



**Филиал «Котельники» государственного
бюджетного образовательного учреждения
высшего образования Московской области
«Университет «Дубна»**

Артамонов Ю.Н.

Методические указания

по выполнению курсовой работы по дисциплине

«Программирование на языке высокого уровня»

Для бакалавров по направлению

«Информатика и вычислительная техника»

Москва – 2018

ОГЛАВЛЕНИЕ

1	Требования по оформлению курсовой работы	4
2	Разработка численных алгоритмов	8
2.1	Суммирование рядов и вычисление элементарных функций	8
2.2	Приближенные методы нахождения корней уравнения	13
3	Разработка игровой программы	20
	Литература	25
А	Образец титульного листа	26
В	Примеры блок-схем	28

Введение

Методическое пособие предназначено для выполнения курсовой работы бакалаврами по направлению "Информатика и вычислительная техника". Курсовая работа включает в себя две части:

1. разработка численных алгоритмов;
2. разработка игровых программ.

Задания первой части курсовой работы базируются на знаниях, полученных студентами по смежным дисциплинам: «Математический анализ», «Линейная алгебра». Реализация соответствующих алгоритмов на языке программирования С позволяет глубже понять суть подходов к решению задач, дополняет полученные теоретические сведения практическими навыками по принципу: «Чтобы полностью понять решение задачи, следует научить решать ее компьютер».

Вторая часть курсовой работы связана с реализацией некоторой игровой программы и стимулирует у студента творческий подход к реализации задания.

ГЛАВА 1

ТРЕБОВАНИЯ ПО ОФОРМЛЕНИЮ КУРСОВОЙ РАБОТЫ

Пояснительная записка к курсовой работе выполняется в текстовом редакторе MS Word: формат A4, отступ слева 3 см., отступ справа 1 см., отступы сверху и снизу 2 см.; гарнитура шрифта Times New Roman; кегль 12 пунктов; интервал 1.5.

Структура пояснительной записки

Пояснительная записка проекта (работы) должна содержать:

1. Титульный лист
2. Содержание
3. Основную часть в соответствии с заданием
4. Заключение (содержит краткие выводы по результатам выполненной работы и рекомендации по её использованию)
5. Список использованных источников

6. Приложения

Титульный лист оформляется в соответствии с приложением А.

Содержание оформляется, используя стандартные средства формирования оглавления в MS Word.

Основная часть оформляется в соответствии с вариантом и состоит из двух частей: 1. Разработка численных алгоритмов, 2. Разработка игровой программы. При этом первая часть разбивается на два раздела: 1.1 Суммирование рядов и вычисление элементарных функций, 1.2 Приближенные методы нахождения корней уравнения.

Все формулы, графики, таблицы следует нумеровать по частям и разделам. Например, (1.1.1) - первая формула из первой части первого раздела; Таблица 2.1 - первая таблица из второго раздела; Рисунок 1.2.3 - третий рисунок первой части второго раздела. Все рисунки и таблицы должны иметь название. Например, Таблица 1.1.2 - Результаты расчета программы.

При описании разработки программ первой части в пояснительной записке необходимо следовать плану:

1. Пояснение математической задачи с выделением пунктов:
1. Дано; 2. Найти, 3. Решение.
2. Описание входных данных для задачи: тип входных данных, ограничения, обработка ошибочного ввода, тестовые наборы входных данных.
3. Описание выходных данных для задачи: тип выходных данных, верификация выходных данных с использованием wolfram (<http://www.wolframalpha.com/>).
4. Блок-схема реализуемого алгоритма (примеры составления блок-схем приведены в приложении В).
5. Листинг программы на языке С. **Обратите внимание!**
Каждая строка кода первой части должна иметь

комментарий, поясняющий, что выполняется в строке. Длинные комментарии можно оформлять с помощью окружения `/* текст комментария */`.

6. Расчетные таблицы соответствия входных и выходных данных.
7. Выводы по результатам тестирования программного приложения на расчетных примерах.

