МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФГБОУ ВО «Кубанский государственный
аграрный университет имени И. Т. Трубилина»

Архитектурно-строительный факультет

Кафедра строительных материалов и конструкций

ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ РАСЧЕТА СТРОИТЕЛЬНЫХ КОНСТРУКЦИЙ

МЕТОДИЧЕСКИЕ УКАЗАНИЯ

по дисциплине
и для самостоятельной работы
студентов по направлению
08.03.01 Строительство

Краснодар
КубГАУ
2019

Составители: С. Е. Пересыпкин, М. В. Чумак

Информационные технологии расчета строительных конструкций : метод. указания по дисциплине и для самостоятельной работы / сост. С. Е. Пересыпкин, М. В. Чумак – Краснодар : КубГАУ, 2019. – 56 с.

В методических указаниях дано описание основных свойств строительных материалов, их классификация, сущность, взаимосвязь, влияние на качество материалов. Приведены технологии и порядок выполнения стандартных методик по испытанию для определения свойств и технических условий материалов. Представлены виды приборов и инструментов для проведения стандартных испытаний. Даны методы подбора состава тяжелого бетона и строительного раствора.

Предназначены для студентов, обучающихся по направлению подготовки 08.03.01 Строительство.

Рассмотрено и одобрено методической комиссией архитектурно-строительного факультета Кубанского государственного аграрного университета, протокол № 2 от 22.10.2019.

Председатель
методической комиссии  А. М. Блягоз

С О Д Е Р Ж А Н И Е

ВВЕДЕНИЕ

Известно, что в математике часто встречаются задачи, решение которых не удаётся получить в виде формулы, связывающей искомые величины с заданными. Про такие задачи говорят, что они не решаются в чистом виде. Для их решения стремятся найти какой-нибудь бесконечный процесс, сходящийся к искомому ответу. Если такой процесс указан, то, выполняя определенное число шагов и затем, обрывая вычисления, получаем приближенное решение задачи.

Процедура, связанная с проведением вычислений по строго определенной системе правил, которая задается характером процесса, называется алгоритмом. А основанные на них методы решения математических задач принято называть численными методами.

В настоящее время успешное решение большинства научно–технических задач в значительной степени зависит от умения оперативно применять ЭВМ. Вместе с тем использование ЭВМ не снимает всех проблем, которые возникают в ходе подготовки и решения этих задач. Так или иначе, процесс решения практической задачи включает ряд этапов, каждый из которых влияет на достоверность результата:

1 Постановка задачи и построение математической модели;

2 Разработка алгоритма;

3 Запись алгоритма на языке программирования;

4 Исполнение программы на ЭВМ;

5 Анализ полученных результатов.

В разделе 1 приводятся основные сведения, необходимые студентам для выполнения третьего этапа процесса решения задач в строительстве.

Раздел 2 содержит постановку задач и алгоритмы решения наиболее часто встречающихся задач при математическом анализе конструкций.

Раздел 3 посвящен вопросам расчета строительных конструкций методом конечных элементов с помощью программы STARK_ES, а так же четвертому и пятому этапам процесса решения задач в области строительства.

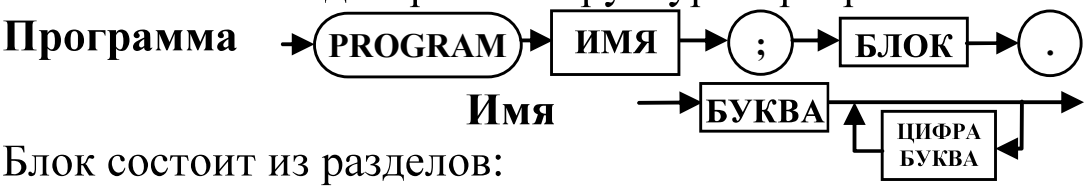
Раздел 1 ОСНОВЫ ПРОГРАММИРОВАНИЯ НА *PASCAL*

Алгоритм языка высокого уровня *PASCAL* разработан в конце 60-х профессором Н.Виртом (Цюрихская высшая техническая школа) – в честь французского математика и философа Блеза Паскаля.

Основные достоинства: гибкость и надежность; простота и ясность конструкций; возможность удовлетворения требованиям структурного программирования и др.

1.1 Программирование на языке PASCAL состоит из заголовка и собственно программы, называемой блоком

Синтаксическая диаграмма структуры программы



Блок состоит из разделов:
меток; констант; типов; переменных; процедур и функции; операторов.

Разделителем между разделами и операторами служит “;”, в конце программы ставится символ “.”. Раздел операторов заключается в операторные скобки BEGIN… END. В разделе операторов указывается последовательность действий, которые должны выполнятся ЭВМ.

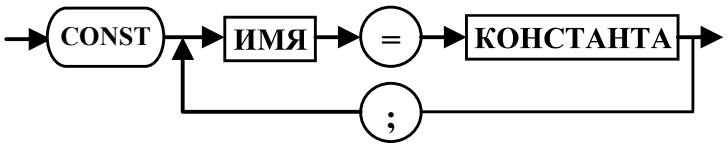
Все остальные разделы носят описательный характер. Любой раздел, кроме последнего, может отсутствовать.

1.2 Программа, написанная на языке PASCAL, оперирует некоторыми объектами, называемыми данными

Существуют 4 стандартных элементарных типа данных: целый (integer); вещественный (real); символьный (char); булевский (boolean). Каждый элемент данных в программе является либо константой, либо переменной.

Именованная константа отличается от переменной тем, что её значение не может изменяться во время выполнения программы. Тип константы однозначно определяется её значением и в явном виде не указывается описание констант в соответствии со следующей синтаксической диаграммой:

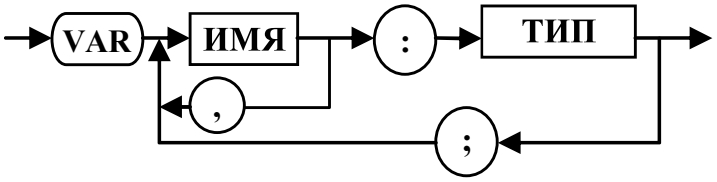
Раздел описания констант



Например: const i=2; a=4.15;

Тип переменной задается в разделе описания переменных в соответствии со следующей синтаксической диаграммой:

Раздел описания переменных



Над переменными целого и вещественного типа определяются операции «+», «-», «*», «/».

С аргументами целого и вещественного типа могут использоваться стандартные функции (табл. 1.1).

Запись вещественного числа возможна в виде числа с фиксированной точкой 151.35 или в экспоненциальной форме 1.5135E2.

Таблица 1.1

Тип аргумента	Имя функции	Математическое значение	Тип результата
integer/real	sin(x)	sin x	real
integer/real	cos(x)	cos x	real
integer/real	ln(x)	ln x	real
integer/real	sqrt(x)	$\sqrt{x}$	real
integer/real	arctan(x)	arctg x	real
integer/real	exp(x)	e^x	real
integer/real	sqr(x)	x^2	integer/real
integer/real	abs(x)	$\lvert x \rvert$	integer/real

Символы (элементы конечного и упорядоченного набора знаков), заключённые в апострофы, обозначают константу символьного типа.: const A='x', txt=’PASCAL’.

Оператор присваивания обозначается так: ”:=”. При выполнении данного оператора вычисляется выражение, стоящее в правой части и его значение присваивается переменной в правой части. При этом тип выражения должен соответствовать типу переменной. Допускается присваивать переменной вещественного типа значение выражения целого типа.

1.3 Организация ввода/вывода

Ввод-вывод связан с обменом информации между оперативной памятью и внешними носителями информации. Существует четыре процедуры ввода-вывода:

Read, readln, write, writeln.

1.3.1 Задание исходных данных

Задание исходных данных осуществляется тремя основными способами.

1) В разделе констант задаются соответствующие значения:

Пример: Вычислить: $y = ax^2 + bx + c$, при $a = 2{,}5; x = 7{,}3; b = -17{,}5; c = 548.$

Фрагмент программы:

const $a = 2.5; x = 7.3; b = -17.5; c = 548;$

var y:real;

begin $y := a * sqr(x) + b * x + c;$... end.

Недостаток: такой способ задания исходных данных не всегда удобен, так как позволяет производить вычислять только для одного набора параметров.

2) В разделе переменных описываются переменные, а в разделе операторов им присваиваются соответствующие значения:

```
var     a,x,b,y : real; c: integer;
begin   a:=2.5; x:=7.3; b:=-17.5; c:=548;
        y:=a*sqr(x)+b*x+c;…end.
```

Недостаток: несмотря на то, что в программе можно изменить значения параметров, набор данных статический.

3) Операторы read и readln позволяют выполнять программы с различными наборами исходных данных:

```
…begin read (a,x,b,c); y:=a*sqr(x)+b*x+c;…
```

1.3.2 Вывод данных

Вывод данных обеспечивают два оператора: write и writeln (печать в текущей строке с переходом на следующую).

Пример. Вывести на экран строку ”Я программирую на PASCAL”, задав её как константу символьного типа.

```
const text=’Я программирую на PASCAL’;
```

begin write(text) end.

Пример. Вывести на экран вещественные числа, заданные в разделе констант.

```
const a=2; b=5; c=4.2;
begin write (a:2, b:2, c:5:2) end.
```

При печати под значения констант a и b отводится 2 позиции. Для c вторая цифра указывает на число позиций после запятой.

Пример: Вычислить и вывести квадрат и корень числа ‘2’

```
const a=2;
var b:integer; c:real;
begin: b:=sqr(a); c:=sqrt(a); writeln(‘a=’, a:2, ‘ b=’, b:2);
writeln (‘c=’, c:5:3) end.
```

Результат выполнения программы:

```
a= 2  b= 4
c=1.414
```

1.4 Сложные операторы

1.4.1 Условный (выбор одного из двух действий):

If B then S1; if B then S1 else S2,

где B–выражение булевского типа.

S1 и S2–отдельные или сгруппированные вместе при помощи операторных скобок BEGIN…END (составные) операторы.

Пример. Необходимо присвоить целым переменным x и y соответственно значения 5 и 23, если A<0, и нули в противоположном случае.

```
…if A<0 then begin x:=5; y:=23 end
else begin x:=0; y:=0 end;…
```

1.4.2 Операторы цикла

При разработке алгоритмов решения большинства задач возникает необходимость повторения ряда шагов. Для организации таких повторов (циклов) при записи алгоритмов на языке PASCAL используются три разновидности операторов цикла : с параметром, с предусловием, с постусловием.

1) Если число повторений числа тела цикла заранее известно, то используется оператор цикла с параметром

for P:=NC to KC do A

for P:=PC downto KC do A

NC, KC – выражения, задающие начальное и конечное значение параметра цикла;

P – параметр цикла;

A – простой или составной оператор.

Пример. Напечатать таблицу квадратов натуральных чисел от 1 до 10.

```
var n, b: integer;
begin for n:=1 to 10 do
  begin b:= sqr (n); writeln (n:3, b:4) end
end.
```

2). Цикл с предусловием.

While B do A

Если выражение B имеет значение TRUE, то выполняется оператор A до тех пор, пока значение выражения B не станет равным FALSE, после чего управление передаётся оператору, следующему за оператором цикла.

Печать таблицы квадратов натуральных чисел от 1 до 10 может быть реализована так:

```
var          n,a: integer;
begin writeln (‘  n’,’   a’); n:=1;
  while n<=10 do
```

```
begin a:=sqr(n); writeln (n:3; a:4); n:=n+1 end
end.
```

В последнем примере перед таблицей будет выведен заголовок.

2) Цикл с постусловием.

REPEAT A1;A2;…;AN UNTIL B,

где A1;A2;…;AN–операторы цикла; B – выражение булевского типа.

Выход из цикла осуществляется, если B получает значение TRUE. (в цикле с предусловием выход осуществляется при получении выражением B значения FALSE)

Ниже приведен фрагмент программы, реализующий указание предыдущего примера:

```
…repeat a:=sqr(n);
writeln(n:3, a:4); n:=n+1 until n>10;…
```

1.5 Процедуры и функции

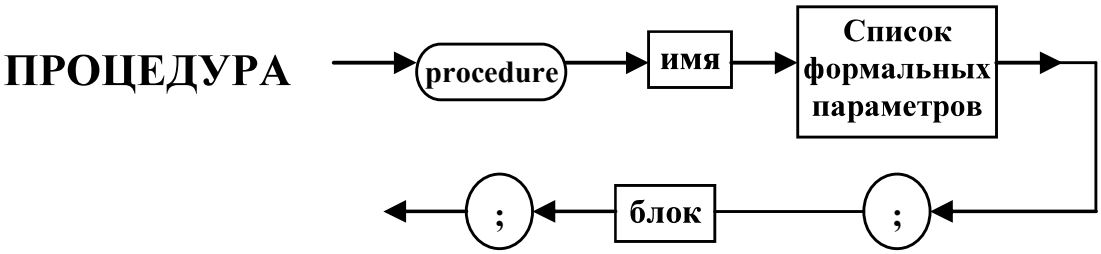
Процедуры и функции аналогичны программам в миниатюре и имеют общее название подпрограммы. Применение подпрограммы даёт возможность уменьшать число повторений одной и той же последовательности операторов, а также конструировать программу как набор отдельных подпрограмм. Это позволяет получить более логичный процесс программирования.

