МИНОБРНАУКИ РОССИИ

Санкт-Петербургский государственный электротехнический университет «ЛЭТИ» имени В. И. Ульянова (Ленина)

МЕТОДИЧЕСКИЕ УКАЗАНИЯ К ПРОВЕДЕНИЮ ЛАБОРАТОРНЫХ И КУРСОВЫХ РАБОТ ПО ДИСЦИПЛИНЕ «ПРОГРАММИРОВАНИЕ» ДЛЯ СТУДЕНТОВ ОЧНО-ЗАОЧНОЙ ФОРМЫ ОБУЧЕНИЯ ПО НАПРАВЛЕНИЮ «ИНФОРМАТИКА И ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА»

Учебно-методическое пособие

Санкт-Петербург
СПбГЭТУ «ЛЭТИ»
2019

Авторы: **С. Э. Миронов, С. А. Кузьмин**

Методические указания к проведению лабораторных и курсовых работ по дисциплине «Программирование» для студентов очно-заочной формы обучения по направлению «Информатика и вычислительная техника»: Учебно-методическое пособие. СПбГЭТУ «ЛЭТИ», 2019. 60 с.

Содержат задания и краткие рекомендации по выполнению лабораторных и курсовых работ по первым трём семестрам дисциплины «Программирование» для студентов очно-заочной формы обучения по направлению «Информатика и вычислительная техника».

В первом семестре рассматривается структура программ на языке программирования C++, изучаются базовые типы данных и основные программные конструкции языка (например, оператор присваивания, условный оператор, оператор цикла, математические функции и т.п.), а также приобретаются навыки разработки программ в интегрированной среде разработки (IDE).

Во втором семестре рассматриваются сложные типы данных (массивы, строки и указатели) и механизм файлового ввода и вывода, а также приобретаются навыки разработки и использования пользовательских функций и методов для обработки текстовой информации.

В третьем семестре рассматриваются различные структуры данных (массивы указателей, структуры и линейные списки), а также приобретаются авыки разработки электронных картотек на основе линейных списков.

Предназначено для студентов очно-заочной формы обучения по направлению 09.03.01 - «Информатика и вычислительная техника».

Одобрено
методической комиссией
факультета компьютерных технологий и информатики
в качестве учебно-методического пособия

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Курс программирования для студентов очно-заочной формы обучения по направлению «Информатика и вычислительная техника» рассчитан на четыре семестра, первые три из которых посвящены изучению основ программирования на языке C++.

В первом семестре студенты осваивают структуру программ, базовые типы данных и основные программные конструкции языка программирования C++ (такие как: оператор присваивания, условный оператор, оператор цикла, математические функции и т.п.), а также получают навыки разработки программ в интегрированной среде разработки (IDE). В ходе выполнения курсовой работы студенты получают навыки выбора и использования переменных подходящих типов данных и программных конструкций для обработки числовой информации.

Во втором семестре студенты изучают сложные типы данных на примере массивов, строк и указателей, учатся создавать пользовательские методы и функции в программе, а также использовать файловый ввод и вывод. В ходе выполнения курсовой работы студенты получают навыки обработки текстовой информации, представленной в виде массива строк.

В третьем семестре студенты осваивают различные структуры данных: массивы указателей, структуры, линейные списки, очереди, стеки и древовидные структуры. В ходе выполнения курсовой работы студенты получают навыки разработки электронных картотек на основе линейных списков.

В данном учебно-методическом пособии приводятся варианты заданий по выполнению каждой лабораторной и курсовой работы соответствующего семестра, а также приведены краткие указания и рекомендации по их выполнению.

ПЕРВЫЙ СЕМЕСТР

Лабораторная работа №1: Знакомство с интегрированной средой

Цель работы

Изучение возможностей и инструментов интегрированной среды разработки. Получение практических навыков разработки, компиляции, запуска и тестирования программы в интегрированной среде (на простом примере).

Задание

Разработать программу, позволяющую вводить данные пользователя, обрабатывать их (согласно своему варианту) и выводить результат на экран.

Варианты заданий

1. Вывести на экран введённое ранее сообщение.
2. Вывести на экран символ, стоящий в n шагах правее от введённого ранее символа (согласно таблице ASCII).
3. Вывести на экран сумму введённых ранее трёх чисел.
4. Вывести на экран разность введённых ранее двух чисел.
5. Вывести на экран произведение введённых ранее трёх чисел.
6. Вывести на экран результат деления введённых ранее двух чисел.
7. Вывести на экран остаток от деления введённых ранее двух чисел.
8. Вывести на экран асолютный модуль введнного ранее числа.
9. Вывести на экран квадрат введённого ранее числа.
10. Вывести на экран квадратный корень введённого ранее числа.
11. Вывести на экран десятичный логарифм введённого ранее числа.
12. Вывести на экран экспоненту от введённого ранее числа (e^x).

Указания по выполнению

В данной лабораторной работе не разрешается использование условных операторов, циклов и пользовательских функций.

Ввод и вывод данных осуществляются через консоль.

Лабораторная работа №2: Вычисление арифметических выражений

Цель работы

Получение практических навыков разработки программы с использованием математических функций и выражений.

Задание

Разработать программу, вычисляющую значение переменной g по своему варианту формулы.

При тестировании работы программы использовать значения переменных x, y и z, указанных в своём варианте.

Варианты формул

1. $g = \dfrac{1+\cos^2(x+y)}{\left|x^3 - \dfrac{2\cdot y}{(x+y)^2}\right|}$, $x = 0.825, y = 2.379$
2. $g = \dfrac{\ln|x|}{x+y}$, $x = -0.729, y = -1.42$
3. $g = x + \dfrac{y^3}{x+y}$, $x = -0.720, y = -1.42$
4. $g = 2^{-x}\sqrt{x+y}$, $x = 3.961, y = -1.625$
5. $g = \sqrt[3]{e^{x-(1/\sin z)}}$, $x = 3.961, y = 0.512$
6. $g = x\left(arctgz + e^{-(x+3)}\right)$, $x = -0.622, y = 5.541$
7. $g = |x-y|\cdot\left(\sin^2 z + tgz\right)$, $x = 7.8, y = 1.3, z = 0.8$
8. $g = \sqrt{x} - x^2 \cdot y$, $x = 17.421, y = 10.36$
9. $g = x\cdot\left(\sin(arctgz) + \cos^2 y\right)$, $x = 0.3, y = 0.25, z = 0.4$
10. $g = e^{|x-y|}\cdot\left(tg^2 z + 1\right)$, $x = -4.5, y = 0.7, z = 0.8$
11. $g = \cos^2\left(arctg\,1/z\right)$, z = 0.160
12. $g = 5\cdot arctgx - 1/4\, arctgy$, $x = -17.825, y = 6.33$
13. $g = (y-x)\dfrac{y-z}{1+(y-x)^2}$, $x = 1.8, y = 18.3, z = -3.2$
14. $g = y^x + |x-y|$, $x = -0.85, y = 1.25$
15. $g = \ln\left(\sqrt{x} + \sqrt{y+2}\right)$, $x = 1.25, y = 33.07$
16. $g = y + \dfrac{x}{y+x^2}$, $x = 0.100, y = 8.750$

17. $g = \ln\left(\sqrt{e^{x-3}} + x^{|y|} + z\right),\ x = 1.5, y = -3.2, z = 8.8$

18. $g = 1 + \dfrac{z^2}{3 + z^2/5} + \arcsin\dfrac{z}{4},\ z = 3.5$

19. $g = \left(x - \dfrac{y}{2}\right) \cdot \ln\left(y^{\sqrt{|x|}}\right),\ x = -15.24, y = 4.642$

20. $g = (|\cos x + 1|) \cdot \left(2 \cdot \sin^2 y\right),\ x = 0.40, y = -0.875$

21. $g = \sqrt{10\sqrt[3]{x} + y^2},\ x = 16.55, y = -2.97$

22. $g = (\sin z)^2 + |x + y|,\ x = 6.5, y = -2.7, z = 0.2$

23. $g = \left|x^{y/z} - x^3 \cdot (x - y)\right|,\ x = 1.825, y = 18.22$

24. $g = e^{x-1} + \sin y,\ x = -2.3, y = -0.8$

25. $g = \sqrt{|y| e^{-(y+x)/2}},\ x = 20.12, y = -12.55$

26. $g = \dfrac{4 - y^2 \cdot e^{3\sin x}}{3 \cdot z^2 + \ln x},\ x = 0.3, y = 4.3, z = 3.8$

27. $g = \ln\left(y^{\sqrt{|x|}}\right) + \dfrac{y}{x + y^3},\ x = -10.5, y = 3.5$

Указания по выполнению

В данной лабораторной работе не разрешается использование условных операторов, циклов и пользовательских функций.

Ввод и вывод данных осуществляются через консоль.

Лабораторная работа №3: Программирование ветвящихся алгоритмов

Цель работы

Получение практических навыков разработки программы с использованием условного оператора.

Задание

Разработать программу, определяющую попадание точки с координатами X и Y в область, которая образуется в результате пересечения множеств:

- круг с радиусом R и с центром, расположенным в начале координат;
- полуплоскость по одну из сторон от прямой $y = ax + b$;
- полуплоскость, ограниченная осями координат.

Варианты расположения плоскости

1. Внутри круга, ниже прямой, правее оси ординат.
2. Внутри круга, ниже прямой, левее оси ординат.
3. Внутри круга, ниже прямой, выше оси абсцисс.
4. Внутри круга, ниже прямой, ниже оси абсцисс.
5. Внутри круга, выше прямой, выше оси абсцисс.
6. Внутри круга, выше прямой, ниже оси абсцисс.
7. Внутри круга, выше прямой, левее оси ординат.
8. Внутри круга, выше прямой, правее оси ординат.
9. Внутри круга, ниже прямой, правее оси ординат и выше оси абсцисс.
10. Внутри круга, ниже прямой, левее оси ординат и ниже оси абсцисс.
11. Внутри круга, выше прямой, левее оси ординат и выше оси абсцисс.
12. Внутри круга, выше прямой, правее оси ординат и ниже оси абсцисс.