При описании разработки программы второй части в пояснительной записке необходимо следовать плану:

1. Формулировка задачи с выделением требований и ограничений (минимальное описание - это текст исходной задачи в соответствии с вариантом).
2. Декомпозиция задачи на подзадачи с выделением необходимых и подлежащих дальнейшей реализации функций (подпрограмм)
3. Описание взаимодействия подпрограмм при работе основной программы в виде блок-схемы.
4. Блок-схемы всех ключевых подпрограмм с листингом их программного кода. В программный код, по аналогии с первой частью, целесообразно включать комментарии (чем больше содержательных комментариев, тем лучше).
5. Примеры работы программы: скриншоты результатов работы.
6. Выводы по результатам тестирования программного приложения на расчетных примерах.

Код всей игровой программы следует поместить в приложение.

При разработке программ, целесообразно использовать не менее 10 литературных источников. В пояснительной записке

следует указывать ссылки на литературные источники. Литературные источники указываются в квадратных скобках. Например, [1]. Разрешается указывать сразу несколько источников: [1, 4, 7], [2-4, 6]. Соответственно в разделе «Список использованных источников» необходимо указать библиографические ссылки, которые следует оформлять по ГОСТ 7.1-2003 (по аналогии с библиографическими ссылками данного методического пособия).

ГЛАВА 2

РАЗРАБОТКА ЧИСЛЕННЫХ АЛГОРИТМОВ

2.1 Суммирование рядов и вычисление элементарных функций

Задания

1. Для функции $\operatorname{tg}(x)$ имеет место представление в виде цепной дроби:

$$\operatorname{tg}(x) = \frac{1}{\frac{1}{x} - \frac{3}{\frac{1}{x} - \frac{5}{\frac{1}{x} - \frac{7}{\frac{1}{x} - \frac{1}{\dots}}}}}$$

Реализуйте функцию, вычисляющую $\operatorname{tg}(x)$ по этому представлению, и проверьте насколько быстро сходится процесс вычислений (т.е. сколько членов дроби надо взять для получения результата с заданной точностью при различных значениях x). Полученные результаты расчета оформите в виде таблицы в пояснительной записке: значение x , точность, количество членов дроби.

2. Известно два представления числа e в виде ряда и бесконечной дроби:

$$e = 2 + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \dots$$

$$e = 1 + \frac{1}{1 - \frac{1}{2 + \frac{1}{3 - \frac{1}{4 + \frac{1}{5 - \dots}}}}}$$

Сравните скорости сходимости с помощью ряда и дроби. Полученные результаты расчета оформите в виде таблицы в пояснительной записке: точность, количество членов ряда, количество членов дроби.

3. Известно три представления числа π :

$$\pi = 4 \cdot \left(1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \dots \right)$$

$$\pi = 3 + 4 \cdot \left(\frac{1}{2 \cdot 3 \cdot 4} - \frac{1}{4 \cdot 5 \cdot 6} + \frac{1}{6 \cdot 7 \cdot 8} - \dots \right)$$

$$\pi = \sqrt{6 \cdot \left(1 + \frac{1}{2^2} + \frac{1}{3^2} + \frac{1}{4^2} + \dots \right)}$$

Сравните скорость сходимости каждого представления. Полученные результаты расчета оформите в виде таблицы в пояснительной записке: точность, количество членов ряда.

4. Известно следующее представление:

$$\prod_{k=1}^{\infty} \left(1 + \frac{(-1)^k}{2k+1} \right) = \frac{\sqrt{2}}{2}$$

Определите, сколько потребуется сомножителей потребуется взять, чтобы равенство выполнялось до шестой значащей цифры. Полученные результаты расчета оформите в виде таблицы в пояснительной записке: точность, количество членов ряда.

5. Известно следующее представление:

$$\frac{\pi^2}{8} - \frac{\pi}{4}|x| = \frac{\cos(3x)}{3^2} + \frac{\cos(5x)}{5^2} + \dots + \frac{\cos((2n+1)x)}{(2n+1)^2} + \dots,$$

$$|x| < 1$$

Реализуйте вычисление по этому представлению, и проверьте насколько быстро сходится процесс вычислений (т.е. сколько слагаемых надо взять для получения результата с заданной точностью при различных значениях x). Полученные результаты расчета оформите в виде таблицы в пояснительной записке: значение x , точность, количество членов ряда.

6. Известно следующее представление:

$$\frac{1}{4} \left(x^2 - \frac{\pi^2}{3} \right) = -\cos(x) + \frac{\cos(2x)}{2^2} - \dots + (-1)^n \frac{\cos(nx)}{n^2},$$

$$\frac{\pi}{5} \leq x \leq \pi$$

Реализуйте вычисление по этому представлению, и проверьте насколько быстро сходится процесс вычислений (т.е. сколько слагаемых надо взять для получения результата с заданной точностью при различных значениях x). Полученные результаты расчета оформите в виде таблицы в пояснительной записке: значение x , точность, количество членов ряда.