В программе описание процедур и функций должно располагаться между разделами переменных и операторов. Каждая процедура или функция определяется только один раз, но может использоваться многократно. Структура процедур и функций аналогична структуре полной программы на языке PASCAL. В процедурах и функциях могут быть описаны собственные константы, типы, переменные и даже собственные процедуры и функции.

1.5.1 Подпрограмма-процедура

Описание процедуры начинается с заголовка, в котором задаётся имя процедуры и список формальных параметров с указанием их типа. Процедура может быть без параметров, тогда в заголовке указывается только её имя.

С помощью параметров осуществляется передача исходных данных в процедуру, а также передача результатов работы обратно в вызывающую программу.



Список формальных параметров может включать в себя параметры–значения, параметры переменных (перед ними должно стоять служебное слово var). После заголовка процедуры следуют разделы в том же порядке, что и в программе.

Вызов и выполнение процедуры осуществляется при помощи оператора процедуры:

<имя процедуры> (<список фактических параметров>)

Список формальных и фактических параметров должны находиться в соответствии (количество, порядок следования, тип).

При вызове процедуры сначала передаются параметры, при этом параметры–значения передаются по значению, а параметры–переменные– по ссылке. Основное отличие этих способов передачи параметров заключается в том, что присваивание значений параметру–переменной внутри процедуры одновременно выполняется и для соответствующего аргумента (фактического параметра). Таким образом, пара-

метры, в которые записываются результаты работы процедуры, должны передаваться только по ссылке.

Параметры, через которые в процедуру передаются исходные данные, передаются по значению.

Пример. Составить программу вычисления суммы факториалов n первых чисел.

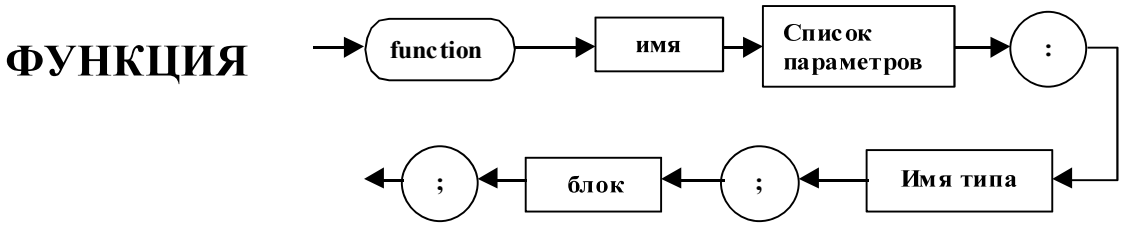
```
var i,n,s,a:integer;
procedure fact (k: integer; var j:integer);
var i:integer;
begin j:=1;
for i:=1 to k do j:=j*i
end;
BEGIN read (n); s:=0;
for i:=1 to n do begin
fact (i,a);s:=s+a end;
writeln ('a=',a:2, 's=', s:5)
END.
```

В списке формальных параметров процедуры fakt k– параметр– значение, по которому подпрограмма получает очередное число, через параметр–переменную j в основную программу передаётся значение факториала числа.

Процедуры возвращают результат в основную программу не только при помощи параметров–переменных, но и непосредственно изменяя глобальные переменные. Переменные, описанные в основной программе, являются глобальными по отношению к внутренним подпрограммам (i,n,s,a). Переменные, описанные внутри подпрограмм, называются локальными(i). Локальные переменные существуют только при выполнении подпрограмм и недоступны в основной программе.

1.5.2 Подпрограмм–функция

Результатом работы функции является одно скалярное значение. Тип результата задается в заголовке функции.



Среди входящих в функцию операторов должен присутствовать хотя бы один оператор присваивания, в левой части которого стоит имя данной функции. Этот оператор определяет значение, вырабатываемое функцией.

Вызов и выполнение функции производятся при вычислении значения указателя функции, который входит в некоторые выражения.

Задание предыдущего примера реализуется с помощью функции так:

```
var i,n,s: integer;
function factor(k: integer):integer;
var i,j:integer;
begin i:=1;
for j:=1 to k do i:=i*j;
factor:=i;
BEGIN read(n); s:=0;
for i:=1 to n do s:=s+factor (i);
writeln (‘a=’,a:2; ‘s=’,s:5)
END.
```

Раздел 2 АЛГОРИТМЫ РЕШЕНИЯ ЗАДАЧ

2.1 Алгебраические и трансцендентные уравнения с одной неизвестной

Решение нелинейных уравнений с одной переменной представляет одну из важных задач прикладного анализа, необходимость в которой возникает в многочисленных и разнообразных разделах физики, техники и других областях.

В общем случае нелинейное уравнение можно записать в виде:

$$F(x) = 0, \qquad (2.1)$$

где функция F(x) определена и непрерывна на конечном или бесконечном интервале $[a;b]$. Всякое число $\xi \in [a;b]$, обращающее функцию F(x) в нуль, т.е. такое, при котором $F(\xi) = 0$, называется корнем уравнения (2.1).

Нелинейные уравнения с одним неизвестным подразделяются на алгебраические и трансцендентные. Уравнение (2.1) называется алгебраическим, если функция является алгебраической. Путем алгебраических преобразований из всякого алгебраического уравнения можно получить уравнение в канонической форме:

$$P_n(x) = a_0 x^n + a_1 x^{n-1} + ... + a_n = 0, \qquad (2.2)$$

Если функция F(x) не является алгебраической, то уравнение (2.1) называется трансцендентным. Такие задачи встречаются уже в курсе строительной механики. Например, при определении критической силы для стержня, изображенного на рис. 2.1, возникает необходимость решать трансцендентное уравнение:

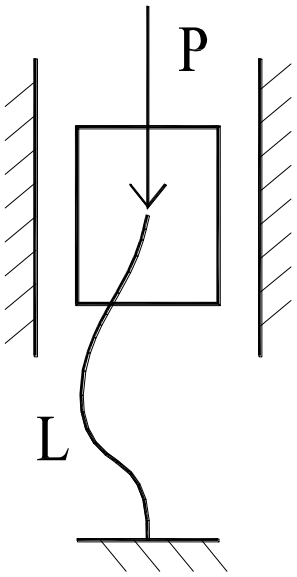


Рисунок 2.1

$$x - tgx = 0, \qquad (2.3)$$

где $x = \sqrt{\frac{P}{EI}}L$.

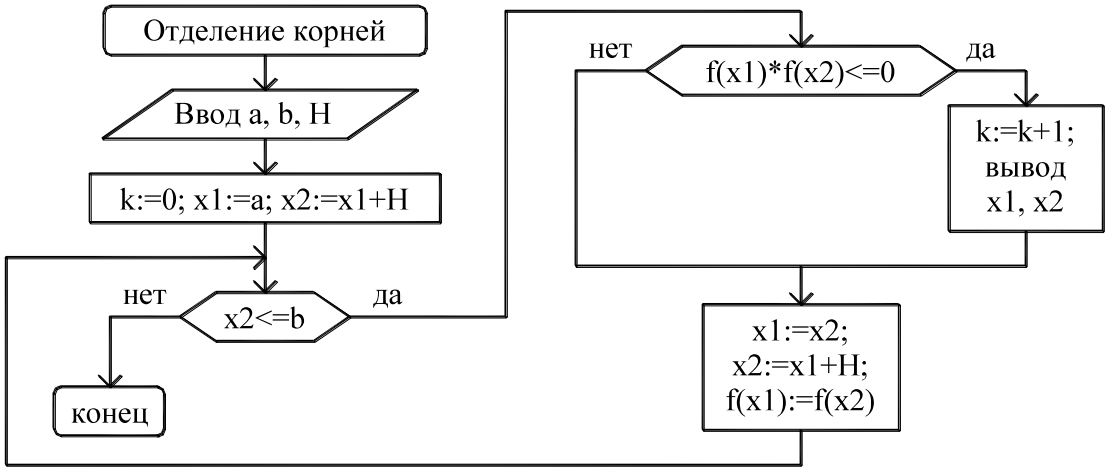
Решение таких уравнений, а также уравнений типа (2.2) при n>4 возможно только численными методами.

Решение уравнения с одной неизвестной состоит из двух этапов:

–отделение корней;

–уточнение корней.

Первый этап, на котором находятся достаточно малые интервалы области определения функции [x1,x2], содержащие один корень, иллюстрирован следующей блок–схемой (k – счётчик корней):



Алгоритм отделения корней реализован в процедуре ABROAD программы EQUIP1.PAS

На втором этапе применяются несколько численных методов. Рассмотрим два из них.

2.1.1 Метод половинного деления

После отделения корней, т.е. нахождения отрезков, содержащих по одному корню, возникает задача их уточнения, т.е. вычисления их значения с заданной точностью.

Пусть уравнение (2.1) имеет на отрезке $[a;b]$ единственный корень, а F(x) на нём непрерывна. Разделим $[a;b]$ пополам точкой c=(a + b)/2. Если F(c)≠0 (что практически наиболее вероятно), то возможны два случая: либо F(x) сменяет знак на отрезке $[a;b]$ (рис. 2.2,а),

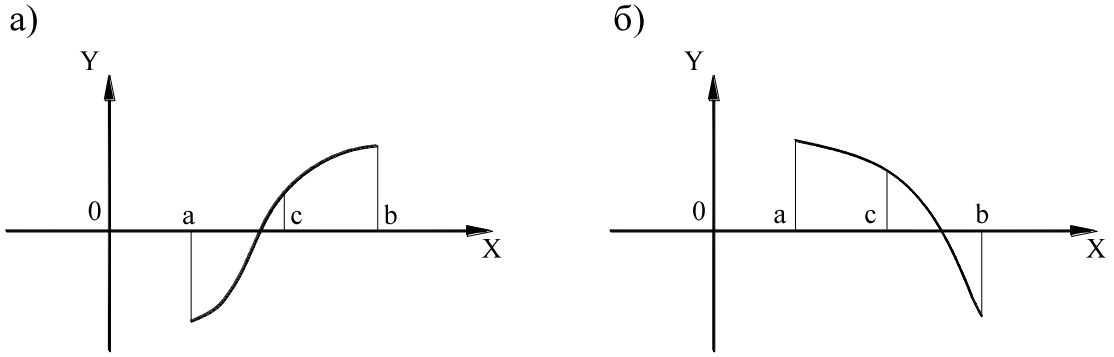
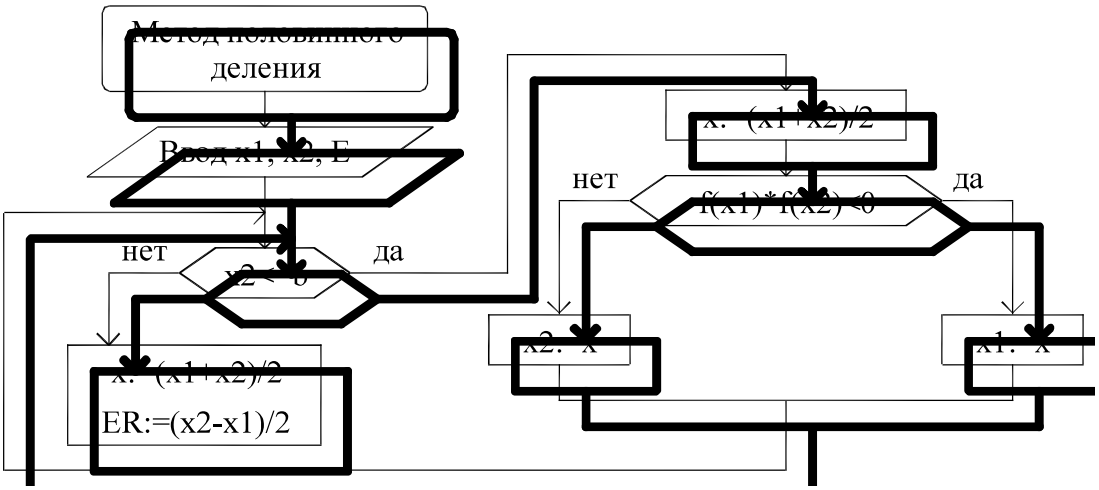


Рисунок 2.2

либо на отрезке $[c;b]$ (рис. 2.2,б). Выбирая в каждом случае тот из отрезков, на котором функция меняет знак, и, продолжая процесс половинного деления дальше, можно дойти до сколь угодно малого отрезка, содержащего корень уравнения (2.1).

