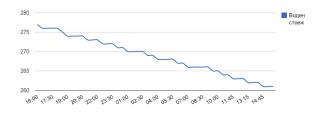
# Рекурсивни програми

21 декември 2016 г.

Редици, индуктивни дефиниции, индукция, рекурсия

# Какво е редица от числа?

#### • Серия от измервания

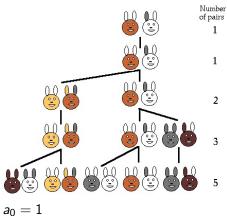


Фигура: Нивото на река Дунав в см.

$$a_0 = 280cm \mid a_1 = 275cm \mid a_2 = 271cm \mid a_3 = 272cm \mid ...$$



# Описание на феномен?

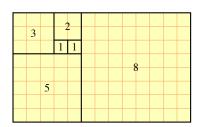


$$a_0 = 1$$
 $a_1 = 1$ 
 $a_{i+2} = a_{i+1} + a_i$ 

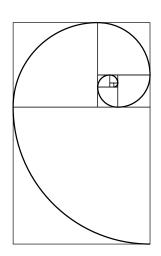


Leonardo Fibonacci (c. 1170 – c. 1250)

## Изчисление?



$$a_0 = 1$$
  
 $a_1 = 1$   
 $a_{i+2} = a_{i+1} + a_i$ 



# Явна vs. индуктивна дефиниция на елементите на редица

0, 2, 6, 12, 20, 30, ...

• явна дефиниция

$$\{a_i\}_{\infty}^{i=0}$$

$$a_i = 2 + 4 + \dots + 2i = \sum_{k=1,\dots,i} 2k$$

• индуктивна дефиниция

$$\begin{aligned}
\left\{a_i\right\}_{\infty}^{i=0} \\
a_0 &= 0 \\
a_i &= a_{i-1} + 2i
\end{aligned}$$

# Явна vs. индуктивна дефиниция на елементите на редица

$$0, 2, 6, 12, 20, 30, \dots$$

• явна дефиниция

$$\{a_i\}_{\infty}^{i=0}$$

$$a_i = 2 + 4 + \dots + 2i = \sum_{k=1,\dots,i} 2k$$

• индуктивна дефиниция

$$\begin{aligned}
\left\{a_i\right\}_{\infty}^{i=0} \\
a_0 &= 0 \\
a_i &= a_{i-1} + 2i
\end{aligned}$$

# Явна vs. индуктивна дефиниция на елементите на редица

$$0, 2, 6, 12, 20, 30, \dots$$

• явна дефиниция

$${a_i}_{\infty}^{i=0}$$

$$a_i = 2 + 4 + \dots + 2i = \sum_{k=1,\dots,i} 2k$$

• индуктивна дефиниция

$$\begin{aligned}
\left\{a_{i}\right\}_{\infty}^{i=0} \\
a_{0} &= 0 \\
a_{i} &= a_{i-1} + 2i
\end{aligned}$$

# От дефиниция до програма: наивен подход

```
void print_first_n (int n)
{
  for (int i = 1; i <= n; i++) {
    int sum = 0;
    for (int k = 1; k <= i; k++)
    //calculate 2+4+..+2*k
    {
       sum = sum + 2*k;
    }
    cout << "a[" << i << "]="
       << sum << endl;
  }
}</pre>
```

$$\begin{aligned}
 \left\{a_i\right\}_{\infty}^{i=0} \\
 a_i &= 2+4+\ldots+2i = \sum_{k=1,\ldots,i} 2k \\
 a_0 &= 0 \\
 a_1 &= 0+2 \\
 a_2 &= 0+2+4 \\
 a_3 &= 0+2+4+6
\end{aligned}$$

. . .

