Применение динамических массивов в структурном подходе

Универсальное и безопасное выделение памяти под динамический массив

```
int n, m;
int *a = new int[n];
cout << “Input number of rows and columns: “;
cin>> n>>m;
int **a=new int *[n]; // выделяется память под массив указателей на строки массива (n).
for (int i = 0;  I < n;  i++) //цикл для выделения памяти под каждую строку массива.
    a[ i ] = new int [m]; //каждому элементу массива указателей на строки присваивается адрес начала участка памяти, выделенного под строку двумерного массива.
```

1. Создать программу для перемножения матрицы на вектор. Данные о размерах матрицы и вектора, об их элементах ввести с клавиатуры.

СОСТАВИМ КОНТРОЛЬНЫЙ ПРИМЕР:

$$\text{Пусть } A=\begin{pmatrix} -2 & 1 & 0 & 2 \\ 3 & 4 & -3 & 7 \\ 5 & 6 & 3 & -1 \end{pmatrix} \text{ и } B=\begin{pmatrix} 2 \\ 0 \\ 1 \\ -1 \end{pmatrix}, \text{ тогда}$$

$$A\cdot B=\begin{pmatrix} -4+0+0-2 \\ 6+0-3-7 \\ 10+0+3+1 \end{pmatrix}=\begin{pmatrix} -6 \\ -4 \\ 14 \end{pmatrix}$$

Словесный алгоритм задачи

1. Составить программу ввода и инициализации вектора
2. Составить программу ввода и инициализации матрицы
3. Составить программу перемножения матрицы на вектор
4. Составить программы вывода на консоль вектора и матрицы.

Прототипы обозначенных функций

```
#include <iostream>
using namespace std;
void InputArray(int*& a, int& n);// выделение
памяти и инициализация одномерного массива
void InputArray2(int**& a, int& n, int& m);//
выделение памяти и инициализация двумерного
массива
void PrintArray(int* a, int n);// печать
одномерного массива
void PrintArray2(int** a, int n, int m);
//печать двумерного массива
void mulply(int **a,int n,int m,int *b,int*& c,
int &nc);// умножение матрицы a на вектор b,
результат в c
```

Главная программа

```
int main()
{
   setlocale(0, "");//оператор для поддержки
кириллицы в консоли Windows
int **a, n, ma, mb;
InputArray2(a, n, ma);
PrintArray2(a, n, ma);

int *b;
InputArray(b, mb);
if (mb!=ma){
   cout<<"размер вектора не равен числу столбцов
матрицы. Перезапустить задачу!";
   system("pause");
   return 1;
}
```

Главная программа (продолжение)

```
PrintArray(b, mb);

int *c, nc;
mulply(a,n, ma,b,c,nc);

cout<<"\n Результат умножения\n";
PrintArray(c, n);

system("pause");
      return 0;
}
```

Коды объявленных функций

```
void InputArray(int*& a, int& n)
{
cout << "Введите размер вектора: "; cin
>> n;
a = new int[n];
cout << "Введите элементы вектора: ";
for (int i = 0; i < n; ++i) cin >> a[i];
return;
}
```

Коды объявленных функций

```
void InputArray2(int**& a, int& n, int& m)
{
cout << "Введите количество строк матрицы: ";
cin >> n;
cout << "Введите количество столбцов матрицы:
"; cin >> m;
a = new int*[n];
cout << "Введите элементы матрицы:" << endl;
for (int i = 0; i < n; ++i)
{
a[i] = new int[m];
for (int j = 0; j < m; ++j) cin >> a[i][j];
}
}
```

Коды объявленных функций

```
void PrintArray(int* a, int n)
{
cout << "Элементы вектора: ";
while (n-- > 0) cout << *a++ << '\t';
cout << endl;
}
```

```
void PrintArray2(int** a, int n, int m)
{
cout << "Элементы матрицы:"
for(int i=0;i<n;i++)
{cout<<endl;
for(int j=0;j<m;j++)
cout<< '\t'<<a[i][j];
}
cout << endl;
}
```

В данной программе указатель "разрушается", то есть адрес, который в нем содержится, постоянно изменяется и после выполнения функции он становится непригодным для использования, поскольку ссылается на область памяти, уже не принадлежащую данному массиву.