Метод иллюстрирует следующая блок-схема:



Алгоритм метода половинного деления реализован в процедуре DICH программы EQUIP.PAS.

2.1.2 Метод простой итерации

Заменим уравнение (2.1) равносильным уравнением

$$x = f(x) \qquad (2.4)$$

Пусть ξ – корень (2.4), а x_0 – полученное каким-либо способом нулевое приближение к корню ξ.

Подставляя x_0 в каждую часть (2.4), получим некоторое число $x_1 = f(x_0)$. Проделаем то же самое с x_1, получим $x_2 = f(x_1)$ и т.д. Применяя шаг за шагом соотношение $x_n = f(x_{n-1})$. для n=1,2,…, образуем числовую последовательность

$$x_0, x_1, \ldots, x_n, \qquad (2.5)$$

которую называют последовательностью приближений или итерационной последовательностью (от лат. Iteratio– повторение).

Графическая интерпретация метода дана на рис. 2.3.

а) б)

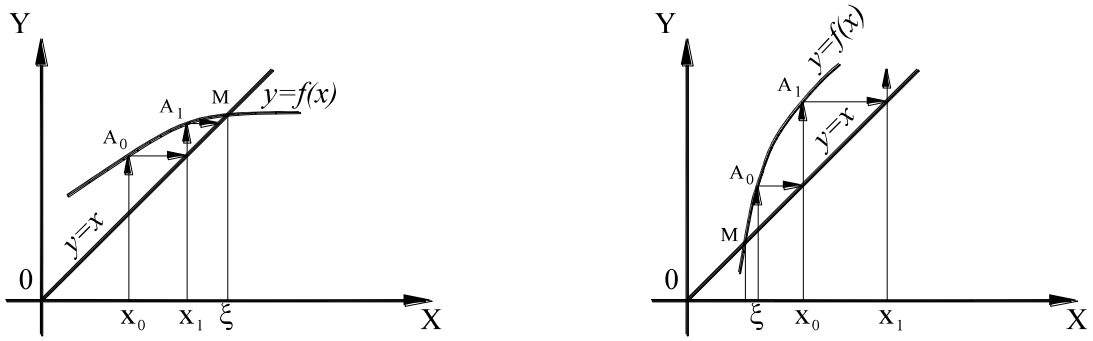


Рисунок 2.3

Последовательность приближений (2.5) может быть как сходящейся, так и расходящейся (рис. 2.3 а, б)

Если последовательность (2.5) сходится, а f(x) – непрерывна, то предел последовательности является корнем уравнения (2.4).

Действительно, пусть $\xi = \lim\limits_{n\to\infty} x_n$.

Перейдя к пределу в равенстве $x_n = f(x_{n-1})$, получим:

$$\lim_{n\to\infty} x_n = \lim_{n\to\infty} f\left(x_{n-1}\right) = f\left(\lim_{n\to\infty} x_{n-1}\right) = f\left(\xi\right) \qquad (2.6)$$

т.е. $\xi = f(\xi)$.

Как видно из рис. 2.3,б применение метода итерации может и не привести к уточнению корня. Достаточные условия сходимости итерационного процесса выясняется следующей теоремой.

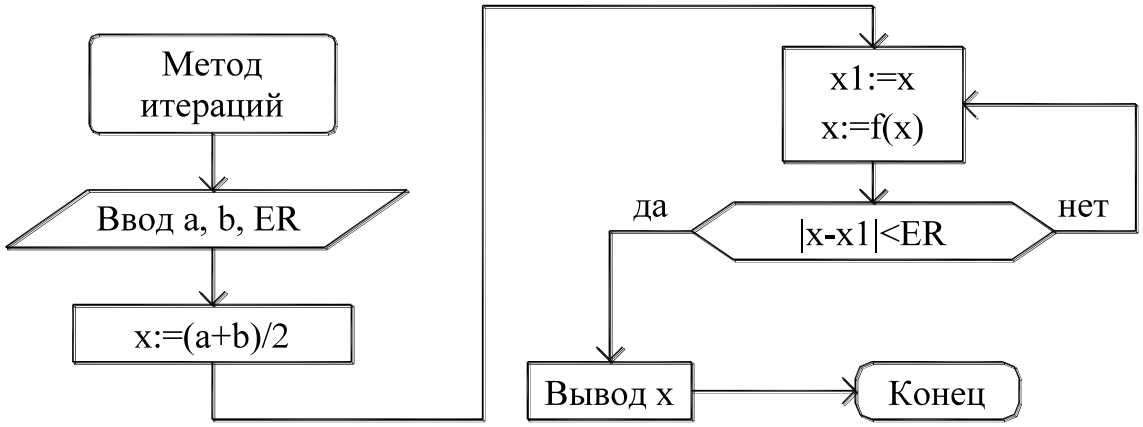
Теорема 2.1. Пусть уравнение $x = f(x)$ имеет единственный корень на [a;b] и выполнены условия:

- f(x) определена и дифференцируема на [a;b];

- $f(x) \in [a;b]$ для всех $x \in [a;b]$;

- существует такое вещественное q, что $|f'(x)| \le q < 1$, для всех $x \in [a;b]$

Тогда итерационная последовательность $x_n = f(x_{n-1})$ (n=1,2,…) сходится при любом начальном члене $x_0 \in [a;b]$. Для нахождения корня (2.4) с точностью ε нужно продолжить итерации до тех пор, пока не выполнится условие: $|x_n - x_{n-1}| \le \varepsilon(1-q)/q$, где q можно получить как верхнюю грань $|f'(x)|$ при $x \in [a;b]$. Алгоритм решения уравнения (2.4) иллюстрирован следующей блок-схемой:



Пример выполнения задания №1

Определить методами хорд и простой итерации корни нелинейного уравнения с одной неизвестной: $x^3 + \ln x + 1.5 = 0$.

Решение. Для определения области, содержащей все корни уравнения и приблизительного определения их значений построим графики двух функций: $u(x) = -x^3$ и $v(x) = \ln x + 1.5$.

По рис. 2.4 единственный корень находится в интервале [0;1]. Исходными данными для программы определения корней уравнения equip.pas являются:

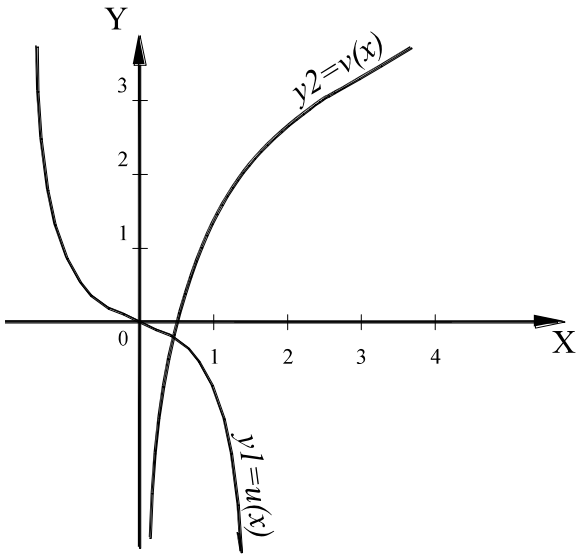


Рисунок 2.4

- границы области, содержащей единственный корень уравнения x0=0, x9=1;

- шаг итераций для определения отрезков отделения корней H=0.1;

- величина, регулирующая погрешность вычисления корня E=0.0001;

- максимальное число итераций при уточнении корня методом простой итерации m=30.

Результаты выполнения программы.
Уточнение корней по методу дихотомии:
x[1]=0,4510742 F(x)=-0.0004676.
Уточнение корней по методу простой итерации:
x[1]=0.4511557 F(x)=-0.0000564.

2.2 Элементы линейной алгебры

2.2.1 Общие сведения

Успешное решение большинства научно–технических задач в значительной степени зависит от умения быстро и точно получать решение систем линейных алгебраических уравнений.

Многие методы решения нелинейных задач также сводятся к решению некоторой последовательности линейных систем.

В линейной алгебре рассматриваются четыре класса основных задач:

–решение систем линейных алгебраических уравнений (СЛАУ);

–вычисление определителей;

–нахождение обратных матриц;

–определение собственных значений и собственных векторов матриц.

Эти методы имеют важное прикладное значение как сами по себе, так и в качестве вспомогательных средств при реализации многих алгоритмов вычислительной математики, математической физики, обработки результатов экспериментальных исследований.

Многообразие численных методов решения СЛАУ можно разделить на прямые (точные) и итерационные.

Прямые методы характеризуются тем, что дают решение системы за конечное число арифметических операций. Если все операции выполняются без ошибок округления, то решение получается точным.

Итерационные методы являются приближенными. Они дают решение системы как предел последовательности приближений, вычисляемых по единообразной схеме.

2.2.2 Методы решения линейных алгебраических уравнений. Метод Гаусса

Метод Гаусса относится к прямым методам решения СЛАУ. Алгоритм метода состоит из двух этапов.

Первый этап называется прямым ходом метода.

Так для СЛАУ:

$$\left.\begin{aligned} &a_{11}x_1 + a_{12}x_2 + ... + a_{1n}x_1 = a_{1,n+1}, \\ &a_{21}x_1 + a_{22}x_2 + ... + a_{2n}x_1 = a_{2,n+1}, \\ &..... \\ &a_{n1}x_1 + a_{n2}x_2 + ... + a_{nn}x_1 = a_{n,n+1}, \end{aligned}\right\} \qquad (2.7)$$

последовательно исключим неизвестные, начиная с x_1.

В результате получим СЛАУ с верной треугольной матрицей, у которой все элементы ниже главной диагонали равны нулю.

$$\left.\begin{aligned} x_1 + a_{12}^{(1)}x_2 + a_{13}^{(1)}x_3 + ... + a_{1n}^{(1)}x_n &= a_{1,n+1}^{(1)}, \\ x_2 + a_{23}^{(2)}x_3 + ... + a_{2n}^{(2)}x_n &= a_{2,n+1}^{(2)}, \\ ..... \\ x_n &= a_{n,n+1.}^{(n)} \end{aligned}\right\} \qquad (2.8)$$

Запишем выражения для неизвестных x_k и преобразования элементов расширенной матрицы системы.

$$\left.\begin{aligned} &x_k = \left(a_{k,n+1}^{(k)} - \sum_{j=k+1}^{n} a_{kj}^{(k)}x_j \right), \\ &a_{kj}^{(k)} = a_{kj}^{(k-1)} / a_{kk}^{(k-1)}, \\ &a_{ij}^{(k)} = a_{ij}^{(k-1)} - a_{ik}^{(k-1)}a_{kj}^{(k)}, k < i \le n, k \le j \le n = 1 \end{aligned}\right\} \qquad (2.9)$$

Второй этап решения СЛАУ называется обратным ходом метода Гаусса и состоит в последовательном определении неизвестного x_k.

Контроль полученных решений можно провести путем их подстановки в исходную СЛАУ и вычисления невязок r_k, разностей между правыми и левыми частями уравнений:

$$r_k = a_{k_1 n+1} - \sum_{j=1}^{n} a_{kj} x_j \qquad (2.10)$$

При малой погрешности решений величины r_k будут близки к нулю.