7. Известно следующее представление:

$$\frac{1}{4} \ln \left(\frac{1+x}{1-x} \right) + \frac{1}{2} \operatorname{arctg}(x) = x + \frac{x^5}{5} + \dots + \frac{x^{4n+1}}{4n+1} + \dots,$$

$$-1 < x < 1$$

Реализуйте вычисление по этому представлению, и проверьте насколько быстро сходится процесс вычислений (т.е. сколько слагаемых надо взять для получения результата с заданной точностью при различных значениях x). Полученные результаты расчета оформите в виде таблицы в пояснительной записке: значение x , точность, количество членов ряда.

8. Известно следующее представление:

$$(1 + 2x^2)e^{x^2} = 1 + 3x^2 + \dots + \frac{2n+1}{n!}x^{2n} + \dots$$

Реализуйте вычисление по этому представлению, и проверьте насколько быстро сходится процесс вычислений (т.е. сколько слагаемых надо взять для получения результата с заданной точностью при различных значениях x). Полученные результаты расчета оформите в виде таблицы в пояснительной записке: значение x , точность, количество членов ряда.

9. Известно следующее замечательное соотношение, установленное С.Рамануджаном:

$$\sqrt{\frac{e \cdot \pi}{2}} = 1 + \frac{1}{1 \cdot 3} + \frac{1}{1 \cdot 3 \cdot 5} + \frac{1}{1 \cdot 3 \cdot 5 \cdot 7} + \dots +$$

$$+ \frac{1}{1 + \frac{1}{1 + \frac{2}{1 + \frac{3}{1 + \frac{4}{1 + \dots}}}}}$$

Вычислить, сколько членов ряда и цепной дроби нужно взять, чтобы достичь заданной точности. Полученные результаты расчета оформите в виде таблицы в пояснительной записке: значение x , точность, количество членов ряда, количество членов дроби.

10. Известно представление $\cos(x)$ в виде произведения:

$$\cos(x) = \left(1 - \frac{4x^2}{\pi^2}\right) \left(1 - \frac{4x^2}{9\pi^2}\right) \dots \left(1 - \frac{4x^2}{(2n-1)^2\pi^2}\right)$$

Реализуйте функцию, вычисляющую $\cos(x)$ по этому представлению, и проверьте насколько быстро сходится процесс вычислений (т.е. сколько множителей надо взять для получения результата с заданной точностью при различных значениях x). Полученные результаты расчета оформите в виде таблицы в пояснительной записке: значение x , точность, количество множителей.

Распределение заданий по вариантам

Номер варианта:	1	2	3	4	5	6	7	8	9	10	11
Номер задания:	5	9	1	2	7	8	7	3	4	6	1

Номер варианта:	12	13	14	15	16	17	18	19	20
Номер задания:	2	3	4	5	6	7	8	9	10

Номер варианта:	21	22	23	24	25	26	27	28	29
Номер задания:	9	3	4	5	1	7	8	2	10

2.2 Приближенные методы нахождения корней уравнения

Ниже рассматриваются численные методы решения уравнений вида $f(x) = 0$.

Метод деления отрезка пополам (номер метода 1). Предположим, что на отрезке $[a, b]$ в точке $x_0 \in [a, b]$ график функции $f(x)$ пересекает ось абсцисс, то есть имеет место соотношение $f(x_0) = 0$ (см. рисунок 2.1). Идея метода состоит в том, чтобы последовательно сдвигать левую или правую границу отрезка $[a, b]$ в точку $c = \frac{a+b}{2}$ в зависимости от знаков функции $f(x)$ на концах отрезка. Например, для представленного рисунка 2.1: если $f(c) < 0$, то сдвигаем левую границу, то есть получаем новый отрезок $[c, b]$, если $f(c) > 0$, то сдвигаем правую границу, то есть получаем отрезок $[a, c]$. В итоге будет сохраняться условие $x_0 \in [c, b]$ или $x_0 \in [a, c]$. Процесс следует завершить, когда получится отрезок длины меньше заданной ε . Следует учесть,

что в реальном вычислительном процессе при малой величине ε может оказаться, что при вычислении по формуле $c = \frac{a+b}{2}$ точное значение c округлится до ближайшего a или b . В результате процедура вычисления корня заикнется.