Однако это – *копия реального указателя*, который был передан из вызывающей программы: то же самое и с параметром n, ведь и указатель, и целая переменная - это всё числа, в конечном счёте, и *они копируются при сопоставлении параметров*. То есть, в принципе, они уже самостоятельны и изолированы от вызывающей программы.

Коды объявленных функций

```
void mulply(int **a,int  n, int m,int *b,
int*& c, int& nc)
{
   c = new int[n];
   for(int i=0; i<n; i++)
   {
      int s=0;
   for (int j=0;j<m;j++)
      s+=a[i][j]*b[j];
   c[i]=s;
   }
   nc=n;
}
```

```
Введите количество строк матрицы: 3
Введите количество столбцов матрицы: 4
Введите элементы матрицы:
-2 1 0 2
3 4 -3 7
5 6 3 -1
Элементы матрицы:
        -2      1       0       2
        3       4       -3      7
        5       6       3       -1
Введите размер вектора: 4
Введите элементы вектора: 2 0 1 -1
Элементы вектора: 2     0       1       -1

 Результат умножения
Элементы вектора: -6    -4      14
Для продолжения нажмите любую клавишу . . .
```

2. Создать программу для проверки коллинеарности двух *n*-мерных векторов.

Два вектора коллинеарны (параллельны), если *отношения их координат равны*.

Допустим, есть два вектора $\vec{x} = \{x_1, x_2, \ldots, x_n\}$ и $\vec{y} = \{y_1, y_2, \ldots, y_n\}$. Тогда, если выполняется условие:

$$\frac{x_1}{y_1} = \frac{x_2}{y_2} = \cdots = \frac{x_n}{y_n} = const$$

то вектора коллинеарны, в противном случае не коллинеарны.

Формальное применение данного определения коллинеарности имеет *существенный минус:* условие неприменимо, если одна из компонент вектора, или даже обе, равны нулю. Этот вариант необходимо предусмотреть: запрограммировать прерывание с соответствующим сообщением.

Вместо словесного алгоритма прототипы функций

```
#include <iostream>
#include <iomanip>
#include <ctime>
#include <windows.h>

using namespace std;

void vector(int *x,int n, int mn, int mx);
//инициализация вектора случайными числами

void print_vect(int *x, int n);// печать вектора

bool check(int*x, int*y, int n);// проверка
коллинеарности векторов
```

Главная программа

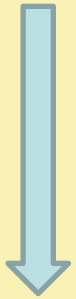
```
int main()
{
   int n;// размер векторов
      bool result;
   cout<<"input vector size:"; cin>>n;
      int *a= new int[n];
      int *b= new int[n];

   vector (a,n,-5,5);print_vect(a, n);
    vector(b,n,-5,5);print_vect(b, n);
   result= check(a,b,n);

      if(result) cout<<"\n vectors are colinear\n";
      else cout<<"\n vectors are non colinear\n";

      system ("pause");
      return 0;
}
```

Коды объявленных функций

```
void vector(int *x,int nx, int mn, int mx)
{
   srand((unsigned int)time(0));
   sleep(1000);//приостановка программы на 1000
миллисекунд
   for(int i=0; i<nx; i++)
   {
      x[i] = rand() % (mx - mn + 1) + mn;
      if(x[i] == 0)
      {cout<<"\n The vector contains a zero
component, \n then the method is not
applicable\n";
   system("pause"); exit(1);
      }
   }
   return;
}
```

Коды объявленных функций

```
bool check(int*x, int*y,int n)
{
float rat=(float)x[1]/y[1];bool res=1;
for(int i=1;i<n;i++)
if((float)x[i]/float(y[i])!=rat)
{res=0.;break;}
return res;
}
```



```
if(abs((float)x[i]/float(y[i])-rat)>1e-6)

1e-6=0,000001
```