Указания по выполнению

В данной лабораторной работе не разрешается использование циклов и пользовательских функций.

Ввод и вывод данных осуществляются через консоль.

В разработанной программе должны быть соблюдены следующие дополнительные условия:

- должен быть дан корректный ответ при любых действительных значениях коэффициентов (заданная прямая может не пересекать окружность);

- перед вводом координат точки, для которой будет осуществляться проверка на принадлежность области, следует проверить факт существования этой области (по введённым значениям a и b, при всех прочих известных условиях). В случае отсутствия области должен осуществляться не ввод точки, а вывод соответствующего сообщения;
- следует учитывать возможность попадания точки на границу области.

Разработку программы рекомендуется начать с анализа всех возможных положений прямой, окружности и осей друг относительно друга. Данный процесс должен помочь выявить граничные условия для определения образования области.

Анализ относительного расположения объектов должен быть осуществлён в графическом виде, разрешение граничных условий должно производиться с помощью формул геометрии.

Необходимо учесть все рекомендации по полноте тестирования.

Граничные условия при подобной постановке задачи не могут быть разрешены обычным равенством вследствие того, что в компьютерном понимании числа 10 и 10,0000001 – это разные числа. Данная проблема решается введением так называемой ширины границы. То есть при вычислении попадания считается не равенство координат точек и ограничивающих прямых ($x = 2$ и $y = 5$), а нахождение их вблизи данной границы ($x - 2 < Eps$ и $y - 5 < Eps$), где Eps – малая ширина границы.

Лабораторная работа №4: Проектирование и отладка циклических алгоритмов

Цель работы

Получение практических навыков разработки программы с использованием оператора цикла.

Задание

Разработать программу, производящую циклические вычисления согласно вашему варианту.

Варианты заданий

1. Вывод таблицы символов ASCII.
2. Вывод таблицы умножения (в форме Пифагора) от 1 до 9.
3. Вывод таблицы значений функции в заданном диапазоне.
4. Найти факториал числа.
5. Найти n-ый член арифметической прогрессии.
6. Найти n-ый член геометрической прогрессии.
7. Найти n-ый член ряда Фибоначчи.
8. Вывести все простые числа до нужного диапазона.
9. Вывести все шестизначные счастливые билеты.
10. Перевести число из двоичной системы в десятичную.
11. Перевести число из десятичной системы в двоичную.
12. Калькулятор с операциями сложения, вычитания, умножения и деления (с зацикленным меню).

Указания по выполнению

В данной лабораторной работе разрешается использование условных операторов и циклов, но не разрешается использование пользовательских функций. Основной алгоритм программы должен содержать хотя бы один цикл.

Ввод и вывод данных осуществляются через консоль.

Допускается проверка введённых данных на корректность с циклическим запросом повторного ввода значения.

Курсовая работа: Обработка числовой информации

Цель работы

Получение практических навыков в технологии управления вычислительным процессом и программировании типовых задач циклической обработки числовых данных.

Задание

Разработать программу, вычисляющую значение функции по формуле сходящегося ряда в соответствии с вариантом задания.

Варианты заданий

1. $\sin(x) = \sum_{k=0}^{\infty}(-1)^k((x^{2k+1})/((2k+1)!))$.
2. $\cos(x) = \sum_{k=0}^{\infty}(-1)^k((x^{2k})/((2k)!))$.
3. $\ln(1+x) = \sum_{k=1}^{\infty}(-1)^{k+1}(x^k/k)$ при $-1 < x < 1$.
4. $e^x = \sum_{k=0}^{\infty} x^k/k!$.
5. $\operatorname{arctg}(x) = \sum_{k=0}^{\infty}(-1)^k((x^{2k+1})/(2k+1))$ при $-1 < x < 1$.
6. $(e^x - e^{-x})/2 = \sum_{k=1}^{\infty} x^{2k-1}/(2k-1)!$.
7. $\ln(x) = 2\sum_{k=0}^{\infty}(((x-1)^{2k+1})/((2k+1)(x+1)^{2n+1}))$ при $x > 0$.
8. $(e^x + e^{-x})/2 = \sum_{k=0}^{\infty}((x^{2k})/((2k)!))$.
9. $1/x = (2/3)\sum_{k=0}^{\infty}((2/3)x-1)^k$ при $1 \le x \le 2$.
10. $1/(a+x) = (1/a)\sum_{k=0}^{\infty}(x/a)^k$ при $|x| < |a|$.
11. $(1+x)^a = \sum_{k=0}^{\infty}((a!)/((a-k)!k!))x^k$ при $|x| < |1|$.
12. $\sin(x)/x = \sum_{k=0}^{\infty}(-1)^k x^{2k}/(2k+1)!$.

Указания по выполнению

В данной курсовой работе разрешается использование условных операторов и циклов, но не разрешается использование пользовательских функций. Вычисление очередного члена ряда и суммы ряда должно осуществляться в цикле.

Ввод и вывод данных осуществляются через консоль.

Допускается проверка введённых данных на корректность с циклическим запросом повторного ввода значения.

При создании программы должны быть выполнены следующие дополнительные условия:

- значение функции, к которой сходится сумма ряда, должно вычисляться с погрешностью, задаваемой пользователем;
- в качестве дополнительного ограничения вводится максимально допустимое число шагов цикла (членов ряда);
- программа должна фиксировать количество членов ряда, необходимое для достижения заданной точности;
- программа должна предусмотреть возможность повторного ввода погрешности, максимального числа шагов и значения аргументов функции, в зависимости от пожеланий пользователя (для этого в программе должно быть предусмотрено меню возможных операций).

Значок “!” означает факториал. Простой пример:
5! = 1*2*3*4*5,
12! = 1*2*3*4*5*6*7*8*9*10*11*12

Обратите внимание на тот факт, что при вычислении отдельно числителя и знаменателя достаточно быстро происходит переполнение типа данных. Например, 12! = 479 001 600, что превышает ограничения сразу нескольких типов данных, таких как int.

Кроме того, при делении одного очень большого числа на другое у нас получается большая погрешность вычисления, которая не всегда компенсируется большой разрядностью используемого типа данных.

Для того чтобы избежать этих проблем, необходимо использовать специальную методику. Попробуем решить данную задачу на примере ряда $\sum_{k=0}^{\infty} x^k / k!$:

- выстраиваем ряд в следующем представлении:

$$\sum_{k=0}^{\infty}\frac{x^k}{k!}=\frac{x^0}{0!}+\frac{x^1}{1!}+\frac{x^2}{2!}+\ldots+\frac{x^N}{N!}+\frac{x^{N+1}}{(N+1)!}+\ldots=y(0)+y(1)+y(2)+\ldots+y(N)+y(N+1)+\ldots \quad (1)$$

- Определяем формулу перехода в формате:

$$\Phi\Pi(N)=\frac{y(N+1)}{y(N)}=\frac{x^{N+1}}{(N+1)!}/\frac{x^N}{N!}=\frac{x^{N+1}}{(N+1)!}*\frac{N!}{x^N}=\frac{x}{N+1}. \quad (2)$$

Дальнейшая работа по вычислению ряда будет выглядеть следующим образом: вычисляем первый член ряда исходя из известных данных. В нашем случае $y(0)=\frac{x^0}{0!}=1$. Второй член ряда равен $y(1)=y(0)\,\Phi\Pi(0)=1\frac{x}{0+1}=x$, исходя из правил вычисления,

$$y(N+1)=y(N)\cdot\Phi\Pi(N). \quad (3)$$

Таким образом можно вычислить сумму ряда до определённого значения N. Погрешность вычисления может определяться по тому, насколько большим получился очередной вычисленный член ряда. Если он меньше погрешности, то после прибавления этого члена вычисление ряда можно считать завершённым.

Тестирование данной работы должно проходить в 2 этапа:

- тестирование итерационного механизма;
- общее тестирование.

Тестирование итерационного механизма заключается в тестировании:

- входа (определение начальных значений);
- перехода от N-го к (N+1)-му члену ряда;
- выхода из суммирования.

Всё это должно осуществляться с использованием необходимых средств отладки и с вычислением члена ряда по формуле, полученной в соответствии с (1).

Общее тестирование проводится путём вычисления значения по формуле, представленной слева от знака равенства, и сравнения его со значением, полученным в результате выполнения программы.

В данной работе необходимо учитывать замечания по выбору единичных и нулевых исходных данных.

ВТОРОЙ СЕМЕСТР

Лабораторная работа №1: Обработка массивов

Цель работы

Получение практических навыков разработки программы, обрабатывающей числовые данные, представленные в форме массива.

Задание

Разработать программу, обрабатывающую элементы двумерного массива размером n × n, в соответствии с вариантом и прилагаемым рисунком (рис. 1, *а-е*). Размерность массива n должна вводиться пользователем с клавиатуры (если среда выполнения программы это не позволяет, тогда данную размерность можно задать в программе заранее как константу). Элементы массива также должны вводиться пользователем с клавиатуры. Исходный и преобразованный массивы должны быть выведены на экран после обработки.

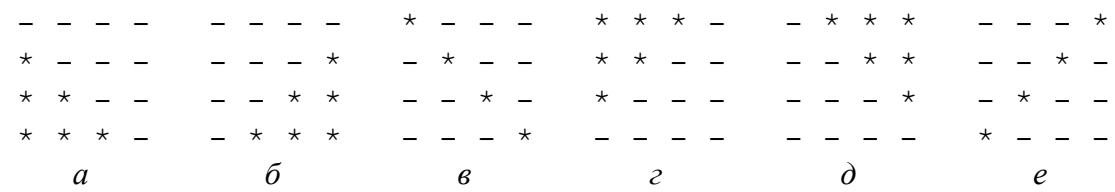

Рис. 1. Иллюстрация вариантов заданий по обработке массива

Варианты заданий

1. Обнулить элементы (рис. 1, *а*).
2. Изменить знак на противоположный (рис. 1, *б*).
3. Сложить с заданным числом (рис. 1, *в*).
4. Вычесть заданное число (рис. 1, *г*).
5. Умножить на заданное число (рис. 1, *д*).
6. Обнулить элементы (рис. 1, *е*).
7. Изменить знак на противоположный (рис. 1, *а*).
8. Сложить с заданным числом (рис. 1, *б*).
9. Вычесть заданное число (рис. 1, *в*).
10. Умножить на заданное число (рис. 1, *г*).
11. Обнулить элементы (рис. 1, *д*).
12. Изменить знак на противоположный (рис. 1, *е*).