2.2.3 Метод Зейделя

Если после преобразования системы (2.7) к виду:

$$\left.\begin{aligned} x_1 &= (a_{1,n+1} - a_{11}x_1 - a_{12}x_2 - ... - a_{1n}x_n)/a_{11} + x_1, \\ x_2 &= (a_{2,n+1} - a_{21}x_1 - a_{22}x_2 - ... - a_{2n}x_n)/a_{22} + x_2, \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ x_n &= (a_{n,n+1} - a_{n1}x_1 - a_{n2}x_2 - ... - a_{nn}x_n)/a_{nn} + x_n; \end{aligned}\right\} \qquad (2.11)$$

задать столбец начальных приближений $x_1^0, x_2^0, ..., x_n^0$ и подставить их в правые части уравнений системы (2.11), можно получить новые приближения $x_1^0, x_2^0, ..., x_n^0$. Продолжая подобные действия до тех пор, пока не будут выполнены условия $|x_k^{(m+1)} - x_k^{(m)}| < \varepsilon$, ($\varepsilon$ – заданная погрешность, k=1,2,…,n) можно получить итерационный процесс, являющийся обобщением метода простых итераций (п.2.2) на системы уравнений. Если использовать приближения к решениям, найденные при выполнении текущей итерации, то можно организовать итерационный процесс, известный как метод Зейделя, который приводит к ускорению сходимости:

$$\left.\begin{aligned} x_1^{(m+1)} &= \varphi_1(x_1^{(m)}, x_2^{(m)}, ..., x_n^{(m)}), \\ x_2^{(m+1)} &= \varphi_2(x_1^{(m+1)}, x_2^{(m)}, ..., x_n^{(m)}), \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ x_n^{(m+1)} &= \varphi_n(x_1^{(m+1)}, x_2^{(m+1)}, ..., x_n^{(m)}). \end{aligned}\right\} \qquad (2.14)$$

Пример выполнения задания N2

Методами Гаусса и Зейделя решить систему уравнений (2.13) с точностью ε=0.001.

$$\left.\begin{aligned} 2.34x_1 - 4.21x_2 - 11.61x_3 &= 14.41 \\ 8.04x_1 + 5.22x_2 + 0.27x_3 &= -6.44 \\ 3.92x_1 - 7.99x_2 + 8.37x_3 &= 55.56 \end{aligned}\right\} \quad (2.13)$$

Решение. В соответствии с методом Гаусса вводим коэффициенты расширенной матрицы в программу GAUSS.PAS.

В результате выполнения программы получаем значения корней:

x[1]=2.293021; x[2]=-4.815522; x[3]=0.967185.

Для решения по методу Зейделя преобразуем исходную систему так, чтобы диагональные коэффициенты были по возможности преобладающими.

$$\left.\begin{aligned} 8.04x_1 + 5.22x_2 + 0.27x_3 &= -6.44 \\ 6.26x_1 - 12.2x_2 - 3.24x_3 &= 69.97 \\ 2.34x_1 - 4.21x_2 - 11.61x_3 &= 14.41 \end{aligned}\right\} \quad (2.14)$$

С этой целью первым уравнением преобразованной системы (2.14) возьмем второе уравнение исходной, третьим - первое, а вторым - сумму первого с третьим. В качестве исходных данных для программы ZEID.PAS (Приложение 2) вводим число уравнений системы n=3, максимальное число итераций m=30, точность решения системы E=0.001, коэффициенты расширенной матрицы системы (3.14).

Результаты решения по методу Зейделя:

x[1]=2.293182, x[2]=-4,815438, x[3]=0.967187, получено на 10-ом шаге.

2.3 Приближение функций

Основу всякой вычислительной работы составляет вычисление значений функций. При этом последняя может быть задана тремя различными способами: аналитическим выражением, в табличном виде и неявно - как решение некоторого уравнения.

Если функции задаются таблично значениями в некоторых точках, называемых узлами, то возникает задача определения их значений в промежутках таблицы.

В таких случаях выбирают некоторую модельную функцию (например, многочлен), которая является хорошим приближением к исходной в промежутках таблицы.

Критерий приближения или согласия определяет тип задачи приближения. Если требуется, чтобы приближающая функция совпадала с узловыми значениями - это задача интерполирования. Если же постулируется условие минимальности суммы квадратов отклонений между заданной функцией и приближающей в узловых точках, то - это задача аппроксимации.

2.3.1 Интерполяция

Использование в качестве модельных функций многочленов удобно, ввиду их простоты и возможности решения задачи численного дифференцирования и интегрирования.

Иллюстрируем данный способ приближения функций интерполяционным многочленом Лагранжа в виде:

$$P_n(x) = \sum_{i=0}^{n} f_i \prod_{j=0}^{n} \frac{x - x_j}{x_i - x_j} \quad , \; j \neq i \, . \qquad (2.15)$$

Программа LAGR.PAS реализует алгоритм интерполяции функции многочленом Лагранжа.

2.3.2 Метод наименьших квадратов

Если некоторая функция, заданная таблично (табл. 2.1), получена в ходе эксперимента, то использование в качестве приближающей функции многочлена Лагранжа не всегда оправдано. Значения экспериментальной функции могут быть сомнительными в силу погрешности измерения, а сама функция может зависеть от многих случайных факторов. Поэтому, совпадение приближающей и исходной функций в узлах таблицы вовсе не означает совпадения характеров их поведения.

Таблица 2.1

x	x_1	x_2	…	x_n
y	y_1	y_2	…	y_n

Сформулируем задачу приближения так, чтобы с самого начала обязательно учитывался характер исходной функции: найти функцию заданного вида: $y=F(x)$, которая в точках $x_1,x_2 \ldots x_n$ принимает значения $\bar{y}_1, \bar{y}_2, ..., \bar{y}_n$ как можно более близкие к табличным значениям $y_1,y_2,\ldots,y_n$. Тогда требование близости можно сформулировать так, чтобы расстояние между точками $M(y_1,y_2,\ldots,y_n)$ и $\bar{M}(\bar{y}_1, \bar{y}_2, ..., \bar{y}_n)$, было наименьшим. Это равносильно условию, в соответствии с которым, сумма квадратов $(y_1-\bar{y}_1)^2+(y_2-\bar{y}_2)^2+...+(y_n-\bar{y}_n)^2$ должна быть наименьшей.

В качестве приближающих функций можно использовать:

линейную- y=ax+b, степенную- $y=a\,x^{b}$, показательную- $y=a\,e^{bx}$ и другие, где a,b- параметры, выбор которых определяет степень близости приближающей и исходной функций.

Для случая приближающей функции с двумя параметрами $F(x_i,a,b)=\overline{y}_i$, i=1,2,…,n необходимое условие экстремума:

$$\left.\begin{aligned}\sum_{i=1}^{n}[y_i-F(x_i,a,b)]F_a'(x_i,a,b)=0,\\ \sum_{i=1}^{n}[y_i-F(x_i,a,b)]F_b'(x_i,a,b)=0.\end{aligned}\right\} \qquad (2.16)$$

Решив систему (2.16) относительно параметров a,b, можно получить конкретный вид искомой функции F(x,a,b).

Например, для случая линейной регрессии система (2.16) примет вид:

$$\left.\begin{aligned}M_{x^2}*a+M_x*b=M_{xy},\\ M_x*a+b=M_y,\end{aligned}\right\} \qquad (2.17)$$

где $M_x=\frac{1}{n}\sum x_i;\ M_y=\frac{1}{n}\sum y_i;\ M_{xy}=\frac{1}{n}\sum x_i y_i;\ M_{x^2}=\frac{1}{n}\sum x_i^{\,2}.$

Из двух разных приближений лучшим надо считать то, для которого сумма квадратов отклонений

$$\sigma=\sum_{i=1}^{n}[y_i-F(x_i,a,b)]^2 \qquad (2.18)$$

имеет наименьшее значение.

Пример выполнения задания N 3

1) Составить интерполяционный многочлен Лагранжа для функции, заданной таблично и определить значения в промежуточных точках. Построить график, отметив на нем

узловые точки. Уплотнить заданную таблицу с помощью программы LAGR.PAS.

2) Построить линейное уравнение регрессии для заданной функции. С помощью программы MNK.PAS в соответствии с принципом наименьших квадратов найти наилучшее приближение заданной функции.

Таблица 2.2

x	1	2	3
y=f(x)	0.8	2.5	5.2

Решение. 1) Строим многочлен Лагранжа в соответствии с формулой (2.15):

$$P_2(x) = 0.8\frac{(x-2)(x-3)}{(1-2)(1-3)} + 2.5\frac{(x-1)(x-3)}{(2-1)(2-3)} + 5.2\frac{(x-1)(x-2)}{(3-1)(3-2)} =$$

$$= 0.4(x^2 - 5x + 6) - 2.5(x^2 - 4x + 3) + 2.6(x^2 - 3x + 2) =$$

$$= 0.5x^2 + 0.2x + 0.1;$$

$$P_2(1.5) = 1.525,\ P_2(2.5) = 3.725$$

Для расчета по программе LAGR.PAS вводим количество узлов n=3, крайние значения узлов и шаг уплотнения табл. 2.2: x0=1, x9=3, H=0.5.

Результаты выполнения программы LAGR.PAS:

X	P	PI	PII
1	0.8	1.2	1
1.5	1.525	1.7	1
2	2.5	2.2	1
2.5	3.725	2.7	1
3	5.2	3.2	1

В третьем и четвертом столбцах приведены значения первой PI и второй PII производных многочлена Лагранжа, вычисленные в процедуре PL. Как видно значения в проме-

жутках узлов, вычисленные аналитически, совпадают с результатами уплотнения заданной таблицы по программе.

2) Строим линейное уравнение регрессии для функции, заданной табл. 2.2.

В соответствии с (2.17):

$$M_{x^2} = \frac{1}{3}(1+4+9) = 4.67, \quad M_x = \frac{1}{3}(1+2+3) = 2,$$

$$M_y = \frac{1}{3}(0.8+2.5+5.2) = 2.83, \quad M_{xy} = \frac{1}{3}(0.8+5+15.6) = 7.13.$$

$$\left.\begin{array}{l} 4.67 * a + 2 * b = 7.13, \\ 2 * a + b = 2.83. \end{array}\right\}$$

Решение системы (4.6) даст: a=2.19, b=-1.56. Тогда уравнение линейной регрессии: y=2.19x-1.56. График линейной регрессии показан на рис. 2.5.

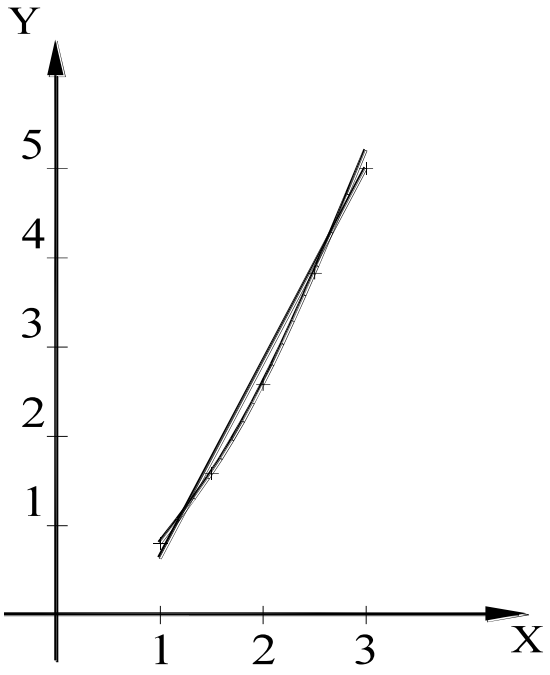


Рисунок 2.5

В программу аппроксимирования по методу наименьших квадратов MNK.PAS вводим число узлов n=3 и узловые значения исходной функции.

Результаты расчета для трех типов уравнений регрессии:

k=1
y=2.2*x-1.566667
s=0.16666667

x	f(x)	y
1	0.8	0.63333
2	2.5	2.83333
3	5.2	5.03333

k=2
y=0.7920457*x^1.6973342
s=0.01252834

x	f(x)	y
1	0.8	0.792046
2	2.5	2.568613
3	5.2	5.111924

k=3
y=0.3358132*exp(0.9359011x)
s=0.23702824

x	f(x)	y
1	0.8	0.856159
2	2.5	2.182786
3	5.2	5.565034

Значение s представляет собой сумму квадратов отклонений σ по формуле (2.18). В соответствии с критерием минимальности этой суммы наилучшим приближением следует признать степенную функцию при k=2.

2.4 Метод оптимизации

Выбор наилучшего варианта из всех возможных – называется процессом оптимизации. С точки зрения инженерных расчетов методы оптимизации позволяют выбрать наилучший вариант конструкции, наилучшее распределение ресурсов.

В процессе решения задач оптимизации должны быть найдены такие значения проектных параметров (линейные размеры, массы конструкций и т.п.), при которых определенная функция, которая называется целевой и зависит от проектных параметров, принимала минимум. Примерами

целевой функции является прочность или масса конструкции, объем выпуска продукции, стоимость перевозок, прибыль и др.

Можно выделить два типа задач оптимизации – безусловная и условная.

Безусловная задача оптимизации состоит в отыскании максимума или минимума функции $u=f(x_1, x_2, \dots, x_n)$ от n действий переменных и определении соответствующего значения аргументов на некотором множестве σ n-мерного пространства.