## Използваме връзката между членовете на редицата

```
void print_first_n (int n)
                                                        \{a_i\}_{\infty}^{i=0}
  int a_i = 0;
                                                        a_i = a_{i-1} + 2i
  for (int i = 1; i <= n; i++)
    cout << "a[" << i << "]=" << a_i << endl;
                                                        a_0 = 0
    a_i = a_i + 2*i;
                                                        a_1 = 0 + 2 = 2
                                                        a_2 = 2 + 4 = 6
                                                        a_3 = 6 + 6 = 12
```

# Индуктивни дефиниции и рекурсивни функции

```
int a (int i)
                                                          \{a_i\}_{\infty}^{i=0}
  if (i == 0)
    return 0;
                                                          a_0 = 0
                                                          a_i = a_{i-1} + 2i
  return a(i-1) + 2*i;
void print_first_n (int n)
  for (int i = 0; i <= n; i++)</pre>
    cout << "a["<< i << "]=" << a(i) << endl;
```

# Доказателство по индукция

**Теорема.** За членовете на редицата  $\left\{a_i\right\}_{\infty}^{i=0}$ , дефинирани по следния начин:

$$a_0 = 0$$
  
$$a_i = a_{i-1} + 2i$$

е изпълнено, че  $a_i=i(i+1)$ , за всяко  $n\in\mathcal{N}$ .

#### Доказателство.

- За i=0 свойството е изпълнено по дефиниция, тъй като  $a_0=0=0(0+1)$ .
- Нека свойството  $a_i=i(i+1)$  е изпълнено за някое  $i\in\mathcal{N}$ . Искаме да покажем, че  $a_{i+1}=(i+1)(i+1+1)$  (заместваме i с i+1). По дефиниция имаме  $a_{i+1}=a_i+2(i+1)$ . Като заместим  $a_i$  с i(i+1) (което сме допуснали), получваме  $a_{i+1}=i(i+1)+2(i+1)=(i+1)(i+2)$ ,

# Прилики / разлики?

```
int a (int i)
{
  if (i == 0)
    return 0;

  return a(i-1) + 2*i;
}
```

**Теорема.** За членовете на редицата  $\left\{a_i\right\}_{\infty}^{i=0}$ , дефинирани по следния начин:

$$a_i = a_{i-1} + 2i$$

е изпълнено, че  $a_i = i(i+1)$ , за всяко  $n \in \mathcal{N}$ .

#### Доказателство.

- За i=0 свойството е изпълнено по дефиниция, тъй като  $a_0=0=0(0+1)$ .
- Нека свойството е изпълнено за  $a_{i-1}=(i-1)i$  при i>0. Искаме да покажем, че  $a_i=i(i+1)$  (заместваме i-1 с i). По дефиниция имаме  $a_i=a_{i-1}+2i$ . Като заместим  $a_{i-1}$  с (i-1)i (което сме допуснали), получваме  $a_i=(i-1)i+2i=i(i-1+2)=i(i+1)$ , което е търсенето свойство за  $a_i$ .

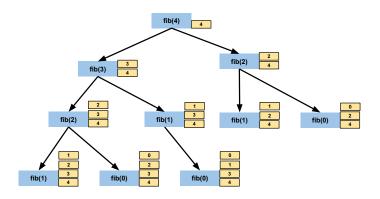
## п-то число на Фибоначи

```
int fib_n (int n)
{
   if (n == 0)
     return 1;
   if (n == 1)
     return 1;

   return fib_n(n-2) + fin_n(n-1);
}
```

```
{a_i}_{\infty}^{i=0}
a_0 = 1
a_1 = 1
a_{i+2} = a_{i+1} + a_i
```

### n-то число на Фибоначи



```
int fib_n (int n)
{
   if (n <= 1)
     return 1;
   return fib_n(n-2) + fin_n(n-1);
}</pre>
```



## Факториел

```
long fact_rec (long n)
  if (n \le 1)
    return 1;
  return n*fact_rec(n-1);
long fact_iter (long n)
  long result = 1;
  while (n > 1)
    result *= n;
    n - -;
  return result;
```

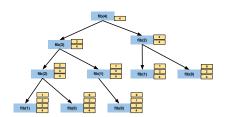
```
n! = \begin{cases} 1 & \text{if } n \leq 1 \\ n \times (n-1)! & \text{otherwise} \end{cases}
0! = 1
n! = n \times (n-1) \times (n-2) \times ... \times 1
```