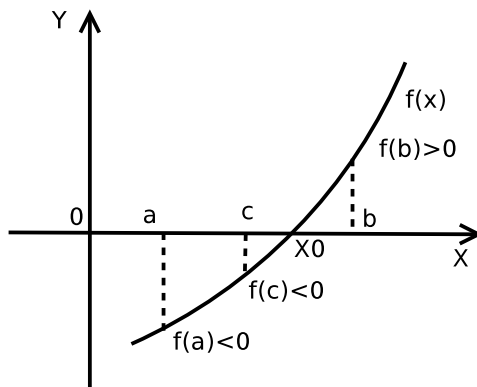


Рис. 2.1: Метод деления отрезка пополам

Программа должна предусматривать возможность подобной ситуации, а также анализировать некорректные входные данные ($f(a)$ и $f(b)$ имеют один знак).

Метод касательных (номер метода 2). Идея метода продемонстрирована на рисунке 2.2. Произвольно выбирается некоторая точка x_n и проводится касательная к функции $f(x)$ в этой точке. Пусть касательная пересечет ось абсцисс в точке x_{n+1} , тогда проводится касательная к функции $f(x)$ уже в точке x_{n+1} . Данный процесс продолжают до достижения заданной точности.

Чтобы найти касательную, используется понятие производной. Известно, что производная в точке x_n равна тангенсу угла наклона касательной к функции $f(x)$ в этой точке. Тогда из ри-

сунка 2.2 имеем:

$$f'(x_n) = \frac{f(x_n) - 0}{x_n - x_{n+1}} \Rightarrow x_{n+1} = x_n - \frac{f'(x_n)}{f(x_n)}$$

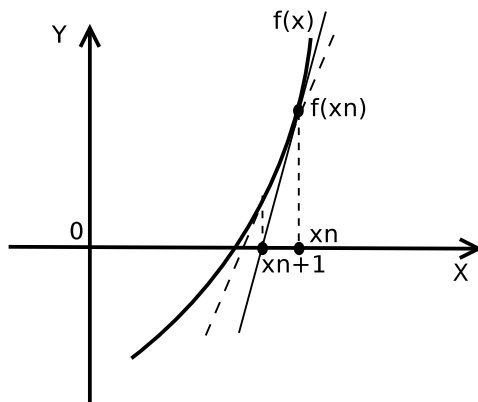


Рис. 2.2: Метод касательных

В качестве предварительного критерия окончания вычислений можно взять неравенство $|x_n - x_{n+1}| < \varepsilon$. Однако, данное неравенство не гарантирует требуемой точности приближения x_{n+1} . Поэтому при выполнении указанного неравенства следует сделать дополнительную проверку знаков функции $f(x)$ на краях отрезка $[x_{n+1} - \varepsilon, x_{n+1} + \varepsilon]$. Если эта проверка даст одинаковые знаки на концах отрезка, то следует продолжить вычисление приближений, взяв меньшее ε . Если точность не достигается за значительное число итераций (например, 200), то вычисления целесообразно прервать и вернуть признак ошибки (например, -1).

Метод секущих (номер метода 3). Данный метод можно назвать модификацией от метода касательных (вместо производной берутся ее приближения в виде конечных разностей).

Идея метода продемонстрирована на рисунке 2.3. Вначале выбираются два начальных приближения x_n и x_{n-1} , находятся значения функции $f(x)$ в этих точках. Затем через две точки с координатами $(x_{n-1}, f(x_{n-1}))$ и $(x_n, f(x_n))$ проводится прямая, которая пересечёт ось абсцисс в точке x_{n+1} . Тогда точка x_{n+1} становится следующим приближением $f(x_{n+1}) \approx 0$.

