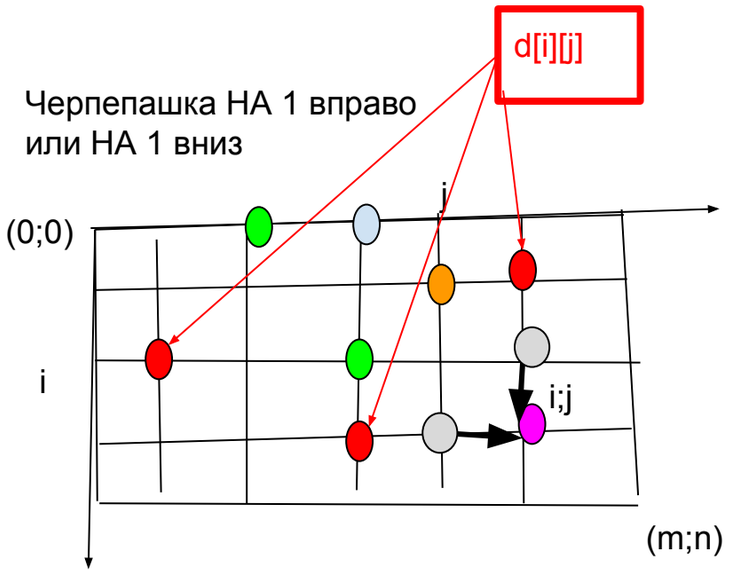
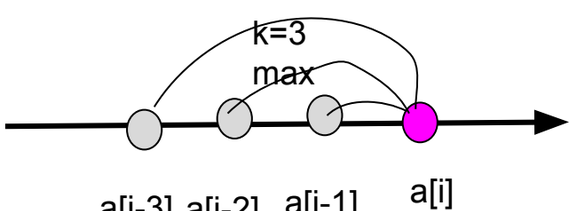
Черепашка монетки





56	45						
	65	36					
		663	663	636	46		
					34	64	66

1.Какие значения мы вычисляем a[i][j] - максимальное количество монет: которые может иметь черепашка, допрыгав до [i][j]-ого места

2.Какое рекурсивное соотношение

было для кузнечика

a[i]=МАКС[j=1;min(k,i)] (a[i-j]) + d[i]

это max из k предыдущих, если они есть иначе это max стольких предыдущих, сколько есть

и ПЛЮС текущая божья коровка

стало черепашки

a[i][j]=MAX(a[i][j-1],a[i-1][j])+d[i][j]

3.Какие начальные значения

a[0][0]=1

4.В каком порядке вычисляются значения

по строкам, затем по столбцам

5.Где искать ответ a[m][n]

Запомнить маршрут from[i][j] - откуда прийти на [i][j]-ую клетку, чтобы максимизировать число монет

from[i][j]=1, если пришли сверху

from[i][j]=2, если пришли слева

```
void cherpepashka_gopniki_volantery(int m, int n)
{
   int **d=new int* [m+1];
   for(int i=0;i<m+1;i++)
   {
      d[i]=new int [n+1];
   }
   int **course=new int* [m+1];
   for(int i=0;i<m+1;i++)
   {
      course[i]=new int [n+1];
   }

   int **a=new int* [m+1];
   for(int i=0;i<m+1;i++)
   {
      a[i]=new int [n+1];
   }
   int **from=new int* [m+1];
   for(int i=0;i<m+1;i++)
   {
      from[i]=new int [n+1];
   }
   mat_zero(from,m+1,n+1);
   mat_zero(a,m+1,n+1);
   mat_zero(course,m+1,n+1);
   mat_rand_money(d,m+1,n+1);
   print_mat(d,m+1,n+1);
   a[0][0]=1+d[0][0];
   for(int i=0;i<=m;i++)
   {
      for(int j=0;j<=n;j++)
      {
         if(j>0)
         {
            if(i==0)
            {
               a[i][j]=a[i][j-1]+d[i][j];
               //from[i][j]=a[i][j-1];
               from[i][j]=2;
            }
            else
            {
               a[i][j]=max(a[i-1][j],a[i][j-1])+d[i][j];
               //from[i][j]=max(a[i-1][j],a[i][j-1]);
               from[i][j]=(a[i-1][j]>=a[i][j-1])?(1):(2);
            }
         }
         else if(i>0)
         {
            a[i][j]=a[i-1][j]+d[i][j];
            //from[i][j]=a[i-1][j];
            from[i][j]=1;
         }
      }
   }
   print_mat(a,m+1,n+1);
   //print_mat(from,m+1,n+1);
   print_turtle_from(from,course,m,n);
   print_mat(course,m+1,n+1);
   cout<<"a[m][n]= "<<a[m][n]<<endl;
}
```

```
void mat_zero(int **mass, int y, int x)
{
   for(int i=0;i<y;i++)
   {
      for(int j=0;j<x;j++)
      {
         mass[i][j]=0;
      }
   }
}
void mat_rand(int **mass, int y, int x)
{
   for(int i=0;i<y;i++)
   {
      for(int j=0;j<x;j++)
      {
         if(rand()%10==0)
         {
            mass[i][j]=1;
         }
         else
         {
            mass[i][j]=0;
         }

      }
   }
}
void mat_rand_money(int **mass, int y, int x)
{
   for(int i=0;i<y;i++)
   {
      for(int j=0;j<x;j++)
      {
         if(rand()%2==0)
         {
            mass[i][j]=rand()%10;
         }
         else
         {
            mass[i][j]=-rand()%10;
         }
      }
   }
}
void print_turtle_from(int **from, int **course, int i, int j)
{
   if(i==0 && j==0)
   {
      //cout<<"0,0"<<endl;
      course[i][j]=1;
   }
   else
   {
      //cout<<i<<","<<j<<endl;
      course[i][j]=1;
      if(from[i][j]==1)
      {
         //cout<<i-1<<","<<j<<endl;
         print_turtle_from(from,course,i-1,j);
      }
      else if(from[i][j]==2)
      {
         //cout<<i<<","<<j-1<<endl;
         print_turtle_from(from,course,i,j-1);
      }
   }
}
```