## Факториел



```
long fact_rec (long n)
{
   if (n <= 1)
      return 1;
   return n*fact_rec(n-1);
}</pre>
```

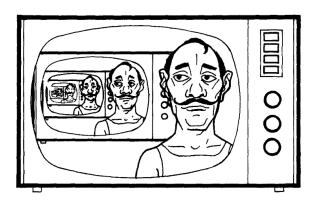
# Сравнение



```
fact(4) 4 fact(3) 3 4 fact(2) 2 3 4 fact(1) 4
```

```
void fib_n (int n)
{
   if (n <= 1)
      return 1;
   return fib_n(n-2) + fin_n(n-1);</pre>
```

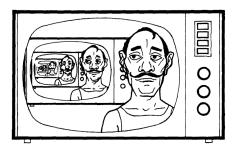
```
long fact_rec (long n)
{
  if (n <= 1)
    return 1;
  return n*fact_rec(n-1);</pre>
```



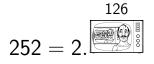
Wirth., N., "Algorithms + Data Structures = Programs", Prentice Hall, 1976

## Разлагане на прости делители

$$252 = ?$$



# Разлагане на прости делители



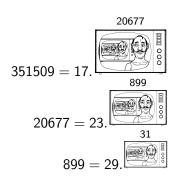


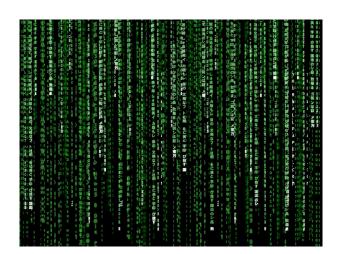
# Разлагане на прости делители

```
void print_divs (int n)
{
  if (n <= 1)
    return;

  int i = 2;
  while (i <= n && n % i != 0)
    i++;

  cout << i << ",";
  print_divs(n/i);
}</pre>
```

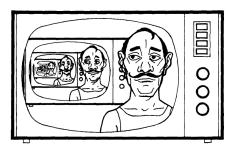




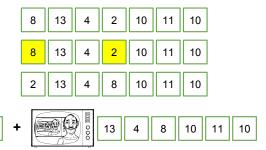


# Сортиране с рекурсия?





# Пряка селекция



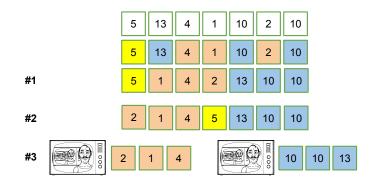
# Сортиране с пряка селекция

```
void ssort (int arr[], int n)
  if (n <= 1)
    return;
  //find the INDEX OF the minimal
  //element of the array
  int minelIx = 0:
  for (int i = 1; i < n; i++)</pre>
    if (arr[minelIx] < arr[i])</pre>
      minelIx = i:
  //swap the minimal element and
  //the element at position 0
  int tmp = a[0];
  a[0] = arr[minelIx];
  arr[minelIx] = tmp;
  //sort the "tail" of the array
  ssort (arr+1.n-1):
```

# Да припомним двоичното търсене

```
bool findrec (int x, int a[], int size)
  if (size == 0)
    return false;
                                             left:0
                                                                   right:6
  if (size == 1)
    return a[0] == x;
  if (a[size/2] > x)
                                                           left:4
                                                                  right:6
    return findrec (x,a,size/2);
  if (a[size/2] < x)
    return findrec (x,a+(int)ceil(size/2.0),ceil(size/2.0)-1);
  return true;
```

## Бързо сортиране



# Бързо сортиране

```
bool qsort (int a[], int size)
                                                                   10
                                                             10
                                                                   10
  if (size <= 1)
    return;
                                                          2
                                                             13
                                  #1
                                                                10
                                                                   10
  //1
                                  #2
                                                             13
                                                                10
      nsmaller
      split (a+1, size-1, a[0]);
                                                                   10
  swap (a[0],a[nsmaller]);
  //3
  qsort(a,nsmaller);
  qsort(a+nsmaller+1, size-nsmaller-1);
```

Благодаря за вниманието!