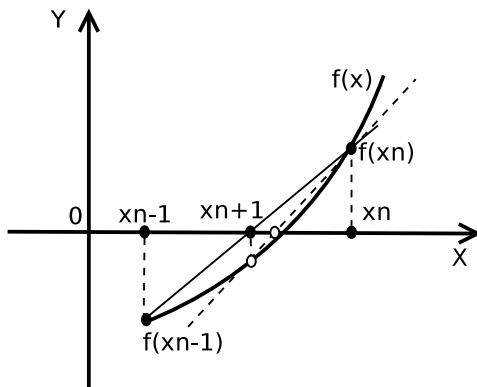


Рис. 2.3: Метод секущих

Из рисунка 2.3 имеем:

$$\frac{f(x_n) - 0}{x_n - x_{n+1}} = \frac{0 - f(x_{n-1})}{x_{n+1} - x_{n-1}}$$

Тогда после преобразований получаем:

$$x_{n+1} = \frac{f(x_n) \cdot x_{n-1} - f(x_{n-1}) \cdot x_n}{f(x_n) - f(x_{n-1})}$$

Обычно данную формулу преобразуют следующим образом:

$$x_{n+1} = \frac{x_n \cdot f(x_n) - x_{n-1} \cdot f(x_n) + f(x_n) \cdot x_{n-1} - f(x_{n-1}) \cdot x_n}{f(x_n) - f(x_{n-1})}$$

Окончательно имеем:

$$x_{n+1} = x_n - f(x_n) \cdot \frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})}$$

Задания

1. Проведите тестирование методов деления пополам, касательных и секущих в соответствии с вариантом на примере решения уравнений:

1. $\sin(cx) - d = 0$

2. $e^{cx} - d = 0$

3. $\log_2(cx) - d = 0$

4. $x^3 + cx^2 + d = 0$

5. $x^4 + cx^3 - dx = 0$

6. $x^5 + cx^2 - d = 0$

при разных значениях параметров c, d . Сравните число итераций этих методов при одном и том же значении точности. Придумайте три своих уравнения и используйте реализованные методы для нахождения их корней. Для оценки корректности получаемых решений целесообразно использовать онлайн калькулятор от Wolfram, доступный по ссылке <http://www.wolframalpha.com/>

Например, на рисунке 2.4 представлен график, соответствующий решению уравнения $e^x - x^2 = 0$ (в пояснительную записку требуется добавить скриншот таких графиков, а также точное (если оно возможно) и приближенное решения уравнения в этом калькуляторе).

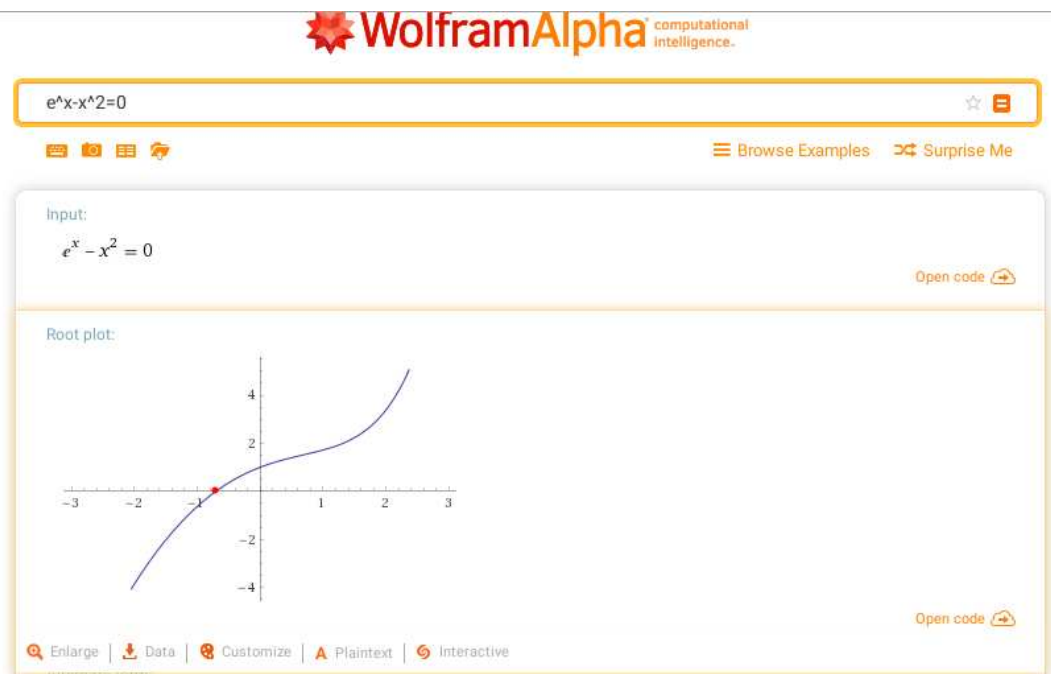


Рис. 2.4: Пример расчета в калькуляторе wolfram

Распределение заданий по вариантам

Номер варианта:	1	2	3	4	5	6
Номера методов:	1,3	2,3	1,2	2,3	2,3	1,2
Номера уравнений:	1,4,5	1,5,6	2,4,5	1,2,3	1,2,6	3,4,6