Указания по выполнению

В данной лабораторной работе разрешается использование программных конструкций, изученных в предыдущем семестре, но пока что не разрешается использование пользовательских функций.

Ввод и вывод данных осуществляются через консоль.

Относительно размерности массива (n) допускается проверка введённого значения на корректность с циклическим запросом его повторного ввода.

Упростить проектирование на первом этапе можно, взяв массив для тестирования размером 4 × 4 элемента. Подобный подход позволит отладить основные функции программы и не усложнит дальнейшее усовершенствование. Дополнительно упростить разработку программы можно и за счёт временного комментирования цикла ввода и использования заранее созданного массива. В таком случае по известным данным всегда можно обнаружить ошибку и не загружать процесс тестирования постоянным вводом данных.

Для работы с элементами массива следует использовать метод полного перебора элементов на основе конструкции for. Но при этом нерационально объединять циклы для ввода и обработки информации - это усложнит разработку и тестирование программы.

Для определения принадлежности элемента условию следует обратить внимание на порядковый номер элемента в строке и столбце, а также на их взаимное соотношение.

Настоятельно рекомендуется создавать программу с использованием двух массивов: исходного и результирующего.

Лабораторная работа №2: Указатели

Цель работы

Получение практических навыков разработки программы, обрабатывающей числовые данные с помощью указателей.

Задание

Разработать программу, обрабатывающую элементы двумерного массива размером n × n, в соответствии с вариантом и прилагаемым рисунком (рис. 2, *а-е*). Размерность массива n должна вводиться пользователем с клавиатуры (если среда выполнения программы это не позволяет, тогда данную размерность можно задать в программе заранее как константу). Элементы массива также должны вводиться пользователем с клавиатуры. Доступ к элементам массива должен осуществляться через указатели. Исходный и преобразованный массивы должны быть выведены на экран после обработки. Ввод, вывод и обработка элементов массива должны осуществляться в пользовательских функциях.

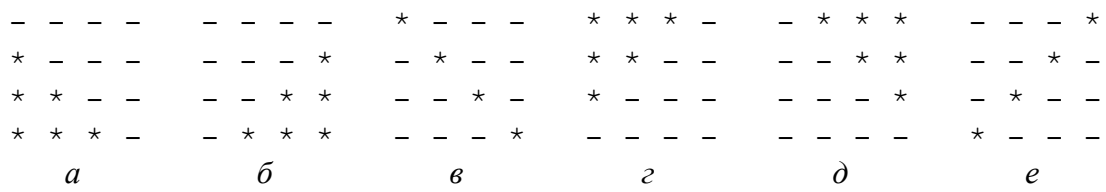

Рис. 2. Иллюстрация вариантов заданий по обработке массива

Варианты заданий

1. Обнулить элементы (рис. 2, *а*).
2. Изменить знак на противоположный (рис. 2, *б*).
3. Сложить с заданным числом (рис. 2, *в*).
4. Вычесть заданное число (рис. 2, *г*).
5. Умножить на заданное число (рис. 2, *д*).
6. Обнулить элементы (рис. 2, *е*).
7. Изменить знак на противоположный (рис. 2, *а*).
8. Сложить с заданным числом (рис. 2, *б*).
9. Вычесть заданное число (рис. 2, *в*).
10. Умножить на заданное число (рис. 2, *г*).
11. Обнулить элементы (рис. 2, *д*).
12. Изменить знак на противоположный (рис. 2, *е*).

Указания по выполнению

В данной лабораторной работе разрешается использование всех изученных ранее программных конструкций, включая теперь и пользовательские функции.

Ввод и вывод данных осуществляются через консоль.

Относительно размерности массива (n) допускается проверка введённого значения на корректность с циклическим запросом его повторного ввода.

Для работы с элементами массива следует использовать метод полного перебора элементов на основе конструкции for. Для определения принадлежности элемента условию следует обратить внимание на порядковый номер элемента в строке и столбце, а также на их взаимное соотношение.

Ввод, вывод и обработка элементов массива должны осуществляться в отдельных пользовательских функциях! Доступ к элементам массива должен осуществляться через указатели. Передача массива в аргументах функций также должна осуществляться через указатели.

Настоятельно рекомендуется создавать программу с использованием двух массивов: исходного и результирующего.

Лабораторная работа №3: Обработка строковых данных

Цель работы

Получение практических навыков разработки программы, обрабатывающей текстовую информацию, представленную в виде строки (массива символов).

Задание

Разработать программу, производящую обработку текстовой строки в соответствии с вариантом задания. Строка и все дополнительные данные должны вводиться с клавиатуры пользователем.

Результат обработки вместе с исходными данными должны быть выведены на экран терминала по окончании работы программы.

Варианты заданий

1. Удалить заданную подстроку.
2. Заменить одну заданную подстроку на другую.
3. Вставить после заданной подстроки другую заданную подстроку.
4. Вставить перед заданной подстрокой другую заданную подстроку.
5. Удалить заданное слово.
6. Заменить одно заданное слово на другое.
7. Вставить после заданного слова другое заданное слово.
8. Вставить перед заданным словом другое заданное слово.
9. Удалить каждое второе вхождение заданного слова в строке.
10. Удалить каждое второе вхождение заданной подстроки в каждом слове текста.
11. Вставить перед вторым вхождением заданного слова другое заданное слово.
12. Вставить перед вторым вхождением заданной подстроки другую заданную подстроку.

Указания по выполнению

В данной лабораторной работе предполагается использование функций для работы со строками библиотеки «string.h» - в частности, функций для определения размера строки, конкатенации и копирования строк.

Ввод и вывод данных осуществляются через консоль.

Строки и подстроки в программе должны быть представлены или с помощью специального типа данных «string», или же с помощью массива

символов типа «char». Обратите внимание на то, что для полноценной работы со строкой как массивом символов недостаточно просто указать количество элементов в строке - необходимо также выставить признак конца строки (символ ‘\0’).

Необходимо предусмотреть хранение исходной строки, вводимого пользователем слова или же подстроки, а также полученную новую строку в отдельных переменных.

Для выполнения некоторых заданий может понадобиться осуществить доступ к отдельному элементу (символу) строки. В этом случае доступ должен осуществляться с помощью указателей.

Ввод каждой строки (подстроки, слова), обработка исходной строки и вывод результата должны осуществляться в отдельных пользовательских функциях.

Для упрощения проектирования программы рекомендуется представить проводимые алгоритмические операции в графическом виде.

Лабораторная работа №4: Обработка файлов

Цель работы

Получение практических навыков разработки программы, осуществляющей ввод и вывод текстовых данных через внешние файлы.

Задание

Разработать программу, обрабатывающую содержимое текстового файла в соответствии с заданием.

Результаты обработки сохранить в отдельном файле. Имена файлов для чтения и записи должны вводиться пользователем с клавиатуры.

Места изменений по желанию пользователя могут быть зафиксированы на экране терминала (не в файле).

В программе должны быть корректно разрешены попытки открытия несуществующего файла.

Варианты заданий

1. Удалить слова, содержащие заданный символ.
2. Удалить числа, содержащие заданную цифру.
3. Удалить все строчные русские буквы, а все прописные превратить в строчные.
4. Удалить все прописные латинские буквы, а все строчные превратить в прописные.
5. Удалить все русские буквы.
6. Удалить все латинские буквы.
7. Удалить символ предшествующий заданному символу.
8. Удалить символ, следующий за заданным символом.
9. Вставить заданный символ перед символом, совпадающим с другим заданным символом.
10. Вставить заданный символ после символа совпадающего с другим заданным символом.
11. Зафиксировать порядковый номер вхождения заданного символа. Формат: “; (5)”, где “;” - заданный символ, “(5)” - пятое вхождение.
12. Вставить заданный символ после слова, содержащего другой заданный символ.

Указания по выполнению

В данной лабораторной работе предполагается использование функций для работы со строками библиотеки «string.h», изученных в ходе выполнения предыдущей лабораторной работы.

Ввод исходной строки осуществляется через внешний файл. Вывод полученной новой строки осуществляется в новый файл. Ввод других необходимых данных для выполнения программы (например, слов или же отдельных символов) может осуществляться как через отдельный файл, так и напрямую с консоли. При этом, необходимо вывести на консоль весь ход выполнения программы (комментарии по действиям программы, копии вводимых и выводимых данных).

Для реализации функциональности программы рекомендуется ознакомиться с разделами справочной литературы в области работы с файлами: изучить типы файлов, способы открытия, чтения, записи и закрытия файлов.

Работу программы следует организовать в концепции посимвольной обработки с использованием буфера соответствующей длины, если это требуется по заданию.

При разработке программы необходимо учитывать и корректно обрабатывать исключительные ситуации, которые могут возникнуть при открытии на чтение или запись несуществующего файла.

Строки и слова в программе должны быть представлены или с помощью специального типа данных «string», или же с помощью массива символов типа «char». В последнем случае обратите внимание на рекомендации по завершению ввода строки, которые приведены в предыдущей лабораторной работе.

Необходимо предусмотреть хранение исходной строки, вводимого пользователем слова или же символа, а также полученную новую строку в отдельных переменных.

Для выполнения некоторых заданий может понадобиться осуществить доступ к отдельному элементу (символу) строки. В этом случае доступ должен осуществляться с помощью указателей.

Ввод каждой строки (подстроки, слова), обработка исходной строки и вывод результата должны осуществляться в отдельных пользовательских функциях.

Для упрощения проектирования программы рекомендуется представить проводимые алгоритмические операции в графическом виде.