Условные задачи оптимизации, или задачи с ограничениями – это такие, при форматирование которых задаются некоторые условия (ограничения) на множестве σ.

Ограничения – неравенства выражают зависимость между проектными параметрами, которая должна учитываться при нахождении решения.

Ко второму типу задач оптимизации относятся задачи линейного программирования.

Пример выполнения задания N 4

На изготовление двух видов продукции p_1 и p_2 требуется три вида сырья s_1, s_2 и s_3.

Запасы каждого вида сырья ограничены и составляют соответственно b_1, b_2 и b_3 условных единиц. При заданной технологии количество сырья, необходимое для изготовления единицы первой продукции – a_{11}, a_{21}, a_{31} и единицы второй продукции - a_{12}, a_{22}, a_{32}.

Необходимо составить такой план выпуска продукции, при котором прибыль от реализации всей продукции была бы максимальной.

Первое изделие стоит - 10 т.руб., второе - 12 т.руб.

Таблица 2.3

Сырье	Продукция		Запас сырья
	P_1	P_2	
S_1 - бетон	2	1	20
S_2 - арматура	10	20	200
S_3 - трудозатраты	5	20	100
Прибыль	10	12	

Решение. Обозначим через x_1 и x_2 количество единиц продукции видов p_1 и p_2, планируемое к выпуску. Составим систему ограничений задачи:

$$\begin{cases} 2x_1 + x_2 \le 20, \\ 10x_1 + x_2 \le 200, \\ 5x_1 + 20x_2 \le 100, \\ x_1 \ge 0, \\ x_2 \ge 0. \end{cases}$$

Любое решение (x_1, x_2) системы ограничений называется планом выпуска продукции или планом задачи. Суммарная прибыль от реализации продукции видов p_1 и p_2, выпущенной согласно плану (x_1, x_2), равна

$F(x_1, x_2) = 10x_1+12x_2 \to \max$ – целевая функция.

Построим область решений системы ограничений, которая представляет собой выпуклый многоугольник

1. Строим область решений первого неравенства $2x_1 + x_2 \le 20$

Для этого строим сначала границу области, которая задается уравнением $2x_1+x_2 = 20$. Воспользуемся уравнением прямой в отрезках на осях: $\frac{x_1}{\frac{20}{2}} + \frac{x_2}{20} = 1$, а для определения нужной нам полуплоскости возьмем произвольную точку

плоскости, не лежащую на прямой $2x_1 + x_2 = 20$, например, (0,0), и подставим ее координаты в неравенство: $2 \cdot 0 + 0 \le 20$, т.е. $0 \le 20$, начало координат лежит в полуплоскости решений данного неравенства. На рис. 2.6 это показано стрелкой.

Аналогично построение проводят и для остальных прямых.

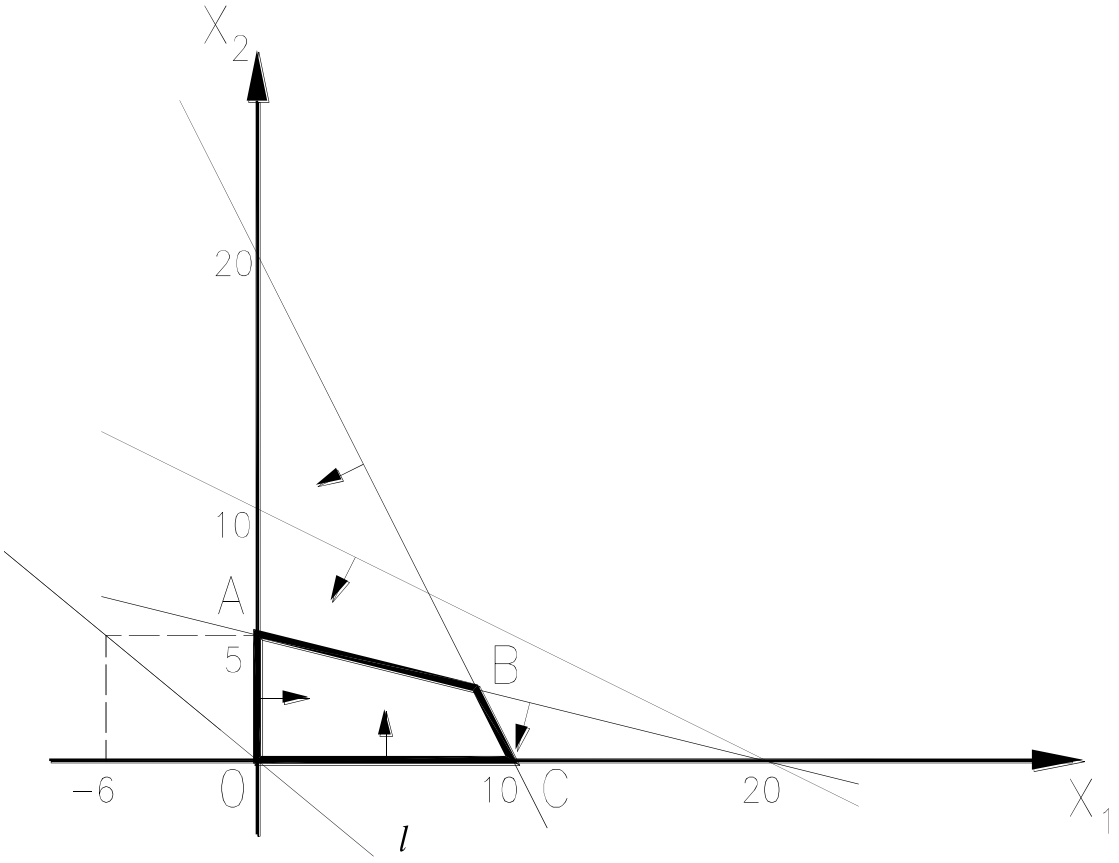


Рисунок 2.6

Таким образом после построения получаем многоугольник OABC.

2. Среди точек многоугольника найдем такую, в которой $F(x_1, x_2)$ принимает максимальное значение. Для этого

построим вектор-градиент, координаты которого $\left(F'_{x_1}, F'_{x_2}\right)$, и проведем прямую, перпендикулярную градиенту (эта прямая является линией нулевого уровня, уравнение которой имеет вид $10x_1 + 12x_2 = 0$, на чертеже – ℓ). Вектор – градиент показывает направление наискорейшего возрастания функции. Будем перемещать прямую l в направлении градиента. Последняя из вершин многоугольника, которую пересекает вектор, и будет искомой. Это вершина B.

3. Найдем координаты вершины B. Для этого решим совместно уравнения прямых, на пересечении которых и стоит эта точка:

$$\begin{cases} 2x_1 + x_2 \le 20, \\ 5x_1 + 20x_2 \le 100. \end{cases}$$

Решая систему, получаем $x_1=60/7$, $x_2=20/7$, тогда $F_{max}=10\cdot 60/7+12\cdot 20/7=120$ (у.е.).

Вывод: Для обеспечения максимальной прибыли от реализации готовой продукции предприятию необходимо выпускать 60/7 единиц продукции вида p_1 и 20/7 единиц продукции вида p_2. При таком плане прибыль от реализации равна 120 т.руб.

Раздел 3 ОСНОВЫ ПРОГРАММИРОВАНИЯ НА *STARK ES*

STARK_ES является программным продуктом семейства программ *MicroFe*, основанным на методе конечных элементов, предназначены для расчета строительных конструкций на рабочих местах.

3.1 Подготовительный этап работы

Прежде чем начать работу с программой необходимо подготовить исходные данные.

Подготовить чертеж расчетной схемы с четкими привязками всех узлов, размерами стержней, указанием типов опорных устройств и их привязку на расчетной схеме. При этом узлами считаются сечения, в которых происходит: изменение нагрузки, изменение размеров поперечного сечения, врезаны шарниры, находятся опорные устройства.

При расчете статически неопределимых конструкций необходимо определить соответствующие поправочные коэффициенты и изменить соответственно «модуль упругости» для соответствующих стержней, оставив неизменными заложенные в программе геометрические характеристики сечения.

Все входные и выходные данные должны соответствовать принятым в системе обозначениям:

Координаты узлов – **[м];**

Площадь поперечного сечения – **[м^2];**

Момент инерции – **[м^4];**

Сила **– [кН];**

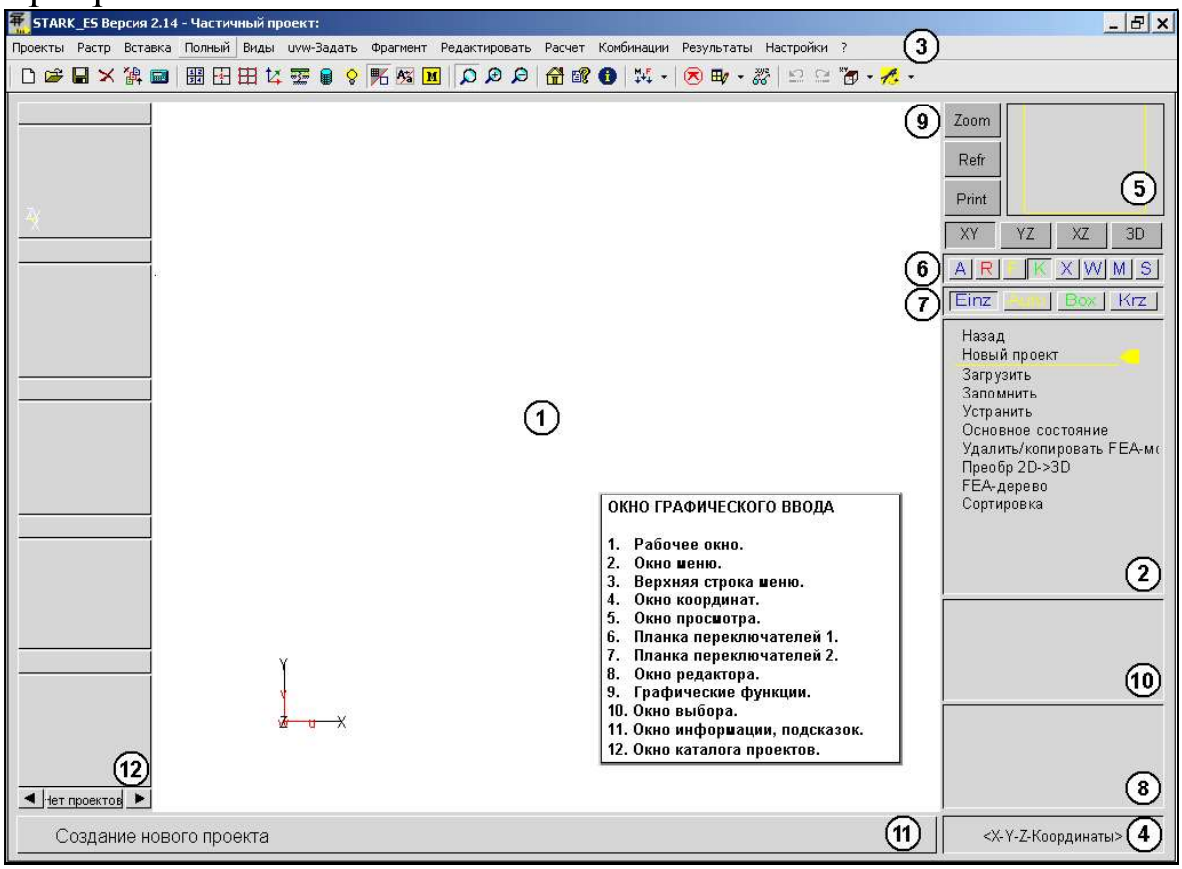
Масса (собственный вес) – **[т];**

Модуль упругости, модуль сдвига – **[кН/м^2];**

Плотность – [**т/м^3**];
Время – [**сек**];
Узловая масса – [**т**];
Перемещения– [**м**]; [**рад**];
Усилия в сечениях балок **– [кН];**
Изгибающие моменты **– [кНм];**
Реакции опор – [**кН**].