Номер варианта:	7	8	9	10	11	12
Номера методов:	1,3	2,3	1,2	2,3	2,3	1,2
Номера уравнений:	2,3,4	1,2,3	3,4,5	4,5,6	1,5,6	1,2,6

Номер варианта:	13	14	15	16	17	18
Номера методов:	1,3	2,3	1,2	2,3	2,3	1,2
Номера уравнений:	1,4,6	2,5,6	3,5,6	1,3,5	2,4,6	1,3,5

Номер варианта:	19	20	21	22	23	24
Номера методов:	1,3	2,3	1,2	2,3	2,3	1,2
Номера уравнений:	1,5,6	2,3,5	1,4,5	2,3,6	2,4,5	2,4,5

Номер варианта:	25	26	27	28	29	30
Номера методов:	1,3	2,3	1,2	2,3	2,3	1,2
Номера уравнений:	2,3,6	1,3,6	2,4,6	1,2,6	3,4,5	1,4,6

ГЛАВА 3

РАЗРАБОТКА ИГРОВОЙ ПРОГРАММЫ

Задания

1. Игра в кости: играющий называет любое число в диапазоне от 2 до 12 и ставку, которую он делает в этот ход. Программа с помощью датчика случайных чисел дважды выбирает числа от 1 до 6 («бросает кубик», на гранях которого цифры от 1 до 6). Если сумма выпавших цифр меньше 7 и играющий задумал число меньше 7, он выигрывает сделанную ставку. Если сумма выпавших цифр больше 7 и играющий задумал число больше 7, он также выигрывает сделанную ставку. Если играющий угадал сумму цифр, он получает в четыре раза больше очков, чем сделанная ставка. Ставка проиграна, если не имеет место ни одна из описанных ситуаций. В начальный момент у играющего 100 очков. Предложите и реализуйте не менее трех вариантов расширения функциональности этой игры (например, 1. вместо человека играет другая программа, 2.

казино копит проигранные ставки и т.п.).

2. Игра в морской бой: на поле 10×10 клеток программа размещает флот кораблей: 1 корабль — ряд из 4 клеток («четырёхпалубный»; линкор); 2 корабля — ряд из 3 клеток («трёхпалубные»; крейсера); 3 корабля — ряд из 2 клеток («двухпалубные»; эсминцы); 4 корабля — 1 клетка («однопалубные»; торпедные катера). При размещении корабли не могут касаться друг друга сторонами и углами. Предложите и реализуйте не менее трех вариантов расширения функциональности этой игры (например, играют два человека, играют две программы и т.п.).
3. Программа календарь-ежедневник: на заданную дату в заданное время реализовать возможность запланировать событие. Реализовать функции создания, редактирования, удаления событий. Предложите и реализуйте не менее трех вариантов расширения функциональности этой программы (можно ориентироваться на функциональность ежедневника Outlook и аналогичных мененджеров).
4. Игра жизнь: имеется поле, поделенное на клетки. Каждая клетка на этой поверхности может находиться в двух состояниях — «живая» или «мёртвая». Распределение живых клеток в начале игры называется первым поколением. Каждое следующее поколение (шаг) рассчитывается на основе предыдущего по таким правилам: пустая («мёртвая») клетка с Nx живыми клетками-соседями оживает; если у «живой» клетки есть не менее $(Nx - 1)$ живых соседей, то эта клетка продолжает жить, если «живых» соседей меньше $(Nx - 1)$ или больше $(Nx + 1)$, то клетка «умирает» (от «одиночества» или от «перенаселённости»). Предложите и реализуйте не менее трех вариантов расширения функциональности этой игры (например, игрок сам устанавливает правила взаимодействия клеток, расставля-

ет «живые» клетки первого поколения, конфигурации генерируются случайно и т.п.)

