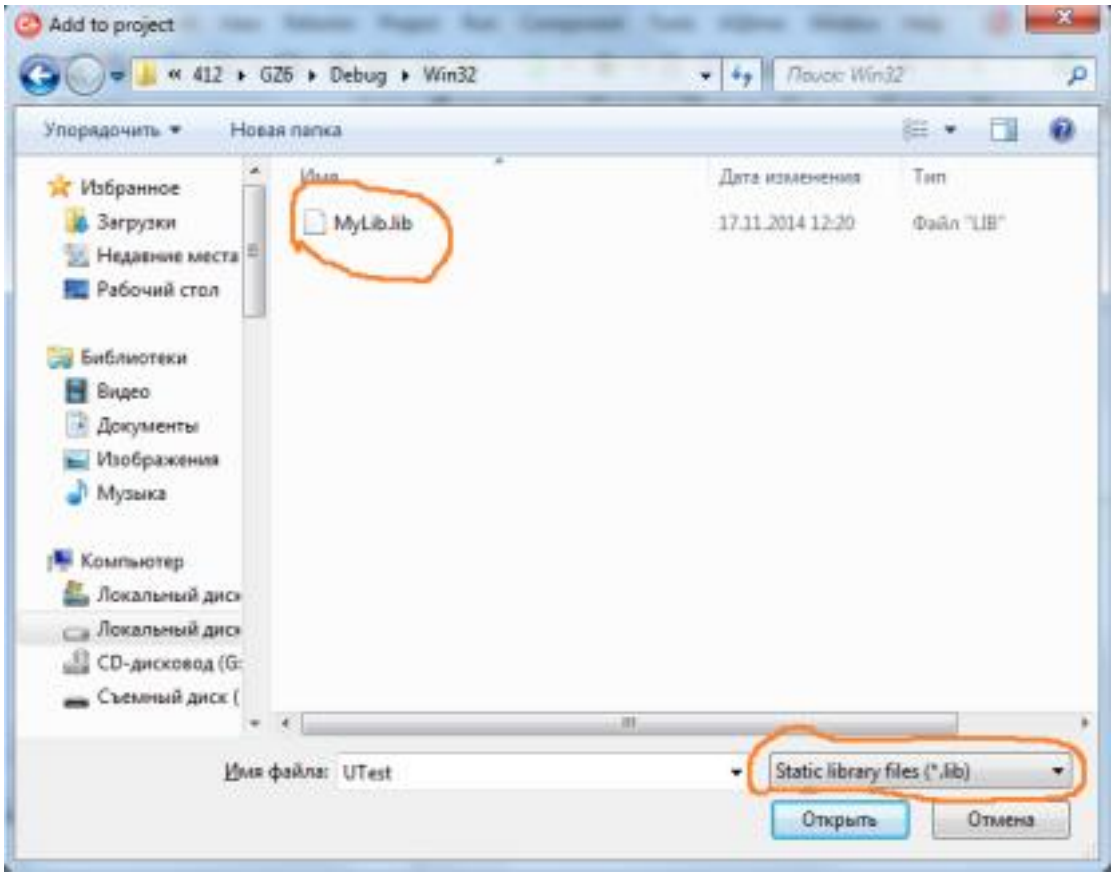
В группу проектов добавить новый проект **TestLib**, **.cpp** файл проекта переименовать в **Utest.cpp**, сохранить в папке **D:\412\GZ6**.

Запустить на выполнение.

Убедиться, что **TestLib.exe** находится в той же папке, что и **MyLib.lib**. С помощью команды **Add…** контекстного меню к проекту **TestLib** добавить к проекту библиотеку **MyLib.lib**.



5



В файл **Utest.cpp** добавить директивы препроцессора

```
#include <conio.h>
#include <iostream.h>
//подключаем заголовочный файл созданной
библиотеки //MyLib.lib
#include "squares.h"
```

Использовать функции разработанной библиотеки **MyLib.lib** в функции **int _tmain()**

```
int _tmain(int argc, _TCHAR* argv[])
{
int i;
cout<<"select figure:\n1-kvadrat\n2-
pryamougolnic\n3- treugilnic\n4-krug\n5-trapeciya\n0-
vyhod\n";
cin>>i;
switch(i)
{
case 1:cout<<kvadrat(5.5);break;
case 2:cout<<pryamougolnik(6.3,4);break;
case 3:cout<<treugolnik(4.3,8.1);break;
case 4:cout<<krug(5.9);break;
case 5:cout<<trapeciya(9.4,2.1,5);break;
case 0:return 0;
}
getch();
return 0;
}
```

ВОПРОС 2. РАЗРАБОТКА ДИНАМИЧЕСКИХ БИБЛИОТЕК

Динамическая библиотека (DLL), с точки зрения программиста, представляет со бой библиотеку функций (ресурсов), которыми может пользоваться любой процесс, загрузивший эту библиотеку. Сама загрузка, кстати, отнимает время и увеличивает расход потребляемой приложением памяти; поэтому бездумное дробление одного приложения на
множество DLL не рекомендуется.