Курсовая работа: Обработка текстовой информации

Цель работы

Углубление знаний в технологии программирования типовых задач обработки текстовых данных.

Задание

Разработать программу, осуществляющую обработку текстовой информации, согласно своему варианту задания.

Задачи данной курсовой работы включают выполнение следующих этапов:

1) ввод исходного текста, хранящегося в виде файла;
2) ввод исходных данных для обработки текста (например, подстрока для определения заданного места обработки; подстрока, соответствующая заданной обработке; требуемые символьные данные; и т.п.);
3) обработку текста, соответствующую индивидуальному заданию (опираясь на технику указателей и стандартные подпрограммы обработки ASCII строк);
4) сохранение файла, соответствующего обработанному тексту;
5) вывод обработанного текста на экран.

Все задания содержат индивидуальные требования к выполнению.

Курсовая работа оформляется в виде пояснительной записки, в которой отражены все полученные результаты разработки.

Варианты заданий

Задание №1. Текст, представляющий собой последовательность строк, длина каждой из которых не превышает некоторого задаваемого при выполнении значения, вводится с клавиатуры или из файла.

Преобразовать текст в соответствии с последовательностью команд редактирования, которые должны позволять вставлять, удалять и заменять заданные строки и их последовательности. *Для локализации места внесения изменения в командах редактирования задаются номера строк текста.*

Примеры команд редактирования:

- вставить последовательность строк, указанных в команде, после строки с заданным номером;
- удалить заданное число строк, начиная со строки с заданным номером;
- заменить строку с заданным номером на строку (строки), указанную в команде;

- вывести на экран заданное число строк, начиная со строки с заданным номером;
- записать текст в файл с именем, указанным в команде.

Дополнительные требования: редактор не должен работать в интерактивном режиме. Операция редактирования (программа редактирования) текста должна выполняться по специальной команде пользователя.

Задание №2. Текст, представляющий собой последовательность строк, длина каждой из которых не превышает некоторого задаваемого при выполнении значения, вводится с клавиатуры или из файла.

Преобразовать текст в соответствии с последовательностью команд редактирования, которые должны позволять вставлять, удалять и заменять заданные строки и их последовательности. *Для локализации места внесения изменения в командах редактирования используется понятие текущей строки.*

Примеры команд редактирования:

- вставить последовательность строк, заданных в команде, после текущей строки;
- вставить последовательность строк, заданных в команде, перед текущей строкой;
- удалить заданное число строк, начиная (заканчивая) текущей строкой;
- заменить текущую строку на строку, указанную в команде;
- перейти на заданное число строк вверх (вниз) относительно текущей строки;
- вывести на экран заданное число строк, начиная с текущей строки;
- записать текст в файл с именем, указанным в команде.

Дополнительные требования аналогичны заданию №1.

Задание №3. Текст, представляющий собой последовательность строк, длина каждой из которых не превышает некоторого задаваемого при выполнении значения, вводится с клавиатуры или из файла.

Преобразовать текст в соответствии с последовательностью команд редактирования, которые должны позволять вставлять, удалять и заменять заданные слова и подстроки. *Для локализации места внесения изменения в командах редактирования задаются номера строк текста и контекст*.

Примеры команд редактирования:

- вставить в последовательность строк, номера которых указаны в команде, после (перед) заданного контекста заданную подстроку;
- удалить в последовательности строк, номера которых указаны в команде, после (перед) заданного контекста заданное число символов, заданную подстроку или префикс (суффикс) строки;
- заменить в строках, номера которых заданы в команде, заданную подстроку на подстроку, указанную в команде;
- вывести на экран заданное число строк, начиная со строки с заданным номером;
- записать текст в файл с именем, указанным в команде.

Дополнительные требования аналогичны заданию №1.

Задание №4. Текст, представляющий собой последовательность строк, длина каждой из которых не превышает некоторого задаваемого при выполнении значения, вводится с клавиатуры или из файла.

Преобразовать текст в соответствии с последовательностью команд редактирования, которые должны позволять вставлять, удалять и заменять заданные слова и подстроки. *Для локализации места внесения изменения в командах редактирования используются понятия текущей строки текста и контекст.*

Примеры команд редактирования:

- вставить в последовательность строк, начиная (заканчивая) с текущей, после (перед) заданного контекста заданную подстроку;
- удалить в последовательности строк, начиная (заканчивая) с текущей, после (перед) заданного контекста заданное число символов, заданную подстроку или префикс (суффикс) строки;
- заменить в заданном числе строк, начиная (заканчивая) с текущей, заданную подстроку на подстроку, указанную в команде;
- вывести на экран заданное число строк, начиная с текущей строки;
- записать текст в файл с именем, указанным в команде.

Дополнительные требования аналогичны заданию №1.

Задание №5. Текст, представляющий собой последовательность строк, вводится с клавиатуры или из файла.

Выполнить пословный перевод текста в соответствии с заданным словарем.

Дополнительные требования:

- словарь вводится с клавиатуры или из файла;
- редактор словаря должен позволять вставлять, удалять и заменять заданные слова и их переводы;
- после работы словарь по запросу может записываться в файл с задаваемым именем.

Задание №6. Разработать правила форматирования исходного текста единицы компиляции знакомого языка программирования и программу, выполняющую форматирование.

Дополнительные требования:

- текст вводится с клавиатуры или из файла;
- текст единицы компиляции синтаксически правилен;
- перед выполнением форматирования должна существовать возможность визуализации правил форматирования и выбора тех правил, которые будут использованы в данном сеансе работы;
- после форматирования текст единицы компиляции должен записываться в файл с заданным именем.

Задание №7. Текст представляет собой последовательность отдельных предложений, содержащих слова и знаки пунктуации. Преобразовать текст в соответствии с последовательностью команд редактирования, которые должны позволять вставлять, удалять и заменять заданные слова в определенных предложениях.

Команды редактирования:

- удалить заданное слово в предложении;
- заменить в предложении знак пунктуации на новый;

Указание определенного предложения:

- предложение, начинающееся с указанного слова;

Указание заданного слова:

- начинающееся с заданной последовательности символов.

Задание №8. Текст представляет собой последовательность отдельных предложений, содержащих слова и знаки пунктуации. Преобразовать текст в соответствии с последовательностью команд редактирования, которые должны позволять вставлять, удалять и заменять заданные слова в определенных предложениях.

Команды редактирования:

- вставить в предложении новое слово перед заданным словом;
- заменить заданное слово в предложении на новое слово;

Указание определенного предложения:

- предложение с заданным номером;

Указание заданного слова:

- «отсутствующее» в предложении.

Задание №9. Текст представляет собой последовательность отдельных предложений, содержащих слова и знаки пунктуации. Преобразовать текст в соответствии с последовательностью команд редактирования, которые должны позволять вставлять, удалять и заменять заданные слова в определенных предложениях.

Команды редактирования:

- вставить в предложении новое слово после заданного слова;
- удалить в предложении знак пунктуации (указанный и/или все);

Указание определенного предложения:

- предложение, содержащее указанное слово;

Указание заданного слова:

- содержащее заданную последовательность символов.

Задание №10. В тексте, разделенном на слова, подсчитать количество гласных и согласных. Упорядочить слова по содержащимся в них гласным и/или согласным, а также по их относительному соотношению. Для слов указать количество вхождений в текст.

Задание №11. В тексте с символами пунктуации выделить все слова, указав, сколько раз каждое встречается в тексте. Для слов с нечетным числом букв отсортировать по алфавиту те слова, у которых в середине находится гласная.

Задание №12. В тексте с символами пунктуации выделить и подсчитать количество разных слов, заканчивающихся на гласные и согласные. В этих двух группах упорядочить слова по относительному соотношению содержащихся в них согласных и гласных. Для слов с равным числом согласных и гласных упорядочить слова по алфавиту.

Задание №13. В тексте с символами пунктуации выделить все слова, указав, сколько раз каждое встречается в тексте. Подсчитать сколько раз в тексте встречается каждая буква слова. Отсортировать слова по их первой и второй буквам (в соответствии с частотой их появления в тексте).

Задание №14. Задан текст из слов, состоящих только из букв (прописные и строчные различаются) и разделенных пробелом. Необходимо осуществить эффективное кодирование этого текста (например, по частоте встречающихся в нем символов) в двоичном алфавите с формированием последовательности двоичных слов, занимающих «минимальный» объем памяти. Для проверки должна быть выполнена обратная процедура раскодирования.

Задание №15. Задан текст, разделенный на абзацы и состоящий из предложений, содержащих слова и знаки пунктуации. Составить списки, характеризующие частоту использования различных слов, символов в них и разных знаков пунктуации в соответствующих предложениях, абзацах и тексте в целом. Найти наиболее часто встречающиеся последовательности символов в словах.

Задание №16. Текст состоит из последовательности предложений. В каждом предложении выделить все слова, указав, сколько раз каждое слово встречается в этом предложении. Подсчитать сколько раз в каждом предложении встречаются слова, оканчивающиеся на разные буквы. Переставить слова в предложении по их последней букве в соответствии с частотой их появления.

Указания по выполнению

В данной курсовой работе предполагается использование функций для работы со строками библиотеки «string.h», изученных в ходе выполнения предыдущих лабораторных работ.

Ввод исходной строки осуществляется через внешний файл. Вывод полученной новой строки осуществляется в новый файл. Ввод других необходимых данных для выполнения программы (например, слов или же отдельных символов) может осуществляться как через отдельный файл, так и напрямую с консоли. При этом, необходимо вывести на консоль весь ход выполнения программы (комментарии по действиям программы, копии вводимых и выводимых данных).

При разработке программы необходимо учитывать и корректно обрабатывать исключительные ситуации, которые могут возникнуть при открытии на чтение или запись несуществующего файла.

Строки и слова в программе должны быть представлены или с помощью специального типа данных «string», или же с помощью массива символов типа «char». В последнем случае обратите внимание на рекомендации по завершению ввода строки, которые приведены в предыдущих лабораторных работах.