5. Устный счет: используя арифметические операции умножения, сложения, вычитания, деления, возведения в степень, программа случайно генерирует арифметическое выражение, используя в случае необходимости скобки, вычисляет его значение и предлагает человеку на время также выполнить вычисление. Предложите и реализуйте не менее трех вариантов расширения функциональности этой игры (например, регулируется длина выражения, используемые арифметические операции, предусматривается система бонусов, если игрок угадывает решение и т.п.)
6. Игра сапёр: на поле 10x10 программа случайно размещает фиксированное количество мин. Требуется разминировать поле. Игроку разрешается открывать любую клетку, или помечать ее признаком - заминировано. После каждой открытой клетки программа сообщает для каждой смежной с ней клетки - сколько смежных с ней мин. Если игрок открыл клетку с миной, он проиграл. Если он обезвредил все мины, он выиграл. Предложите и реализуйте не менее трех вариантов расширения функциональности этой игры (например, игрок сам выбирает размерность поля, количество мин, количество раз ошибиться и т.п.)
7. Игра пятнашки: в клетках поля 4x4 размещаются фишки от 1 до 15, одна клетка остается пустой. В пустую клетку можно перемещать любую и соседних с ней клеток. Реализовать перемещение фишек, визуализацию поля. Требуется вернуть искомую нумерацию. Предложите и реализуйте не менее трех вариантов расширения функциональности этой игры (например, программа сама тасует фишки, сама пытается собрать требуемую конфигурацию и т.п.)
8. Игра лабиринт: играющий перемещается в двухмерном пространстве по помещениям здания, план которого играюще-

му неизвестен. Начиная с произвольного помещения, путешественник должен найти выход из здания. Каждое помещение может иметь четыре двери: север, восток, юг, запад и соединяться с другими помещениями. План здания необходимо сгенерировать случайно. Порядок следования помещений в списке должен быть произвольным. Находясь в N-ом помещении, игрок может получить подсказку о правильном направлении движения, если верно выполнит тестовое задание по теме «Программирование на языке высокого уровня». Предложите и реализуйте не менее трех вариантов расширения функциональности этой игры (например, автоматическое или ручное прохождение роботом лабиринта, несколько выходов и т.п.)

9. Программа моделирование компьютера: имеется специальный регистр - аккумулятор, в который помещается результат арифметических операций, различных операций сравнения. Имеется оперативная память из 100 ячеек, куда можно записать любое целое число. Программа состоит из команд, каждая команда - это четырехзначное число. Первые две цифры это код команды, которую нужно выполнить: 10 - вводится слово с терминала в указанное место памяти; 11 - выводится слово на терминал из указанного адреса памяти; 20 - в аккумулятор помещается слово из указанного адреса памяти; 21 - в память помещается слово из аккумулятора; 30 - сложение слова аккумулятора и слова из указанного места в памяти (результат остается в аккумуляторе); 31 - вычитание слова аккумулятора и слова из указанного места в памяти (результат остается в аккумуляторе); 32 - деление слова аккумулятора на слово из указанного места памяти; 40 - переход по указанному адресу памяти; 41 - переход по указанному адресу памяти, если в аккумуляторе находится отрицательное число; 42 - переход по указанному адресу памяти, если в аккумуляторе находится нуль; 43 - останов, выполняется при завершении

программой своей работы. Реализовать ввод и выполнение программы. Предложите и реализуйте не менее трех вариантов расширения функциональности этой программы (например, можно расширить набор команд, пытаться программой писать программы самой себе, составить несколько полезных программ т.п.)

10. Крестики-нолики: в любую пустую клетку на поле 3x3 программа может ставить нолик, человек - крестик. Выигрывает тот, кто первым составит горизонтальный, вертикальный или диагональный ряд своих символов (крестиков или ноликов). Предложите и реализуйте не менее трех вариантов расширения функциональности этой игры (например, меняется размер поля, программа следует какой-то стратегии в игре, программы играют друг с другом и т.п.)