Однако, если какие-то функции используются несколькими приложениями, то, поместив их в одну DLL, мы избавимся от дублирования кода и сократим общий объем приложений − и на диске, и в оперативной памяти. Можно выносить в DLL и редко используемые функции отдельного приложения.

Загрузившему DLL процессу доступны не все ее функции, а лишь явно предоставляемые самой DLL для "внешнего мира" − т. н. ***экспортируемые***. Функции, предназначенные сугубо для "внутреннего" пользования, экспортировать бессмысленно (хотя и не запрещено). Чем больше функций экспортирует DLL − тем медленнее она загружается; поэтому к проектированию ***интерфейса*** (способа взаимодействия DLL с вызывающим кодом) следует отнестись повнимательнее.

Для экспортирования функции из DLL - перед ее описанием следует указать ключевое слово ***__declspec(dllexport)***, как показано в следующем примере

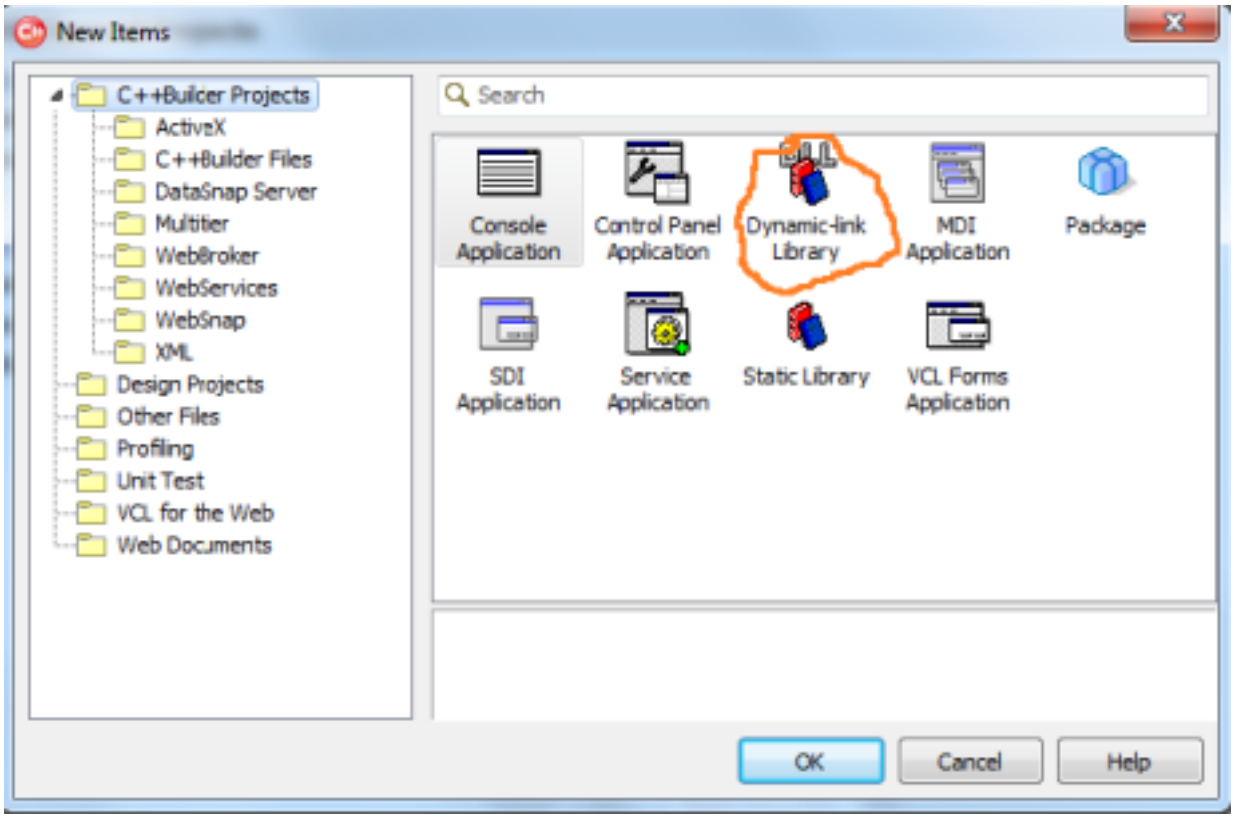
Пример 3. Разработать динамическую библиотеку следующих функций:

$f1(n,m) = n + m;$

$f1(n,m,k) = (n + m)k;$

$f1(n,m) = n - m;$

В имеющуюся группу проектов добавить проект для разработки динамической библиотеки: File/ New/ Other…/Dynamic-linc Library



Так же как и в предыдущем задании добавить с помощью команды контекстного меню **Add New .h** файл (**UMyDLL.h**). Переименовать файл **.cpp** в **MyDLL. cpp.** Сохранить проект под именем **MyDLL.dll** в папке **D:\412\GZ6**. В **.cpp** файле разместить определения функций. В заголовочном файле – их прототипы.

Определения функций в динамической библиотеке нужно дописать в конце со зданного ***.cpp** файла. Они имеют следующий вид:

```
extern "C" double __declspec(dllexport) __stdcall f1 (double n, double
m) {return n+m;}

extern "C" double __declspec(dllexport) __stdcall f2 (double n, double
m,  double k)
{return (n+m)*k;}

 extern "C" double __declspec(dllexport) __stdcall f3 (double n, double
m) {return n-m;}
```

Прототипы функций в заголовочном файле:

```
extern "C" double __declspec(dllexport) __stdcall f1 (double n, double m) ;

extern "C" double __declspec(dllexport) __stdcall f2 (double n, double
m,  double k);

extern "C" double __declspec(dllexport) __stdcall f3 (double n, double m);
```

```
//Здесь конструкция __declspec(dllexport) означает, что функция
может  //экспортироваться из библиотеки, то есть может вызываться
внешними  //приложениями
//stdcall – соглашение о вызовах, применяемое в ОС Windows
для  //вызова функций WinAPI. Аргументы функций передаются через
стек,  //справа налево. Очистку стека производит вызываемая
подпрограмма.
```

Создайте библиотеку с помощью команды: **Project/Build MyDLL**. В папке проекта **D:\412\GZ6\Debug\Win32** должен появиться файл **MyDLL.lib**, который можно использо вать как статическую библиотеку, а также файл **MyDLL.dll**, применяемый для динамиче ского связывания .

Задание 4.

Использовать разработанную ранее динамическую библиотеку **MyDLL.dll** для определения значения выражения:

$$y = \begin{cases} f1(n,m), \text{если } m > n \\ f2(n,m,k), \text{если } m = n \\ f3(n,m), \text{если } m < n \end{cases}$$

Решение:

В группу проектов добавить новый проект **TestDLL**.
.cpp файл проекта переименовать в **Utest1.cpp**.
Все сохранить в папке **D:\412\GZ6**.
Запустить на выполнение.
Убедиться, что **TestDLL.exe** находится в той же папке, что и **MyLib.lib**. Изменить содержимое **Utest1.cpp**

```
#include <conio.h>
#include <iostream.h>

int _tmain(int argc, _TCHAR* argv[])
{
//загрузка DLL
 HINSTANCE load;
 load=LoadLibrary(L"MyDLL.dll");

//получение указателя на функцию
// pfsum – произвольное имя
typedef double (__stdcall *pfsum)(double,double);
pfsum f1,f3;

typedef double (__stdcall
*pfsum1)(double,double,double); pfsum1 f2;

// функция API Windows GetProcAddress используется
для //получения указателя на функцию, где
//load – указатель на загруженный модуль DLL
//f1 – имя функции
```

```
 f1=(pfsum)GetProcAddress(load,"f1");
 f2=(pfsum1)GetProcAddress(load,"f2");
 f3=(pfsum)GetProcAddress(load,"f3");

 double m,n,k;
 cin>>m>>n>>k;
 if (m>n) cout<<f1(n,m);
 else if (m==n) cout<<f2(n,m,k);
 else cout<<f3(n,m);


//освобождение DLL
 FreeLibrary(load);
getch();
    return 0;
}
```

Темы для подготовки к теоретическим вопросам: Понятие функции C++, аргументы функции, аргументы по умолчанию, параметры функции main(), понятие рекурсии, передача по ссылке/указателю/значению, типы возвращаемых значений, тело и прототип функции, необязательные элементы объявления функции (constexpr, extern, static, inline и т.д.), способы возвращения нескольких значений из функции C++.