3.2 Запуск программы

Запуск программы осуществляется двойным нажатием на пиктограмму *STARK_ES* на рабочем столе или через кнопку *Пуск*. После чего открывается оперативное поле программы:



3.3 Функциональные клавиши

Ниже привиден список и назначение основных функциональных клавиш программы:

Клавиша	Назначение
0	ZOOM-ALL – показ проекта в реальных размерах
1	ZOOM-MOVE – перемещение фрагмента (области) масштабирования вниз налево
2	ZOOM-MOVE– перемещение фрагмента (области) масштабирования вниз
3	ZOOM-MOVE– перемещение фрагмента (области) масштабирования вниз направо
4	ZOOM-MOVE– перемещение фрагмента (области) масштабирования налево
5	REFRESH – перерисовка рабочего окна
6	ZOOM-MOVE– перемещение фрагмента (области) масштабирования направо
7	ZOOM-MOVE– перемещение фрагмента (области) масштабирования вверх налево
8	ZOOM-MOVE– перемещение фрагмента (области) масштабирования наверх
9	ZOOM-MOVE– перемещение фрагмента (области) масштабирования наверх направо
+	ZOOM-IN – размер фрагмента масштабирования уменьшается в два раза относительно центра
-	ZOOM-OUT - размер фрагмента масштабирования увеличивается в два раза относительно центра
[Tab]	Смена актуального частичного проекта: переход к следующему частичному проекту
[Shift+Tab]	Смена актуального частичного проекта: переход к предыдущему частичному проекту
←,↑,→,↓	Вращение объекта (только при изображении проекта в перспективе)
[Пробел]	Переключение между растрами активного частичного проекта

При работе с программой *STARK_ES* выбор того или иного пункта меню осуществляется с помощью левой клавиши мыши, или набором буквенного кода с клавиатуры.

Подтверждение выбора производится щелчком левой клавиши мыши или нажатием клавиши *Enter*.

Необходимо следить за сообщениями, появляющимися в информационном окне. Они содержат указания о дальнейших действиях на текущем этапе работы и позволяют контролировать выполненные действия.

Для отмены последней операции необходимо нажать правую клавишу мыши.

Ввод числовой или текстовой информации осуществляется после появления мигающего курсора с клавиатуры и заканчивается нажатием клавиши *Enter*.

3.4 Редактирование рабочих проектов

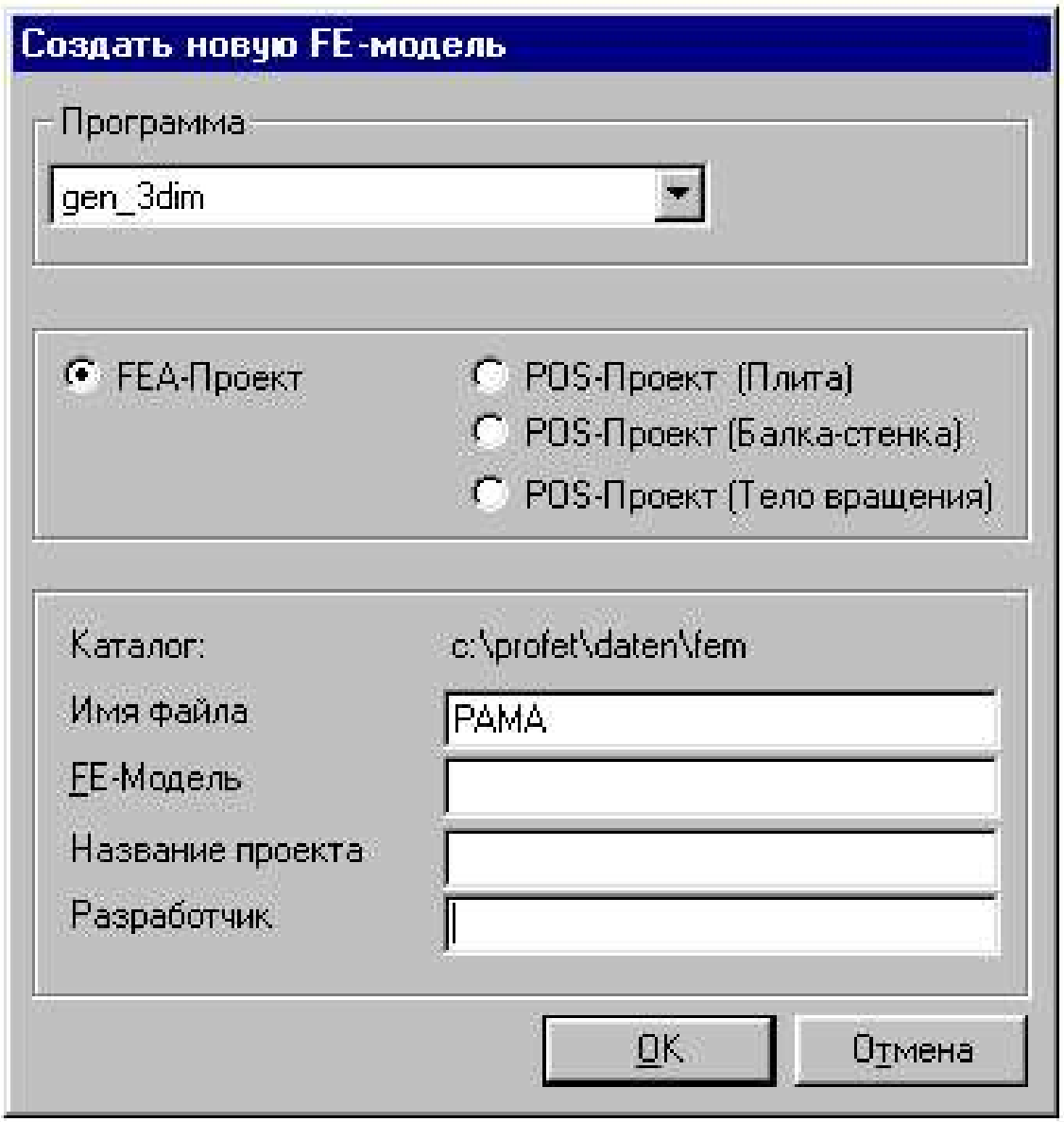
Последовательность выполнения работы на программе *STARK_ES*:

1 Создание документа,
2 Создание сетки растра,
3 Создание расчетной схемы,
4 Ввод свойств материала,
5 Ввод шарниров,
6 Ввод краевых условий,
7 Ввод нагрузок,
8 Сохранение файла,
9 Выполнение расчета,
10 Просмотр результатов расчета.

3.4.1 Создание документа

В окне текстового меню выбрать **Проекты ⇒ Новый проект**.

В открывшемся диалоговом окне в графе *Имя файла* набрать с клавиатуры имя файла - не более 8 символов и без пробелов.



3.4.2 Создание сетки растра

Растры оказывают существенную помощь при вводе расчетных схем конструкции, ориентации и размещении элементов на плоскости.

Одновременно может быть использовано до 10 растров для одного и того же проекта.

В *STARK_ES* предоставляются два вида растров: прямоугольные и полярные. Оба вида имеют только жесткие расстояния между линиями растра.

В окне текстового меню выбрать **Растр ⇒ Определение.**

Тип растра задать **Прям** (Прямоугольный).

Затем нужно определить три точки. Первая точка обозначает начало координат растра, вторая – направление «r» - оси растра, третья – «r-s» - плоскость, в которой будет располагаться растр.

Для прямоугольных растров необходимо затем задать расстояние между линиями растра dr и ds, а также угол поворота линий растра относительно осей координат. Для полярных растров задаются расстояния между окружностями растра dr, углы между радиальными лучами dw, а также поворот растра Alpha относительно оси «z» растра.

Задать координаты первой точки x=0, y=0, z=0.

Координаты второй точки x=1, y=0, z=0.

Координаты третьей точки x=0, y=1, z=0.

Далее задать dx=1, dy=1, dφ=0.

Таким образом, закончена подготовка растра к вводу расчетной схемы. В рабочем окне изображена сетка растра, на сетке – система координат желтого цвета, в информационном окне сообщение: Растр [Начало = {0, 0, 0}, 1 x 1, 0 grad]активен.

В окне графических функций нажать **xy.**

3.4.3 Создание расчетной схемы

Требуется определить усилия в раме изображенной на рис. 3.1.

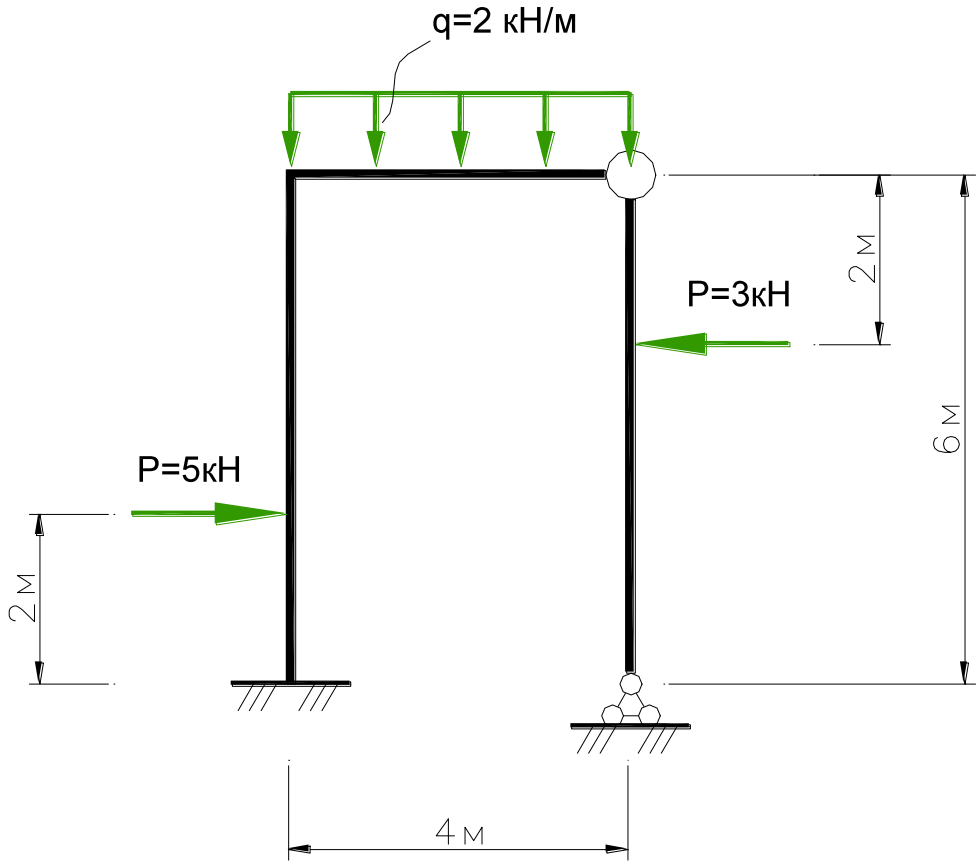
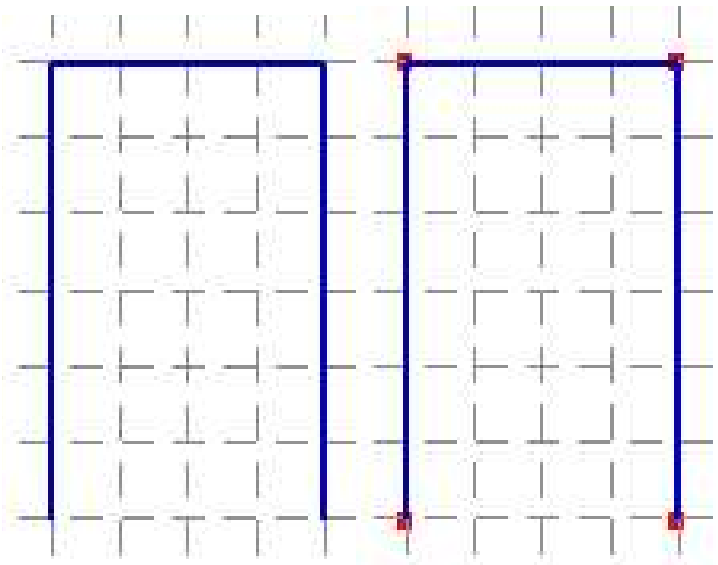


Рисунок 3.1

В окне текстового меню выбрать **Редактировать ⇒ Геометрия ⇒ Элементы ⇒ Установка.**

В окне выбора нажать **[2-D балки].**

Начало и конец элемента указывается курсором по точкам растра.



Для маркировки узлов и вывода на экран номеров узлов, предварительно необходимо в меню выбрать **Узлы** и **Номера узлов.**

Введение узлов может быть проведено делением элементов. Для этого в окне текстового меню выбрать **Редактировать ⇒ Геометрия ⇒ Деление балок.**

В окне выбора установить значение N=…, где N – число промежуточных узлов (на единицу меньше числа элементов). Указать на элементы и нажать **Старт**.

3.4.4 Ввод свойств материала

В окне текстового меню выбрать **Редактировать ⇒ Материал ⇒ Установка.**

В окне выбора нажать **[2-D балки].**

Нажать в окне редактора. После чего в окне редактора появляется сообщение: «материал не определен».