Распределение заданий по вариантам

Номер варианта:	1	2	3	4	5	6	7	8	9	10	11
Номер задания:	1	2	3	4	5	6	7	8	9	10	1

Номер варианта:	12	13	14	15	16	17	18	19	20
Номер задания:	2	3	4	5	6	7	8	9	10

Номер варианта:	21	22	23	24	25	26	27	28	29
Номер задания:	9	3	4	5	1	7	8	2	10

ЛИТЕРАТУРА

- [1] *Юркин А.* Задачник по программированию. – СПб.: Питер, 2002. – 192 с.
- [2] *Эпштейн М.С.* Практикум по программированию на языке С. – М.: Издательский центр «Академия», 2007. – 128 с.
- [3] *Валединский В.Д., Корнев А.А.* Методы программирования в примерах и задачах. – М.: Изд-во ЦПИ при механико-математическом ф-те МГУ, 2000. – 152 с.

ПРИЛОЖЕНИЕ А _____

_____ ОБРАЗЕЦ ТИТУЛЬНОГО ЛИСТА

Филиал «Котельники» государственного бюджетного
образовательного учреждения высшего образования
Московской области «Университет «Дубна»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
по курсовой работе по дисциплине
«Программирование на языке высокого уровня»

ВАРИАНТ №1

Выполнил:_____

студент группы ИВТ-11 Иванов И.И.

Проверил:_____

доцент, к.т.н. Артамонов Ю.Н.

Котельники – 2018

ПРИЛОЖЕНИЕ В

ПРИМЕРЫ БЛОК-СХЕМ

Требуется написать программу приближенного вычисления суммы ряда с точностью ε :

$$1 + \frac{x}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$$

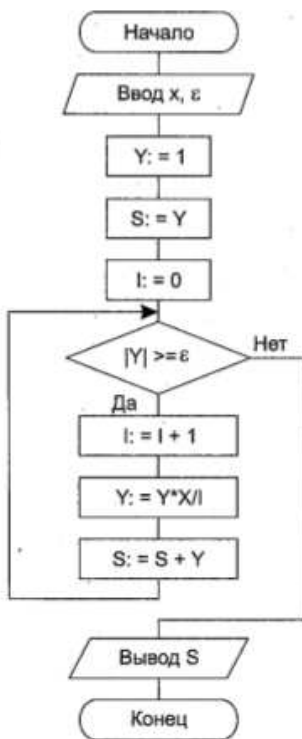


Рис. В.1: Блок-схема алгоритма приближенного вычисления суммы ряда

Листинг программного кода, соответствующего блок-схеме

```
#include <stdio.h>
#include <math.h>
main()
{
    float x, epsilon;
    printf("x="); scanf("%f", &x);
    printf("epsilon="); scanf("%f", &epsilon);
    float Y=1,S;
    S = Y;
    int i=0;
    while (fabs(Y) >= epsilon)
    {
        i++;
        Y = Y*x/i;
        S = S+Y;
    }
    printf("S=%f\n", S);
    return 0;
}
```

Дан числовой массив A , требуется найти индексы максимального и минимального элементов массива (в данной блок-схеме принята нумерация элементов массива с единицы)

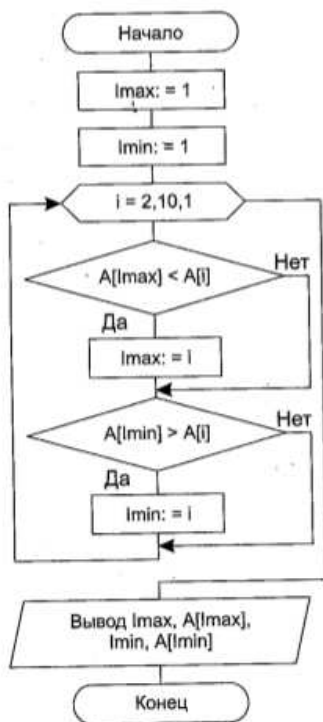


Рис. В.2: Блок-схема алгоритма поиска индексов максимального и минимального элементов массива

Листинг программного кода, соответствующего блок-схеме

```
#include <stdio.h>
main()
{
    int A[10] = {5,8,3,2,1,0,9,8,7,5};
    int Imax = 0, Imin = 0, i;
    for (i = 1; i<10; i++)
    {
        if (A[Imax]<A[i]) Imax = i;
        if (A[Imin]>A[i]) Imin = i;
    }
    printf("Imax=%d\n", Imax);
    printf("A[Imax]=%d\n", A[Imax]);
    printf("Imin=%d\n", Imin);
    printf("A[Imin]=%d\n", A[Imin]);
    return 0;
}
```