Результаты работы программы исчерпывают **проверку полноты решения:**

```
input vector size:3

 The vector contains a zero component,
 then the method is not applicable
Для продолжения нажмите любую клавишу . . .
```

```
input vector size:3
   -2   -5    4

   -2   -5    4

 vectors are colinear
Для продолжения нажмите любую клавишу . . .
```

```
input vector size:5
   -1   -1   -5   -2   -3

 The vector contains a zero component,
 then the method is not applicable
Для продолжения нажмите любую клавишу . . .
```

```
input vector size:3
   -1   -2   -3

    2   -4    2

 vectors are non colinear
Для продолжения нажмите любую клавишу . .
```

3. Изменение энергопотребления жилого дома по дням определяется формулой:

$$P(d) = \left|\sin\left(e^{\cos d} + \varphi\right)\right|, \varphi = 3{,}1 \cdot 10^{-2}$$

Электронный регистратор записывает в свою память значения потребляемой энергии каждые ***n*** дней.

По истечении ***N*** дней созданный массив значений обрабатывается и определяются дни, когда потребление энергии максимально и минимально.

Значения ***n*** и ***N*** настраиваемые.

Смоделируйте данный процесс.

Словесный алгоритм задачи

1. Разработать функцию P(d): $P(d) = \left|\sin\left(e^{\cos d} + \varphi\right)\right|, \varphi = 3{,}1 \cdot 10^{-2}$
2. Определить размер np динамического массива, в котором будут храниться измерения P(d) начиная с d=1 до d=N с шагом n: np=(N-0)/n=N/n.
3. Создать функцию **`create_array`**, которая будет хранить измерения в массиве PP[np] вычисленные измерения на текущий день d.
4. Создать функцию **`min_max`**, которая найдёт в массиве PP[np] индексы максимального и минимального элемента.
5. Разработать функцию вывода массива PP[np], аргументы которого в днях измеряются с шагом n.

Разбиваем задачу на несколько функций, назначение которых прописано в комментариях к прототипам.

```
#include <iostream>
#include <iomanip>
#include<cmath>
using namespace std;
float p(int d);//изменение энергии p от d
void create_array(float* u,int nu, int
n);//создание массива измерений
void min_max(float *u, int n, int &min_i,
int &max_i);//определение индексов для
min\max
void print(float *x, int nx, int n);//печать
массива
```

```
int main()
{int n, N, np;
int i_mn, i_mx;
cout<<"input interval n:"; cin>>n;
cout<<"input period N:"; cin>>N;
 np=N/n+1;//вычисление числа точек на
интервале изменений
float *PP= new float[np];
create_array(PP,np,n);
print(PP,np,n);
min_max(PP,np,i_mn,i_mx);
cout << "\n min = "<<PP[i_mn] <<" is
reached at "<<i_mn*n<<" day\n";
cout << "\n max = "<<PP[i_mx] <<" is
reached at "<<i_mx*n<<" day\n";
system ("pause");
return 0;}
```

```
void create_array(float* u,int nu,int n)
{
   for(int i=0;i<nu;i++)
      u[i]=p(n*i);
return;
}


float p(int d)
{float phi=3.1e-2;
return abs(sin(exp((double)d)+phi));
}
```

```
void min_max(float *u, int nu,int & min_i,int & max_i)
{
   float umax,umin;
   umax = umin = u[0]; min_i = max_i = 0;
for(int i=0;i<nu;i++)
{
   if(u[i]<umin){umin=u[i];min_i=i;}
   if(u[i]>umax){umax=u[i];max_i=i;}
}
return;
}
  void print(float  *tf,   int  nf, int n)
  {
     cout << endl;
     for(int i=0; i<nf; i++)
        cout <<'\t' << n*i << '\t'<<tf[i]<<endl;
       cout << endl ;
     return;
  }
```

Результаты

```
input interval n:2
input period N:14

        0       0.857813
        2       0.907322
        4       0.93981
        6       0.972546
        8       0.373186
        10      0.710846
        12      0.995261
        14      0.474347


 min = 0.373186 is reached at 8 day

 max = 0.995261 is reached at 12 day
Для продолжения нажмите любую клавишу . .
```

```
input interval n:5
input period N:50

        0       0.857813
        5       0.709863
        10      0.710846
        15      0.983654
        20      0.428888
        25      0.664902
        30      0.105911
        35      0.174314
        40      0.131219
        45      0.861768
        50      0.607714


 min = 0.105911 is reached at 30 day

 max = 0.983654 is reached at 15 day
Для продолжения нажмите любую клавишу . . .
```