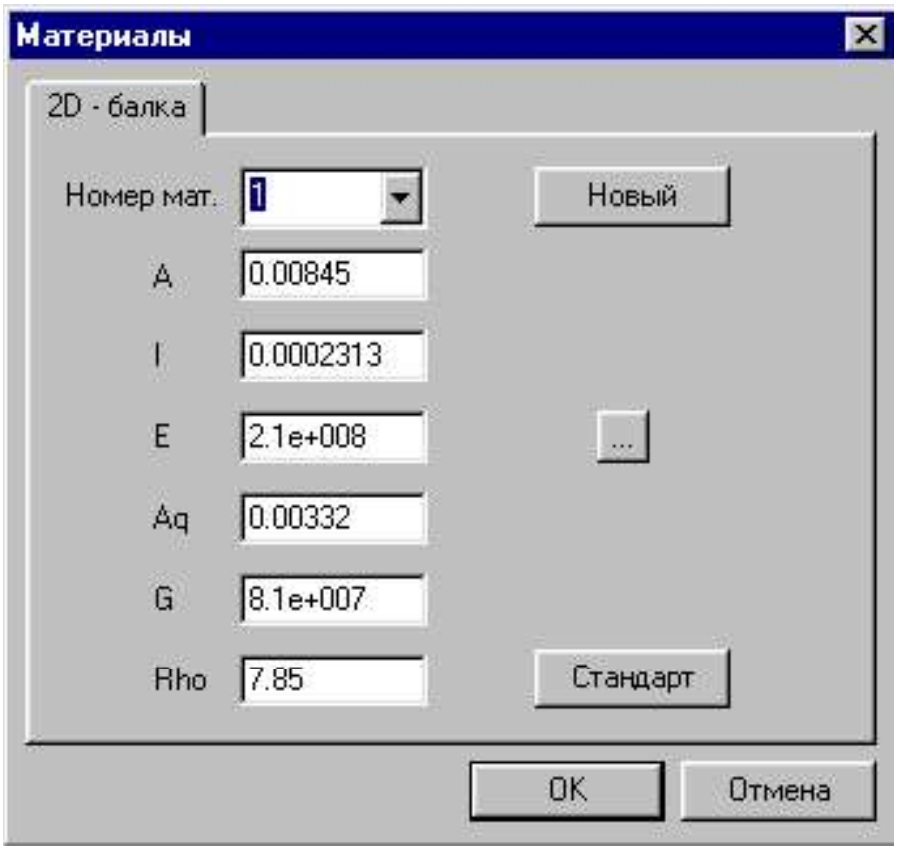
Перевести курсор в окно редактора и щелчком активизировать его.

В центре экрана выпадает панель свойств материала №1, в ячейках которой содержится информация о свойствах конструкционной стали. Вы можете принять или изменить содержимое ячеек в соответствии с особенностями проектируемого материала.

С помощью кнопки  можно вызвать справочную информацию по характеристикам материала (модуль упругости, плотность, модуль сдвига, коэффициент Пуассона) для бетона и стали.

В последней ячейке содержится информация о плотности материала R_{ho}=*7.85*. Если расчет выполняется без учета собственного веса, необходимо изменить содержимое этой ячейки на R_{ho}=**0**.

В окне выбора видов нажать переключатель *[Einz]*, затем перевести курсор на расчетную схему и пометить щелчком левой клавиши каждый из стержней, имеющих введенные свойства материала.



Если содержатся элементы с другими характеристиками материала, необходимо повторить операцию ввода свойств материала 2, 3, 4... и т.д.

Если все элементы имеют одинаковые свойства материала в окне выбора видов нажать переключатель *[Box]*, рам-

кой выбрать все элементы, щелчком левой кнопки подтвердить ввод свойств материала.

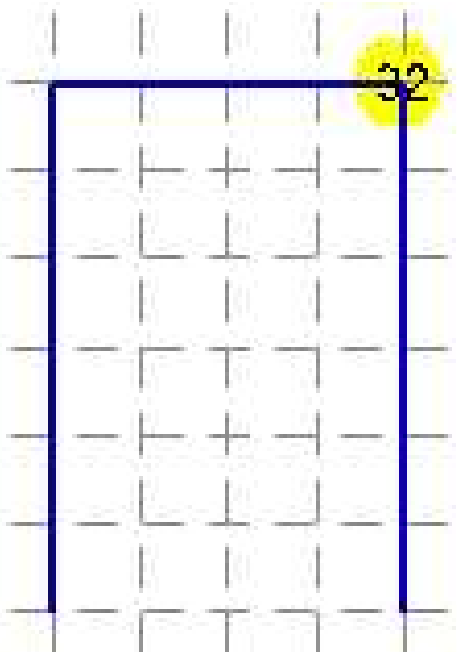
3.4.5 Ввод узловых и элементных шарниров

Если расчетная схема содержит промежуточные шарниры, то в окне текстового меню выбрать **Редактировать ⇒ Графика ⇒ Узловые ⇒ Установка.**

Если расчетная схема содержит элементные шарниры, то в окне текстового меню выбрать **Редактировать ⇒ Графика ⇒ Элементные ⇒ Установка.**

Переключатели первой группы: «X», «Y», «Z», «Rx», «Ry», «Rz» задают степени свободы шарниров («X», «Y», «Z» - перемещения вдоль осей OX, OY и OZ соответственно, «Rx», «Ry», «Rz» - повороты вокруг тех же осей).

Переключатели второй группы: «локал.», «глобал.» и «элем.» задают тип системы координат, в которой задаются шарниры.



Переключатели третьей группы: «Балки» и «Обол.» определяют тип конечных элементов (балочные или оболочные), для которых устанавливаются или удаляются шарниры.

Переключатель четвертой группы: «0», «+1», «-1», «+2» и «-2» задают тип шарниров.

Для узловых шарниров включить **[глобальные], [Rz],** отключить все остальные клавиши и указать курсором на соответствующие узлы, щелчком левой кнопки подтвердить ввод узловых шарниров в указанных узлах.

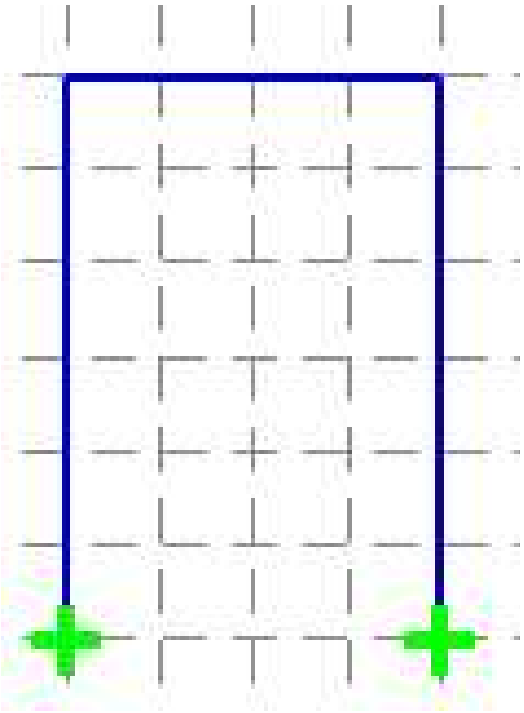
Для элементных шарниров включить **[глобальные], [Rz/t], [Балки]** отключить все остальные клавиши и указать курсором на стержень вблизи расположения элементного шарнира.

В рабочем окне степени свободы шарниров изображаются при помощи кодирования числами. Изображаемое число образуется в результате суммирования для каждой степени свободы шарнира числа 2, возведенного в степень:

0 – для перемещения вдоль OX,
1 - для перемещения вдоль OY,
2 - для перемещения вдоль OZ,
3 – для поворота вокруг OX,
4 - для поворота вокруг OY,
5 - для поворота вокруг OZ.

3.4.6 Ввод краевых условий

В окне текстового меню выбрать **Редактировать ⇒ Кр.усл… ⇒ Краевые условия ⇒ Установка.**



Переключатели первой группы: «X», «Y», «Z», «Rx», «Ry», «Rz» определяют степени свободы, по которым задаются граничные условия.

Переключатели второй группы: «локал.», «глобал.» задают тип системы координат, в которой задаются краевые условия.

Переключатели третьей группы «Раст.», «Сжатие» и «Сж.+Р.» задают тип краевых условий: растяжение, сжатие и жесткое защемление соответственно.

Включить **[глобал.], [$С_Ж$ +Р],** отключить все остальные клавиши.

В окне выбора нажать переключатели, соответствующие тем перемещениям, которым препятствует вводимая связь (для ввода шарнирно-неподвижной опоры нажать [X] и [Y], отжать [Rz]; для ввода шарнирно-подвижной опоры, препятствующей вертикальному перемещению нажать [Y], отжать [X] и [Rz]; для ввода шарнирно-подвижной опоры, препятствующей горизонтальному перемещению нажать [X], отжать [Y] и [R_Z], для ввода жесткой заделки нажать [X], [Y] и [Rz]).

Перевести курсор в узел, и щелчком левой клавиши подтвердить ввод опоры выбранного типа в указанный курсором узел.

3.4.7 Ввод нагрузки

3.4.7.1 Ввод узловой нагрузки

В окне текстового меню выбрать **Редактировать ⇒ Нагрузки ⇒ Узловые ⇒ Узловые нагрузки ⇒ Установка.**

В окне выбора необходимо нажать кнопки, соответствующие виду вводимой узловой нагрузки (для ввода вертикальной нагрузки нажать [Py] и отжать все остальные кнопки, для ввода горизонтальной нагрузки нажать [Px] и отжать все остальные кнопки, для ввода сосредоточенного момента нажать [Mz] и отжать все остальные кнопки).

В окне выбора установить порядковый номер нагружения **N=1**.

В окне редактора установить величину нагрузки, с учетом правила знаков.

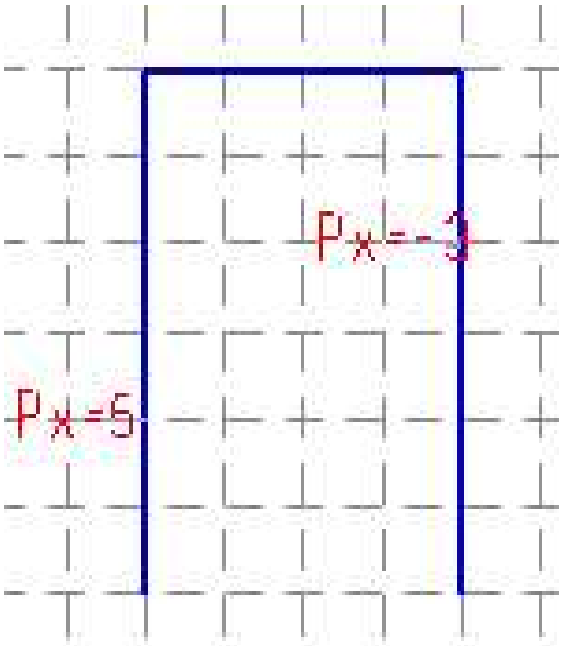
Перевести курсор в рабочее окно, указать им на узел, в котором приложена нагрузка.

3.4.7.2 Ввод элементной нагрузки

3.4.7.2.1 Ввод сосредоточенной нагрузки

В окне текстового меню выбрать **Редактировать ⇒ Нагрузки ⇒ По элементам ⇒ Соср. нагр. ⇒ Установка**, для установки сосредоточенной нагрузки.

В окне выбора необходимо нажать кнопки, соответствующие виду вводимой сосредоточенной нагрузки (для ввода вертикальной нагрузки нажать [Py/s] и отжать все остальные кнопки, для ввода горизонтальной нагрузки нажать [Px/r] и отжать все остальные кнопки, для ввода сосредоточенного момента нажать [Mt] и отжать все остальные кнопки).



Переключатели второй группы: «локал.», «глобал.» задают тип системы координат, в которой задаются сосредоточенные нагрузки.

Включить **[глобал.]**.

В окне выбора установить порядковый номер нагружения **N=1**.

В окне редактора установить величину отношения расстояния до приложения нагрузки ко всей длине участка. Если нагрузка приложена по середине участка, то отношение равно **0.5**.