Необходимо предусмотреть хранение исходного массива строк, вводимых пользователем слов (подстрок) или же отдельных символов, а также полученный новый массив строк в отдельных переменных.

Для выполнения некоторых заданий может понадобиться осуществить доступ к отдельной строке, а также отдельномк элементу (символу) строки. В этом случае доступ должен осуществляться с помощью указателей.

Каждая операция из меню программы должна осуществляться в отдельных пользовательских функциях.

Для упрощения проектирования программы рекомендуется представить проводимые алгоритмические операции в графическом виде.

ТРЕТИЙ СЕМЕСТР

Лабораторная работа №1: Массивы указателей. Динамическое управление памятью

Цель работы

Получение практических навыков разработки программы, обрабатывающей данные, представленные массивом указателей.

Задание

Разработать программу, обрабатывающую текстовую информацию, представленную массивом указателей на строки (массивы символов).

Программа должна считать данные (текстовую информацию) из входного файла, представленного набором строк (каждая строка представляет собой последовательность символов, среди которых могут быть буквы, пробелы, знаки препинания и т.п.). Далее над текстом выполняется операция, определённая в вариантах задания. После чего результаты выполнения операции должны быть отражены на экране и записаны в новый файл.

Доступ к каждой строке массива, а также к отдельному элементу (символу) строки должно осуществляться через указатели.

Варианты заданий

1. Вставить новую строку в текст по заданному номеру.
2. Заменить строку по заданному номеру в тексте на другую строку.
3. Удалить строку из текста по заданному номеру.
4. Вставить новый символ в заданное место в каждой строке текста.
5. Заменить все вхождения заданного символа на другой символ в каждой строке текста.
6. Заменить все строчные (латинские) буквы на прописные, а все прописные (латинские) буквы превратить в строчные.
7. Удалить все вхождения заданного символа в каждой строке текста.
8. Удалить символ, предшествующий заданному символу, в каждой строке текста.
9. Удалить символ, следующий за заданным символом, в каждой строке текста.
10. Удалить все слова, содержащие заданный символ, в каждой строке текста.
11. Вставить после заданной подстроки другую заданную подстроку в каждой строке текста.

12. Вставить перед заданной подстрокой другую заданную подстроку в каждой строке текста.
13. Заменить все вхождения заданной подстроки на другую подстроку в каждой строке текста.
14. Удалить все вхождения заданной подстроки в каждой строке текста.

Указания по выполнению

1. Основные данные программы должны быть представлены в программе в виде массива указателей на строки (массивы символов). Причём, каждая строка может иметь переменную длину.

Для того, чтобы сформировать эту структуру в памяти, необходимо вначале уточнить размер текста (число строк), а потом узнать длину каждой строки по-отдельности. Число строк - это размер массива указателей. А длина каждой строки - это размер соответствующего одномерного массива символов (который должен быть создан в памяти через соответствующий указатель).

Для организации массивов запрещается использовать готовые контейнерные типы данных. Всю структуру и связи нужно будет прописать в программе вручную.

2. Для выполнения некоторых заданий может понадобиться осуществить доступ к отдельной строке, а также к отдельному элементу (символу) строки. В этом случае доступ должен осуществляться с помощью указателей.

3. Непосредственная операция по обработке текста должна быть записана в отдельной функции! На входе функции (для большинства операций) задаётся исходная строка, а также необходимые для её работы данные и параметры. На выходе функция должна возвращать полученную строку.

4. Объявление функции желательно поместить в отдельный заголовочный файл, который подключается к основному файлу программы. А текст функции (как и текст функции main()) должен быть записан в основном файле программы.

5. Ввод строк в память программы нужно произвести из файла, а ввод необходимых для выполнения операции данных и параметров - с клавиатуры.

6. Полученные строки необходимо записать в новый файл в следующем формате:

- вначале должны быть записаны исходные строки;
- потом должно быть записано название совершённой операции;

- потом должны быть записаны введённые данные и параметры;
- после чего уже должны быть записаны полученные строки.

7. При разработке программы необходимо учитывать и корректно обрабатывать исключительные ситуации, которые могут возникнуть при открытии на чтение или запись несуществующего файла.

8. При этом, необходимо вывести на консоль весь ход выполнения программы (комментарии по действиям программы, копии вводимых и выводимых данных).

9. Для упрощения проектирования программы рекомендуется представить проводимые алгоритмические операции в графическом виде.

Лабораторная работа №2: Функции, обеспечивающие обработку массива

Цель работы

Получение практических навыков разработки алгоритмов для обработки массивов.

Задание

Разработать программу, обрабатывающую элементы многомерного массива в соответствии с вариантом.

Размерности массива должны вводиться пользователем с файла, или же с клавиатуры. На их основе массивы должны быть созданы в динамической памяти программы с помощью указателя. Доступ к элементам массива также осуществляется с помощью указетеля.

Элементы массива могут вводиться пользователем как с файла, так и с клавиатуры.

Исходный и преобразованный массивы должны быть выведены на экран и записаны в выходной файл после обработки.

Варианты заданий

1. Разработать программу для сортировки элементов одномерного числового массива по возрастанию.
2. Разработать программу для сортировки элементов одномерного числового массива по убыванию.
3. Разработать программу для объединения двух одномерных числовых массивов в один (с учётом упорядочивания элементов по возрастанию).
4. Разработать программу для объединения двух одномерных числовых массивов в один (с учётом упорядочивания элементов по убыванию).
5. Разработать программу для упорядочивания элементов одномерного символного массива по алфавиту.
6. Разработать программу для упорядочивания элементов одномерного символного массива по алфавиту в обратном порядке.
7. Разработать программу для объединения двух одномерных символьных массивов в один (с учётом упорядочивания символов по алфавиту).
8. Разработать программу для объединения двух одномерных символьных массивов в один (с учётом упорядочивания символов по алфавиту в обратном порядке).

9. Разработать программу для транспонирования прямоугольной матрицы (двумерного массива).
10. Разработать программу для сложения двух прямоугольных матриц (двумерных массивов).
11. Разработать программу для вычитания двух прямоугольных матриц (двумерных массивов).
12. Разработать программу для умножения прямоугольной матрицы (двумерного массива) на скалярную величину.
13. Разработать программу для умножения двух прямоугольных матриц (двумерных массивов).
14. Разработать программу для тензорного произведения двух прямоугольных матриц (двумерных массивов).

Указания по выполнению

1. Основные данные программы должны быть представлены в программе в виде динамических массивов, создаваемых и удаляемых из памяти (в конце работы программы) с помощью указателей. При этом, размерности создаваемых массивов не должны быть заранее заданы в программе, а должны определяться пользователем.

Для организации массивов запрещается использовать готовые контейнерные типы данных. Всю структуру и связи нужно будет прописать в программе вручную.

2. Доступ к отдельным элементам массивов должен осуществляться с помощью указателей.

3. Непосредственная операция по обработке массива (или массивов) должна быть записана в отдельной функции! На входе функции (для большинства операций) задаётся исходный массив (в виде указателя), а также другие данные и параметры, необходимые для её работы. На выходе функция должна возвращать полученный новый массив.

4. Объявление функции желательно поместить в отдельный заголовочный файл, который подключается к основному файлу программы. А текст функции (как и текст функции main()) должен быть записан в основном файле программы.

5. Ввод элементов массива в память программы нужно произвести как с клавиатуры, так и из файла, а ввод необходимых для выполнения операции данных и параметров - с клавиатуры.

6. Полученный новый массив необходимо записать в новый файл в следующем формате:

- вначале там должен быть записан исходный массив (или массивы);
- потом должно быть записано название совершённой операции;
- потом должны быть записаны введённые данные и параметры;
- после чего уже должен быть записаны полученный новый массив.

7. При разработке программы необходимо учитывать и корректно обрабатывать исключительные ситуации, которые могут возникнуть при открытии на чтение или запись несуществующего файла.

8. При этом, необходимо вывести на консоль весь ход выполнения программы (комментарии по действиям программы, копии вводимых и выводимых данных).

9. Для упрощения проектирования программы рекомендуется представить проводимые алгоритмические операции в графическом виде.

Лабораторная работа №3: Списки. Функции, выполняющие создание, сохранение и восстановление списка

Цель работы

Получение практических навыков разработки программы, использующей динамическую структуру данных - линейный однонаправленный список. Разработка собственной структуры данных для представления списка в памяти компьютера. Разработка пользовательских функций для выполнения операций создания, сохранения и восстановления списка.

Задание

Разработать программу, осуществляющую операции создания, сохранения и восстановления линейного однонаправленного списка по своему заданию.

Каждый элемент списка должен представлять собой структуру, состоящую из полей, заданных в своём варианте, а также указателей на соседний (следующий) элемент списка.

Список должен быть создан в динамической памяти программы с помощью указателя. Доступ к отдельным элементам списка также должен осуществляться через указатели.

В программе должно быть предусмотрено меню для выбора тех или иных операций над списком. Каждая операция над списком должна быть выполнена в отдельной пользовательской функции.

В программе должны быть предусмотрены следующие операции над списком:

- создание нового списка;
- ввод данных списка пользователем с клавиатуры;
- сохранение данных списка во внешнем файле;
- чтение данных списка (его восстановление) из внешнего файла.

Варианты заданий

1. Список книг, имеющихся в библиотеке (номер ISBN, авторы, название, год выпуска, количество экземпляров и т.п.).
2. Список читателей библиотеки (Ф.И.О., номер читательского билета и т.п.).
3. Список журналов, имеющихся в библиотеке (номер ISSN, название, год выпуска, номер выпуска, количество экземпляров и т.п.).