4. Разработать программу-функцию для вычисления функции

$$Z(x) = 1 - \frac{1}{3}x + \frac{1\cdot4}{3\cdot6}x^2 - \frac{1\cdot4\cdot7}{3\cdot6\cdot9}x^3 + \frac{1\cdot4\cdot7\cdot10}{3\cdot6\cdot9\cdot12}x^4 + \cdots$$

для любого |x|≤1 до члена, который по абсолютному значению будет давать вклад меньший некоторого наперёд заданного ε <<1. Использовать рекуррентные соотношения. Рассчитать значения функции на интервале x∈[-1;1] с шагом Dx=0,2 и занести их в массив, в котором найти значение, наиболее близкое к заданному числу F

```
float z(float x, float eps);// вычисление функции по заданному
                            ряду
int search_ind(float a[ ], int n, float f); // поиск индекса элемента
                            массива, наиболее близкого к F
void array_z(float a[ ], float xn, float xk, float dx, float e);
// иниализация массива значениями функции на
//заданном интервале.
```

$$\cdot Z(x) = 1 - \frac{1}{3}x + \frac{1\cdot 4}{3\cdot 6}x^2 - \frac{1\cdot 4\cdot 7}{3\cdot 6\cdot 9}x^3 + \frac{1\cdot 4\cdot 7\cdot 10}{3\cdot 6\cdot 9\cdot 12}x^4 + \cdots$$

```
float z(float x, float eps)
{
    float ch = 1, cha = -2, zn = 1, zna = 0, u=1, s = 1;
    do {
        cha = cha + 3; ch *= x * cha;
        zna += 3; zn *= zna;
        u = -u * ch / zn;
        s += u;
    } while (abs(u) > eps);
    return s;
}
```

i=1	Cha=−2+3=1	Zna=0+3=3	ch=1*x	Zn=1*3=3	U=−x/3
i=2	Cha=1+3=4	Zna=3+3=6	ch=1*4*x*x	Zn=3*6	U=1*4*x*x/(3*6)
i=3	Cha=4+3=7	Zna=6+3=9	ch=1*4*7*x*x*x	Zn=3*6*9	U=−1*4*7*x*x*x/(3*6*9)

```
void array_z(float a[], float xn, float xk, float dx, float e)
{
    int i = 0, n = int((xk - xn) / dx) + 1;
    for (float x = xn; i < n;  i++,  x += dx)
        a[i] = z(x, e);
}
```

```
int search_ind(float a[], int n, float f)
{
    float df = abs(f - a[0]); int ind = 0;
    for (int i = 1; i < n; i++)
        if (abs(f - a[i]) < df) { df = abs(f - a[i]); ind = i; }
    return ind;
}
void print(float a[], int n)
{
    cout << endl;
    for (int i = 0; i < n; i++)
        printf("a[%d] = %4.2f", i, a[i]);
    cout << endl;
}
```

```
void print(float *a, int n);  // печать массива
int main()
{
float xn = -0.4, xk = 0.4, dx = 0.2, eps = 0.001, f;
int  n = int((xk - xn) / dx) + 1;
int index;
float *a_z=new int [n];
```

Блок описания и инициализации переменных

```
array_z(a_z, xn, xk, dx, eps);
print(a_z, n);
```

Блок инициализации массива – Производство 1

```
cout << "\ninput f : "; cin >> f;
index = search_ind(a_z, n, f);
```

Производство 2

```
cout << "\n element a[" << index << "] = " << a_z[index] << " is the
closest to f = " << f << endl;
system("pause");
return 0;
}
```

Блок вывода результатов

```
a[0] = 1.13a[1] = 1.07a[2] = 1.00a[3] = 0.93a[4] = 0.87

input f : 1.035

 element a[1] = 1.06607 is the closest to f = 1.035
Для продолжения нажмите любую клавишу . . .
```

```
a[0] = 1.13a[1] = 1.07a[2] = 1.00a[3] = 0.93a[4] = 0.87

input f : 1.12

 element a[0] = 1.12854 is the closest to f = 1.12
Для продолжения нажмите любую клавишу . . .
```

Лекция закончена