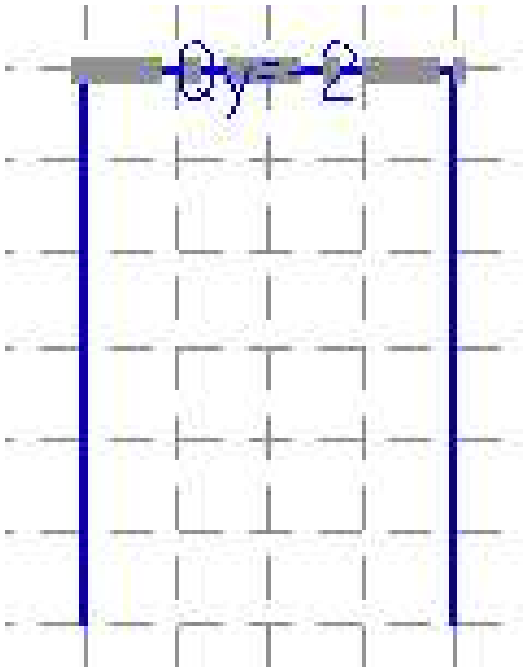
В окне редактора установить величину сосредоточенной нагрузки, с учетом правила знаков.

Перевести курсор в рабочее окно, указать им на элемент, к которому приложена нагрузка.

3.4.7.2.2 Ввод равномернораспределенной нагрузки

В окне текстового меню выбрать **Редактировать ⇒ Нагрузки ⇒ По элементам ⇒ Равн. распр. ⇒ Установка**, для установки равномернораспределенной нагрузки.

В окне выбора необходимо нажать кнопки, соответствующие виду вводимой равномернораспределенной нагрузки (для ввода вертикальной нагрузки нажать [Qy/s] и отжать все остальные кнопки, для ввода горизонтальной нагрузки нажать [Qx/r] и отжать все остальные кнопки).



Переключатели второй группы: «локал.», «глобал.», «проекц.» задают тип системы координат, в которой задаются равномернораспределенные нагрузки.

Включаем **[глобал.]**.

В окне выбора установить порядковый номер нагружения **N=1**.

В окне редактора установить величину равномернораспределенной нагрузки, с учетом правила знаков (Знак «+» ставится если направление нагрузки совпадает с направлением оси, в противном случае ставится знак «-»).

Перевести курсор в рабочее окно, указать им на элемент, к которому приложена нагрузка.

3.4.8 Сохранение информации

В окне текстового меню выбрать **Проекты ⇒ Запомнить**.

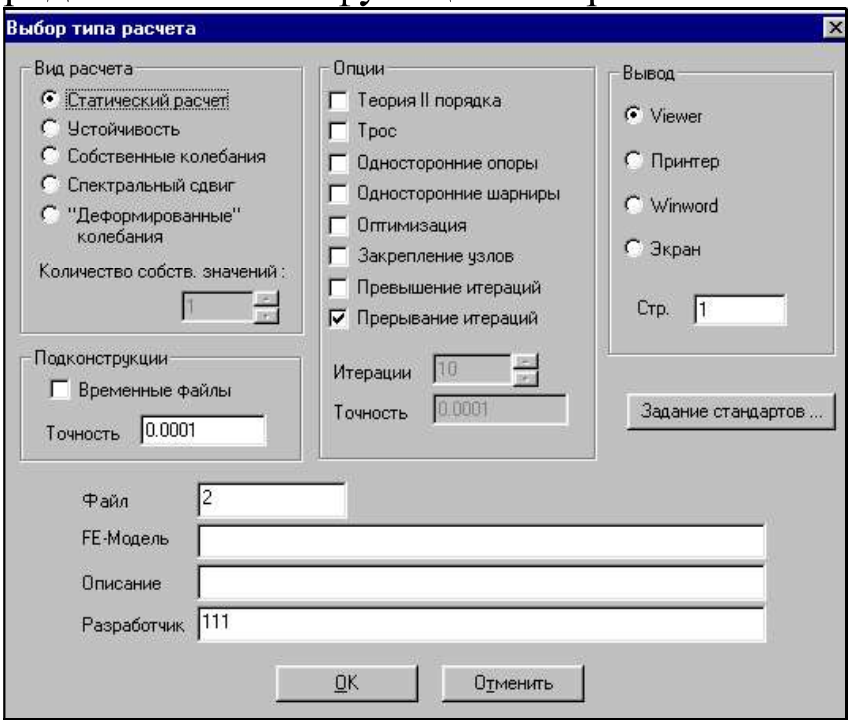
Из представившихся функций выбрать **Сохранить**.

Если необходимо произвести расчет, то необходимо предварительно сохранить все изменения.

3.4.9 Расчет

В окне текстового меню выбрать **Расчет ⇒ Общий**.

Из представившихся функций выбрать **Статический**.



3.4.10 Просмотр результатов расчета

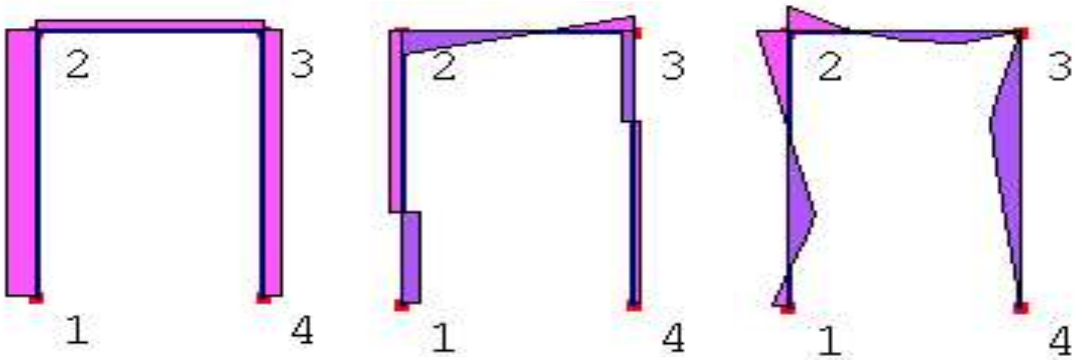
3.4.10.1 Просмотр эпюр N,Q и M

В окне текстового меню выбрать **Результаты ⇒ Усилия ⇒ Усилия в балках**.

Для просмотра эпюры продольных сил N, поперечных сил Q или изгибающих моментов M в окне выбора видов нажать N, Q или M.

Для просмотра величин усилий в стержнях указать курсором на стержень, после щелчка левой клавиши в информационном окне загорается два значения – максимальная и минимальная величина усилия в указанном стержне.

ЭпN(кН) ЭпQ(кН)
ЭпM(кНм)



3.4.10.2 Просмотр значений усилий

В окне текстового меню выбрать **Результаты ⇒ Таблицы ⇒ Усилия ⇒ Усилия в балках**.

В окне выбора нажимаем **Вывести ⇒ Вывести все**.

После нажатие **OK** открывается программа Viewer, содержащая таблицу с промежуточными численными значениями эпюры продольных сил N, поперечных сил Q и изги-

бающих моментов M. Координаты промежуточных сечений отсчитываются от первого узла элемента, которым считается узел с меньшим номером.

Усилия в балках (статический расчет)

El	Lnk	Rs [м]	N [кН]	Q [кН]	M [кНм]
1	1	0.00	-5.07	3.00	-2.27
		1.50	-5.07	3.00	2.23
		2.00	-5.07	3.00	3.73
		2.00	-5.07	-2.00	3.73
		3.00	-5.07	-2.00	1.73
		4.50	-5.07	-2.00	-1.27
		6.00	-5.07	-2.00	-4.27
2	1	0.00	-2.00	5.07	-4.27
		1.00	-2.00	3.07	-0.20
		2.00	-2.00	1.07	1.86
		3.00	-2.00	-0.93	1.93
		4.00	-2.00	-2.93	0.00
3	1	0.00	-2.93	2.00	0.00
		1.50	-2.93	2.00	3.00
		2.00	-2.93	2.00	4.00
		2.00	-2.93	-1.00	4.00
		3.00	-2.93	-1.00	3.00
		4.50	-2.93	-1.00	1.50
		6.00	-2.93	-1.00	0.00

ПРИЛОЖЕНИЕ

Варианты исходных данных.

Номер варианта выдается преподавателем индивидуально.

Задание 1. Определить методами дихотомии и простой итерации корни нелинейного уравнения с одной неизвестной. Уравнение задано в виде u(x)=v(x). Вид функций взять из табл. 1 по варианту.

Таблица 1

1-ая цифра шифра	1	2	3	4	5	6	7	8	9	0
u(x)	$-x^3$	-cos(x)	-cos(2x)	cos2(x-2)	-lg(x+3)	$-e^x$	5sin(x)	3sin(3x)	e^{-x}	-(x+7)
2-ая цифра шифра	1	2	3	4	5	6	7	8	9	0
v(x)	$-x^2+1$	$-x^2+2$	ln(x)+1	2ln(x)+1	2x+0.5	x-4	2ln(x-1)	$2x^4-2.5$	x^2-7	0.5x-2

Задание 2. Методами Гаусса и Зейделя решить систему трех уравнений. Коэффициенты расширенной матрицы принять по варианту из табл. 2.

Таблица 2

1-ая цифра шифра	Коэффициенты матрицы									2-ая цифра шифра	Свободные члены		
	a_{11}	a_{12}	a_{13}	a_{21}	a_{22}	a_{23}	a_{31}	a_{32}	a_{33}		a_{14}	a_{24}	a_{34}
1	0,21	-0,45	-0,20	0,30	0,25	0,43	0,60	-0,35	-0,25	1	1,91	0,32	1,83
2	-3	0,5	0,5	0,5	-6	0,5	0,5	0,5	-3	2	-56,5	-100	-210
3	0,63	0,05	0,15	0,15	0,10	0,71	0,03	0,34	0,10	3	0,34	0,42	0,32
4	-0,20	1,60	-0,10	-0,30	0,10	-1,50	1,20	-0,20	0,30	4	0,30	0,40	-0,60
5	0,20	0,44	0,81	0,58	-0,29	0,05	0,05	0,34	0,10	5	0,71	0,02	0,32
6	-9,11	1,02	-0,73	7,61	6,25	-2,32	-4,64	1,13	-8,88	6	-1,25	2,33	-3,75
7	0,06	0,92	0,03	0,99	0,01	0,07	1,01	0,02	0,99	7	-0,82	0,66	-0,48
8	0,10	-0,07	-0,96	0,04	-0,99	-0,85	0,91	1,04	0,19	8	-2,04	-3,73	-1,67
9	0,62	0,81	0,77	0,03	-1,11	-1,08	0,97	0,02	-1,08	9	-8,18	0,08	0,06
0	0,63	-0,37	1,76	0,90	0,99	0,05	0,13	-0,95	0,69	0	-9,29	0,12	0,69

Задание 3. Для функции заданной таблично, подобрать интерполяционный многочлен Лагранжа. Уплотнить таблицу, вычислив значения в середине интервалов. Аппроксимировать заданную функцию по методу наименьших квадратов, используя несколько эмпирических формул и выбрать из них наилучшую. Данные взять из табл. 3 по варианту.

Таблица 3

1-ая цифра шифра	Значения независимой переменной			2-ая цифра шифра	Значения функции		
	X_0	X_1	X_2		Y_0	Y_1	Y_2
1	0	1	2	1	0.09	0.8	1.9
2	0	2	4	2	0.38	3.5	7.8
3	0	3	6	3	0.85	8.7	18.4
4	0	4	8	4	1.5	15	33
5	0	5	10	5	2.6	24	51
6	0	6	12	6	3.7	35	73
7	0	7	14	7	4.8	50	95
8	0	8	16	8	0.6	7	10
9	0	9	18	9	0.8	8.5	15.8
0	0	10	20	0	1.1	10.5	19

Задание 4. Составить план выпуска продукции, при котором прибыль от реализации всей продукции была бы максимальной. Данные взять из табл. 4 по варианту.

Таблица 4

1-ая цифра шифра	Расход материала на 1 единицу изделия						2-ая цифра шифра	Ресурсы материала		
	a_{11}	a_{12}	a_{21}	a_{22}	a_{31}	a_{32}		b_1	b_2	b_3
1	2	1	10	20	5	20	1	20	160	100
2	4	2	12	18	10	12	2	30	180	80
3	2.5	1	14	10	6	10	3	25	160	80
4	3	1	15	30	5	25	4	30	200	100
5	2	1.5	10	15	6	18	5	20	180	90
6	3	2	10	20	5	20	6	25	200	80
7	2	2	10	12	5	20	7	30	160	100
8	3	1	10	16	10	20	8	25	180	100
9	3	2	20	30	5	10	9	30	160	80
0	2	1	12	18	5	20	0	20	200	90

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Л.И. Турчак. Основы численных методов. Учебное пособие. – М: Наука. Гл. ред. физ-мат. лит. 1987. – 320с.

2 Н.Н. Калиткин. Численные методы. – М: Наука. 1978г.