4. Список подписок на газеты и журналы (подписной индекс, название, тираж, сроки подписки, периодичность выхода, цена, льготы, издательство и т.п.).
5. Список врачей поликлиники (Ф.И.О., специальность, график приёма пациентов и т.п.).
6. Список студентов университета (факультет, кафедра, № группы, № студ. Билета, Ф.И.О., дата рождения, телефон, e-mail).
7. Список курсов повышения квалификации (направление переподготовки, наименование курса, виды занятий, Ф.И.О. преподавателя и т.п.).
8. Список вакансий фирмы (название, сфера деятельности, контактные данные, наименование должности, график работы, оклад, требования по образованию и квалификации и т.п.).
9. Список грузовиков транспортной компании (марка, грузоподъёмность, максимальная дальность перевозки, плановый пробег в пути за сутки и т.п.);
10. Список объектов продаваемой недвижимости (район, площадь квартиры, количество комнат, этажность, цена, адрес и т.п.).
11. Список картин на выставке (автор, название, стиль (художественная школа), текущее состояние).
12. Список договоров о страховании физических лиц страховой компании (фирма, Ф.И.О. застрахованного лица, вид страхования, дата заключения, срок, страховая сумма и т.п.).

Указания по выполнению

Хранение данных программы необходимо осуществить в форме линейного однонаправленного списка. Каждый элемент списка должен представлять собой структуру, состоящую из полей, заданных в своём варианте, а также указателей на соседний (следующий) элемент списка.

Для организации линейного списка запрещается использовать готовые контейнерные типы данных. Всю структуру и связи линейного списка нужно прописать в программе вручную.

Список должен быть создан в динамической памяти программы с помощью указателя. Доступ к отдельным элементам списка также должен осуществляться через указатели.

При создании нового списка ввод его элементов должен осуществляться с клавиатуры. Должно быть предусмотрено сохранение списка во внешний файл, а также последующее его чтение из внешнего файла.

При разработке программы необходимо учитывать и корректно обрабатывать исключительные ситуации, которые могут возникнуть при открытии на чтение или запись несуществующего файла, а также на возможные некорректные значения некоторых важных данных.

В программе должно быть предусмотрено меню для выбора тех или иных операций над списком. Каждая операция над списком должна быть выполнена в отдельной пользовательской функции.

Для упрощения проектирования программы рекомендуется представить проводимые алгоритмические операции в графическом виде.

Лабораторная работа №4: Списки. Функции, выполняющие обработку списка

Цель работы

Получение практических навыков разработки программы, использующей линейный однонаправленный список. Разработка собственной структуры данных для представления списка в памяти компьютера. Разработка пользовательских функций для выполнения операций вставки, редактирования, поиска, сортировки и удаления его элементов.

Задание

Разработать программу, осуществляющую операции вставки, редактирования, поиска, сортировки и удаления элементов линейного однонаправленного списка по своему заданию.

Каждый элемент списка должен представлять собой структуру, состоящую из полей, заданных в своём варианте, а также указателей на соседний (следующий) элемент списка.

Список должен быть создан в динамической памяти программы с помощью указателя. Доступ к отдельным элементам списка также должен осуществляться через указатели.

В программе должно быть предусмотрено меню для выбора тех или иных операций над списком. Каждая операция над списком должна быть выполнена в отдельной пользовательской функции.

В программе должны быть предусмотрены следующие операции над списком:

- вставка нового элемента списка;
- поиск нужного элемента списка;
- редактирование данных выбранного элемента списка;
- сортировка элементов списка по заданному критерию;
- удаление выбранного элемента списка.

Варианты заданий

1. Список книг, имеющихся в библиотеке (номер ISBN, авторы, название, год выпуска, количество экземпляров и т.п.).
2. Список читателей библиотеки (Ф.И.О., номер читательского билета и т.п.).

3. Список журналов, имеющихся в библиотеке (номер ISSN, название, год выпуска, номер выпуска, количество экземпляров и т.п.).
4. Список подписок на газеты и журналы (подписной индекс, название, тираж, сроки подписки, периодичность выхода, цена, льготы, издательство и т.п.).
5. Список врачей поликлиники (Ф.И.О., специальность, график приёма пациентов и т.п.).
6. Список студентов университета (факультет, кафедра, № группы, № студ. Билета, Ф.И.О., дата рождения, телефон, e-mail).
7. Список курсов повышения квалификации (направление переподготовки, наименование курса, виды занятий, Ф.И.О. преподавателя и т.п.).
8. Список вакансий фирмы (название, сфера деятельности, контактные данные, наименование должности, график работы, оклад, требования по образованию и квалификации и т.п.).
9. Список грузовиков транспортной компании (марка, грузоподъёмность, максимальная дальность перевозки, плановый пробег в пути за сутки и т.п.);
10. Список объектов продаваемой недвижимости (район, площадь квартиры, количество комнат, этажность, цена, адрес и т.п.).
11. Список картин на выставке (автор, название, стиль (художественная школа), текущее состояние).
12. Список договоров о страховании физических лиц страховой компании (фирма, Ф.И.О. застрахованного лица, вид страхования, дата заключения, срок, страховая сумма и т.п.).

Указания по выполнению

Хранение данных программы необходимо осуществить в форме линейного однонаправленного списка. Каждый элемент списка должен представлять собой структуру, состоящую из полей, заданных в своём варианте, а также указателей на соседний (следующий) элемент списка.

Для организации линейного списка запрещается использовать готовые контейнерные типы данных. Всю структуру и связи линейного списка нужно прописать в программе вручную.

Список должен быть создан в динамической памяти программы с помощью указателя. Доступ к отдельным элементам списка также должен осуществляться через указатели.

Должно быть предусмотрено сохранение списка во внешний файл, а также последующее его чтение из внешнего файла.

При разработке программы необходимо учитывать и корректно обрабатывать исключительные ситуации, которые могут возникнуть при открытии на чтение или запись несуществующего файла, а также на возможные некорректные значения некоторых важных данных.

В программе должно быть предусмотрено меню для выбора тех или иных операций над списком. Каждая операция над списком должна быть выполнена в отдельной пользовательской функции.

Для упрощения проектирования программы рекомендуется представить проводимые алгоритмические операции в графическом виде.

Курсовая работа: Разработка электронной картотеки

Цель работы

Приобретение навыков разработки и отладки многомодульных программ на языке C++. Разработка программы, позволяющей создать, обработать, вывести и удалить базу данных на основе линейных списков.

Задание

Разработать программу, позволяющую выполнять различные операции над базой данных, представленной в виде линейного списка (тема базы данных и набор операций есть в своём варианте задания).

Курсовая работа «собирается» студентом из функций, объединённых с помощью меню в головной программе, выполняющей обработку связанного линейного списка.

В виде отдельных пользовательских функций оформляются части программы, реализующие операции:

- создание списка (выделение памяти, создание и заполнение вводимыми с клавиатуры данными элементов списка);
- сохранение введённой информации в заданном пользователем файле;
- восстановление списка (заполнение его информацией, считываемой из файла, поиск элемента по признаку (признак - одно из полей структуры));
- сортировка найденных элементов и вывод информации о них на экран;
- корректировка полей записи выбранного элемента (идентификация элемента по номеру в выводимом на экран перечне (по номеру указателя на элемент));
- удаление выбранного элемента (одного из найденных по признаку);
- вставка нового элемента (после/перед выбранным).

Все задания содержат индивидуальные требования к выполнению.

Курсовая работа оформляется в виде пояснительной записки, в которой отражены все полученные результаты разработки.

Варианты заданий

Задание №1. Организовать информационную систему библиотеки.

Должны храниться сведения о:

- книгах (автор, название, количество экземпляров);
- читателях (ФИО, номер читательского билета);
- факте выдачи экземпляра книги конкретному читателю (дата выдачи).

Факт выдачи и возврата экземпляра книги фиксируются.

Необходимо обеспечить эффективные по времени:

- выдачу справки о наличии книг (конкретного автора, по названию);
- выдачу справки о взятых книгах определенным читателем;
- формирование списка должников (с выдачи прошло более месяца);
- формирование списка книг, которые были выданы в определённом месяце или в определённый день.

Задание №2. Организовать каталог книг, хранящихся в библиотеке, а также тех, которые могут быть получены по запросу из других хранилищ. Необходимо обеспечить эффективную обработку требования читателя по его запросу (наличие конкретной книги, книг определенного автора, подходящих по названию и т.д.).

Задание №3. Создать каталог журналов, имеющихся в библиотеке. Необходимо обеспечить эффективную обработку требования читателя по его запросу (наличие конкретного журнала по названию и номеру, по дате, статей по названиям, статей определённого автора и т.д.).

Задание №4. Подготовить справочник для подписки на издания.

Для проведения подписки имеется информация о газетах и журналах (индекс, название, тираж, сроки подписки, периодичность выхода, цена, льготы, издательство и т.п.), а также об издающих их издательствах (наименование, адрес и т.п.).

Необходимо обеспечить эффективную выдачу сведений:

- упорядоченных по цене, тиражу, названию газет и/или журналов;
- о газетах и журналах, выпускаемых определённым издательством;
- об определённой газете или журнале;
- на какие газеты и/или журналы предоставляется льготная подписка.

Задание №5. В поликлинике работают врачи (ФИО) различных специальностей, имеющие разную квалификацию. Каждый врач осуществляет приём по своему заданному графику приёма пациентов. Каждый пациент (ФИО) может обращаться в поликлинику несколько раз к разным специалистам. Необходимо организовать запись пациентов на приём к врачам поликлиники по дням недели, каждый пациент в один день может

быть записан не более чем к D врачам, а также отказаться от имеющейся записи.

При выполнении записи необходимо иметь информацию о:

- занятости конкретного врача в течение недели;
- занятости врачей конкретной специальности;
- когда и куда пациент уже записан.

Задание №6. Обеспечить эффективное хранение данных и быстрый поиск информации в системе учета курсов повышения квалификации.

Должна быть представлена информация о:

- направлениях переподготовки (наименование курса, виды занятий - лекции, практические занятия, лабораторные работы и т.п.);
- списках номеров групп, сформированных для каждого курса;
- списочном составе преподавателей (ФИО, на каких курсах может преподавать, виды проводимых занятий на этих курсах);
- закреплении групп по видам занятий за конкретными преподавателями.

Необходимо обеспечить быстрое получение данных о:

- всех преподавателях, которые обеспечивают занятия в опредеённых группах;
- всех группах, с которыми работает определённый преподаватель с указанием видов проводимых им занятий;
- всех группах, для которых должен быть обеспечен определённый вид занятий;
- преподавателях, которые могут быть привлечены на замену по определённому виду занятий в рамках конкретного направления переподготовки.

Задание №7. Организовать быстрый доступ к спискам картин, объединённых в группы по разным информационным признакам. Информация о картинах должна учитывать специфику письма при создании, сходство сюжетных линий (персонажей), состояние, которое определяет необходимость реставрационных работ (косметических или большего объема). Реализовать операции по занесению новых картин в списки, фиксации изменений в их состоянии, отправки картин, имеющих плохое состояние, на реставрацию (всех или определённых художественных школ).

Задание №8. Организовать информационную систему выставки попугаев.

Должны храниться сведения о породе, окрасе, стране происхождения, возрасте, хозяине, особенностях (говорит, поёт, ...) и т.п.

Необходимо быстро сгруппировать попугаев по какому-то признаку (например, окрасу или стране происхождения), проведя, при этом, внутри группы сортировку по другому признаку (например, возрасту).

Задание №9. Обеспечить эффективное хранение данных и быстрый поиск информации в бюро по трудоустройству.

Должна быть представлена информация о:

- работодателях (название, сфера деятельности, адрес, телефон и т.п.);
- предлагаемых ими вакансиях (наименование должности, график работы, оклад, требования по образованию и квалификации, предложения и заявки работодателей и т.д.);
- соискателях (фамилия, имя, отчество, возможные должности, сфера деятельности, стаж работы, ожидаемый оклад и т.д.).

Соискатель должен получать список подходящих по его желаниям предложений от работодателей, работодатель должен получать список всех подходящих под его требования работников. Должны добавляться новые соискатели, работодатели, вакансии. В случае совпадения интересов работодателя и соискателя вакансия должна быть занята и перемещена в список удовлетворённых заявок.

Задание №10. Обеспечить работу компании, осуществляющей грузовые перевозки на основе наличия:

- списка парка грузовиков (марка, грузоподъёмность, максимальная дальность перевозки, плановый пробег в пути за сутки);
- списка водителей (ФИО, разрешение на использование марки грузовика);
- списка маршрутов перевозки (конечный пункт, дальность, время погрузки/разгрузки в конечных пунктах, количество водителей).

При поступлении очередного заказа (маршрут, дата выезда, масса груза, пожелание по марке грузовика) необходимо сформировать для поездки комбинацию грузовик-водитель(-и).

Дополнительно необходимо выдавать информацию:

- о свободных водителях на определённую дату;
- о свободных грузовиках на определённую дату;

- о грузовиках, находящихся на определённом маршруте;
- о водителях, находящихся на определённом маршруте;
- о плановой дате прибытия грузовика с водителем(-ями).

Задание №11. Подготовить справочник по продаже недвижимости.

Имеется информация о характеристиках продаваемой недвижимости (например, район, площадь квартиры, количество комнат, этажность, цена, адрес и т.п.) и заявках на покупку недвижимости с аналогичными характеристиками, при этом в заявках могут присутствовать списки желаемых вариантов.

Необходимо обеспечить эффективную выдачу сведений:

- о подходящих по площадям квартирах (в определённом районе или по выбору районов);
- о подходящих по цене квартирах (в определённом районе или по выбору районов) с учётом дополнительных условий (например, этаж, площадь и др.);
- справку по определённому количеству комнат в квартире;
- варианты встречных покупок/продаж.

Задание №12. Организовать информационную систему страховой компании по договорам страхования частных лиц.

Должны храниться сведения о:

- филиалах кампании (название, местонахождение);
- страховых агентах в каждом филиале (ФИО);
- застрахованных лицах (ФИО);
- договорах страхования (дата заключения, срок, страховая сумма, филиал, агент) по виду страхования (например, страхование автотранспорта от угона, страхование домашнего имущества, добровольное медицинское страхование и т.п.).

Факт заключения нового договора страхования фиксируется.

Необходимо обеспечить эффективную выдачу сведений:

- о результатах работы определённого страхового агента;
- о результатах работы определённого филиала;
- о договорах страхования определённого частного лица;
- о договорах страхования определённого вида (по филиалам и компании в целом).

Задание №13. Обеспечить эффективное хранение данных и быстрый поиск информации об элементах жизнеобеспечения в помещениях здания. Имеется список помещений и коридоров (пронумерованы) по этажам здания. Для каждого помещения или коридора имеется список размещённых в нём элементов жизнеобеспечения (наименование, количество, место установки - например, пол, потолок, стена) с указанием их типов: кондиционеры, электророзетки, светильники, водопроводные краны и т.п.

Осуществить быстрый поиск следующей информации:

- месторасположение элементов жизнеобеспечения определённого типа;
- все элементы жизнеобеспечения на определённом этаже по заданному месту установки;
- месторасположение элементов жизнеобеспечения определённого типа для заданного этажа;
- группировка элементов жизнеобеспечения в коридорах по этажам здания.

Задание №14. Организовать информационную систему клуба туристов.

Должны храниться сведения о:

- доступных пунктах (место посещения - наименования местности, количество дней пребывания, форма передвижения до пункта - например, пешком или на определённых видах транспорта, и т.д.);
- туристических маршрутах (совокупность пунктов, расставленных в определённом порядке посещения).

Необходимо обеспечить эффективные:

- выбор туристического маршрута по определённым критериям (например, список желаемых для посещения пунктов, использование определённых вариантов передвижения и т.п.);
- формирование желаемых туристических маршрутов (помимо списка уже имеющихся) с проверкой их соответствия корректности задания и сведений о доступных пунктах.

Задание №15. Организовать справочник по географическим объектам.

Варианты географических объектов: страны и города мира (их расположение по континентам, столицы, название валюты, численность населения, данные о крупнейших городах, числе жителей и т.п.).

Необходимо обеспечить эффективную выдачу сведений:

- группировка по определённой стране/континенту;
- об определённом географическом объекте;

- суммарная количественная характеристика по стране/континенту.

Задание №16. Организовать справочник по географическим объектам.

Варианты географических объектов: водоёмы - моря, озера, реки (местонахождение по странам, название, количественные характеристики - протяжённость, глубина, бассейн и т.п.).

Необходимо обеспечить эффективную выдачу сведений:

- группировка по определённой стране/континенту;
- об определённом географическом объекте;
- суммарная количественная характеристика по стране/континенту.

Задание №17. Организовать справочник по географическим объектам.

Варианты географических объектов: горные вершины (горная система, название, количественные характеристики - высота, местонахождение по странам).

Необходимо обеспечить эффективную выдачу сведений:

- группировка по определённой стране;
- об определённом географическом объекте;
- суммарная количественная характеристика по стране.

Задание №18. Организовать быстрый поиск запрашиваемой информации в справочной службе. Необходимо обеспечить эффективное (для обслуживания запросов) представление данных. Возможно поступление новых записей, обновление их отдельных полей или исключение из хранения.

Указания по выполнению

Хранение данных программы (базы данных, картотеки, справочника, информационной системы и т.п.) необходимо осуществить в форме линейного списка. Для организации линейного списка запрещается использовать готовые контейнерные типы данных. Всю структуру и связи линейного списка нужно прописать в программе вручную.

Ввод исходных данных осуществляется через внешний файл. Вывод полученных новых данных должно осуществляться в новый файл. Ввод других необходимых данных для выполнения программы может осуществляться как через отдельный файл, так и напрямую с консоли. При этом, необходимо вывести на консоль весь ход выполнения программы

(комментарии по действиям программы, копии вводимых и выводимых данных).

При разработке программы необходимо учитывать и корректно обрабатывать исключительные ситуации, которые могут возникнуть при открытии на чтение или запись несуществующего файла, а также на возможные некорректные значения некоторых важных данных.

Для выполнения некоторых заданий может понадобиться осуществить доступ к отдельному элементу списка. В этом случае доступ должен осуществляться с помощью указателей.

Каждая операция из меню программы должна осуществляться в отдельных пользовательских функциях.

Для упрощения проектирования программы рекомендуется представить проводимые алгоритмические операции в графическом виде.

УКАЗАНИЯ ПО ОФОРМЛЕНИЮ ОТЧЁТОВ ПО ЛАБОРАТОРНЫМ РАБОТАМ

Титульный лист

Титульный лист оформляется согласно шаблону отчётов по лабораторным работам, принятому в СПбГЭТУ «ЛЭТИ»:

https://etu.ru/assets/files/3004_1_ShABLON-lab.r..doc

Сверху название кафедры пишется сокращённо.

Темы лабораторных работ должны соответствовать названиям из данных методических указаний.

Цель работы

Формулировку цели работы можно взять с соответствующего раздела описания каждой лабораторной работы в данных методических указаниях.

Формулировка задания

В этом разделе отчёта нужно будет повторить текст общей части задания из данных методических указаний, а также добавить формулировку своего варианта задания.

Форматы входных и выходных файлов

Данный раздел присутствует только в отчётах лабораторных работ, в которых используются внешние файлы для ввода и вывода данных (т.е. начиная с лабораторной работы №4 второго семестра)!

В этом разделе отчёта нужно будет описать содержание входных и выходных файлов, используемых в программе. При описании нужно будет указать, из каких частей (компонент) состоит каждый файл и где в нём будут располагаться требуемые данные.

Описание структур данных

Данный раздел присутствует только в отчётах второго и третьего семестров!

В этом разделе отчёта нужно будет привести графические рисунки, описывающие используемые структуры данных для всех значимых переменных программы.

Описание пользовательских функций

Данный раздел присутствует только в отчётах лабораторных работ, в которых используются пользовательские функции и методы (т.е. начиная с лабораторной работы №2 второго семестра)!

В этом разделе отчёта нужно будет привести краткое описание разработанных пользовательских функций и методов, включая:

- имя функции/метода;
- тип возвращаемого значения;
- список аргументов (с указанием их назначения и типов данных).

Алгоритм программы

В этом разделе отчёта нужно будет привести описание алгоритма (или, хотя бы, его основной его части или же идеи) не псевдокоде, или же своими словами.

Если в программе используются пользовательские функции или методы, то описание алгоритма приводится по каждой из них в отдельности.

В отчётах только первого семестра нужно будет также нарисовать блок-схему программы. Для удобства, её можно поместить в приложение 2.

Рисовать блок-схемы можно, например, в одной из следующих программ:

- Microsoft Word;
- Microsoft Office Visio;
- DIA.

На всякий случай, ниже приводятся ссылки на сайты, откуда можно скачать дистрибутив программы DIA:

https://wiki.gnome.org/Apps/Dia

http://dia-installer.de/

https://sourceforge.net/projects/dia-installer/files/dia-win32-installer/0.97.2/dia-setup-0.97.2-2-unsigned.exe/download

Тестирование программы

В этом разделе отчёта нужно будет привести примеры вводимых данных в программу: 1-2 примера на правильные данные (т.е. при которых программа работает правильно) + (желательно) 1 пример на неправильные данные.

По каждому примеру:

- вначале нужно выписать в тексте те данные, что сейчас будут вводиться в программу;
- выписать ожидаемый результат (ответ программы);
- после чего привести скриншоты работы программы по шагам, последовательно вводя нужные данные и получая результат;

- если вывод данных идёт не на экран, а в файл - то после скриншотов программы нужно привести скриншот (или же копию текста) этого файла.

Выводы

Своими словами постарайтесь написать, что Вы изучили в этой лабораторной работе и что сделали в итоге. При написании, опирайтесь на формулировки цели работы и существенную часть задания.

Приложение 1. Листинг программного кода

В данном приложении приводится полный текст (листинг) программы.

Если программа состоит из нескольких исходных файлов, то нужно привести листинг кода каждого файла! При этом, все тексты файлов помещаются в один раздел (приложение), друг за другом, отделяются между собой несколькими пустыми строками, а перед текстом каждого файла пишется его имя (жирным шрифтом).

Приложение 2. Блок-схема программы

Данное приложение есть только в отчётах первого семестра!

В него помещается блок-схема программы, или же набор блок-схем её отдельных модулей (функций) - в случае, если эти блок-схемы не поместились в соответствующем разделе основной части отчёта.

УКАЗАНИЯ ПО ОФОРМЛЕНИЮ ПОЯСНИТЕЛЬНЫХ ЗАПИСОК К КУРСОВЫМ РАБОТАМ

Состав страниц пояснительной записки к курсовой работе и её оформление должно соответствовать принятому в СПбГЭТУ «ЛЭТИ» шаблону оформления курсовых работ (и курсовых проектов):

https://etu.ru/assets/files/3004_3_ShABLON-kursovika.doc.

В содержании пояснительной записки обязательно должны присутствовать следующие страницы и разделы:

Титульный лист

На титульном листе должны быть имена и подписи всех студентов бригады, а также должна быть указана общая для всех бригад тема курсовой работы (её пишите так, как она записана в данных методических указаниях).

Сверху название кафедры пишется сокращённо.

Задание на курсовую работу

В пункт *«Исходные данные»* помещается копия текста общей части задания на курсовую работу из данных методических указаний с добавлением формулировки своего варианта.

Содержание пояснительной записки: *«Содержание», «Введение», Основные главы, «Заключение», «Список использованных источников»*.

Предполагаемый объём пояснительной записки - *не менее 30 страниц*.

Аннотация

Здесь своими словами нужно кратко описать выполняемую работу на русском и английском языках.

Содержание

Здесь вместо приведённой в шаблоне таблицы нужно сделать автоматическое оглавление. Для этого названия каждой главы и её подразделов должны быть оформлены в соответствующем стиле типа *«Заголовок 1»*, *«Заголовок 2»* и т.д.

Введение

В этот раздел можно поместить копию цели и задания на курсовую работу, а также указаний по её выполнению.

1. Внешние форматы хранения данных

Данный раздел присутствует только в пояснительных записках к курсовым работам второго и третьего семестров!

В этом разделе пояснительной записки нужно будет описать содержание входных и выходных файлов, используемых в программе. При описании нужно будет указать, из каких частей (компонент) состоит каждый файл и где в нём будут располагаться требуемые данные.

2. Внутренние форматы хранения данных

Данный раздел присутствует только в пояснительных записках к курсовым работам второго и третьего семестров!

В этом разделе пояснительной записки нужно будет привести графические рисунки, описывающие используемые структуры данных для всех значимых переменных программы.

3. Описание пользовательских функций и модулей программы

Данный раздел присутствует только в пояснительных записках к курсовым работам второго и третьего семестров!

В этом разделе отчёта нужно будет привести краткое описание разработанных модулей программы, а также пользовательских функций и методов в следующем формате (табл. 1):

Табл. 2. Описание модулей, структур/классов, функций/методов программы

Имя модуля	**Имя структуры/класса или функции**	**Назначение**	**Параметры для функции**	**Возвращаемое функцией значение**

4. Описание интерфейса пользователя

Здесь нужно привести фразы, которые выводятся на экране программы: какие данные программа просит пользователя ввести и в каком порядке, что происходит, если пользователь выберет тот или иной пункт меню и т.п.

5. Описание алгоритма работы программы

Здесь нужно на псевдокоде или же своими словами описать основные шаги и операции, что выполняет программа.

В пояснительных записках к курсовой работе только первого семестра также нужно будет нарисовать блок-схему программы. Для удобства, её можно поместить в приложение 2.

6. Примеры работы программы

По каждой реализованной в меню программы операции нужно привести хотя бы один пример выполнения. По каждому примеру нужно вначале выписать в тексте те данные, что сейчас будут вводиться в программу, потом выписать ожидаемый результат (ответ программы), после чего привести скриншоты работы программы по шагам, последовательно вводя нужные данные и получая результат.

Заключение

Своими словами постарайтесь кратко подвести итоги выполненной работы: написать, что вы изучили в этой работе и описать полученный результат. При написании, опирайтесь на формулировки цели работы и существенную часть задания.

Список использованных источников

В качестве списка литературы можно взять, например, источники, указанные в конце данных методических указаний.

Приложение 1. Листинг программного кода

Здесь приводится полный текст (листинг) программы.

Если программа состоит из нескольких исходных файлов, то нужно привести листинг кода каждого файла. При этом, все тексты файлов помещаются в один раздел (приложение), друг за другом, отделяются между собой несколькими пустыми строками, а перед текстом каждого файла пишется его имя (жирным шрифтом).

Приложение 2. Блок-схема программы

Данное приложение есть только в пояснительных записках к курсовым работам первого семестра!

В данное приложение помещается блок-схема программы - в случае, если она не поместилась в соответствующем разделе основной части пояснительной записки.

Рисовать блок-схемы можно, например, в одной из следующих программ:

- Microsoft Word;
- Microsoft Office Visio;
- DIA.

На всякий случай, ниже приводятся ссылки на сайты, откуда можно скачать дистрибутив программы DIA:

https://wiki.gnome.org/Apps/Dia

http://dia-installer.de/

https://sourceforge.net/projects/dia-installer/files/dia-win32-installer/0.97.2/dia-setup-0.97.2-2-unsigned.exe/download

ЗАКЛЮЧЕНИЕ

В ходе выполнения цикла лабораторных работ по дисциплине «Программирование» студенты очно-заочной формы закрепляют знания основных типов данных, структур данных и программных конструкций языка C++, получают навыки разработки программ в интегрированной среде разработки (IDE).

В ходе выполнения курсовых работ студенты получают навыки разработки комплексных программ по обработке числовой, текстовой и списочной информации на основе изученных ими ранее структур данных, программных конструкций и алгоритмов.

Данная дисциплина закладывает основы для последующего изучения студентами дисциплин «Алгоритмы и структуры данных», «Объектно-ориентированное программирование» и «Базы данных» направления «Информатика и вычислительная техника».

СПИСОК ЛИТЕРАТУРЫ

1. Страуструп Б. Язык программирования C++. Специальное издание. - М.: Бином; СПб.: Невский диалект, 2008. - 1104 с.
2. Дейтел Х.М., Дейтел П.Дж. Как программировать на C++. 5-е издание. Перевод с англ. - М.: ООО «Бином-Пресс», 2008. - 1456 с.
3. Хигай А.Г., Зуев И.С., Грушвицкий Р.И. Программирование на языке С в среде Borland 3.1: Методические указания к лабораторным работам по дисциплинам «Информатика», «Программирование». - СПб.: Издательство СПбГЭТУ «ЛЭТИ», 2006.
4. Калмычков В.А., Чугунов Л.А. Представление и обработка математических данных на языке C++: Учебное пособие. - СПб.: Издательтство СПбГЭТУ «ЛЭТИ», 2010.
5. Ивановский С.А., Калмычкова В.А., Лисс А.А. Разработка корректных программ: Практикум по программированию. - СПб.: Издательство СПбГЭТУ «ЛЭТИ», 2001.
6. Литвинова Л.А. Методические указания к лабораторным занятиям по дисциплине «Алгоритмизация и программирование» (раздел «Решение задач в среде Turbo-Pascal») для студентов направления 6050202 «Автоматизация и компьютерно-интегрированные технологии» дневной формы обучения. - Севастополь: Издательство СевНТУ, 2011. - 36 с.
7. Павловская Т.А. C/C++. Программирование на языке высокого уровня. СПб.: Лидер, 2010. - 461 с.
8. Павловская Т.А., Щипак Ю.А. C/C++. Структурное программирование: Практикум. - СПб.: Питер, 2004, 2005, 2007. - 240 с.

Учебно-методическое пособие

Миронов Сергей Эльмарович,
Кузьмин Сергей Алексеевич

Методические указания к проведению лабораторных и курсовых работ по дисциплине «Программирование» для студентов очно-заочной формы обучения по направлению «Информатика и вычислительная техника»

Издание публикуется в авторской редакции.

СПбГЭТУ «ЛЭТИ»
197376, Санкт-Петербург, ул. Професора Попова, д. 5.