Информатика

Лекция 5

Переход от алгоритма к программе

Пример

- Содержательная постановка задачи
  - Дан вектор A = (a1, a2, …, an).
    Найти первый отрицательный элемент и его номер.
  - A = (0,5; -1,52; 22,0; -98,75)
    ответ -1,52; номер второй
  - A = (2,57; 0,008; 151,83)
    ответ – вектор не содержит отрицательных элементов

Формальная постановка задачи

- Дан вектор $A = (a_1, a_2, \ldots, a_n)$, $a_i \in R$; $i = 1,n$
  Найти $b \in R$ – значение первого отрицательного элемента и
  $m \in N$ – номер первого отрицательного элемента
- Пусть $C = \{ i \mid a_i < 0, i = 1,n \}$

$C = \varnothing$ – вектор не содержит отр. элементов

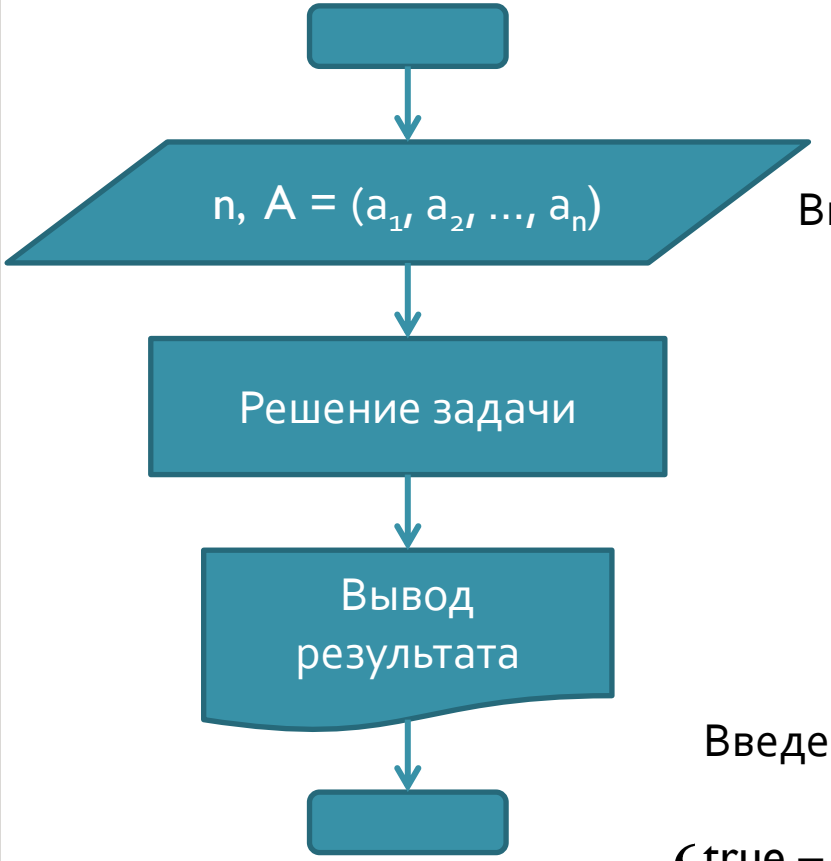
$C = \{ i_1, i_2, \ldots i_p \}$ $1 \le i_k \le n$; $k = 1,p$

- Тогда $m = \min(i_1, i_2, \ldots i_p)$ и $b = a_m$

Алгоритмизация

- Метод – просматриваем последовательно все элементы вектора A, начиная с первого
- Возможны два варианта:
  - На каком-то шаге обнаружили отрицательный элемент, фиксируем его значение и номер – конец работы;
  - Просмотрены все элементы вектора и не обнаружены отрицательные – вывод, что таких элементов нет – конец работы

Первый уровень алгоритма

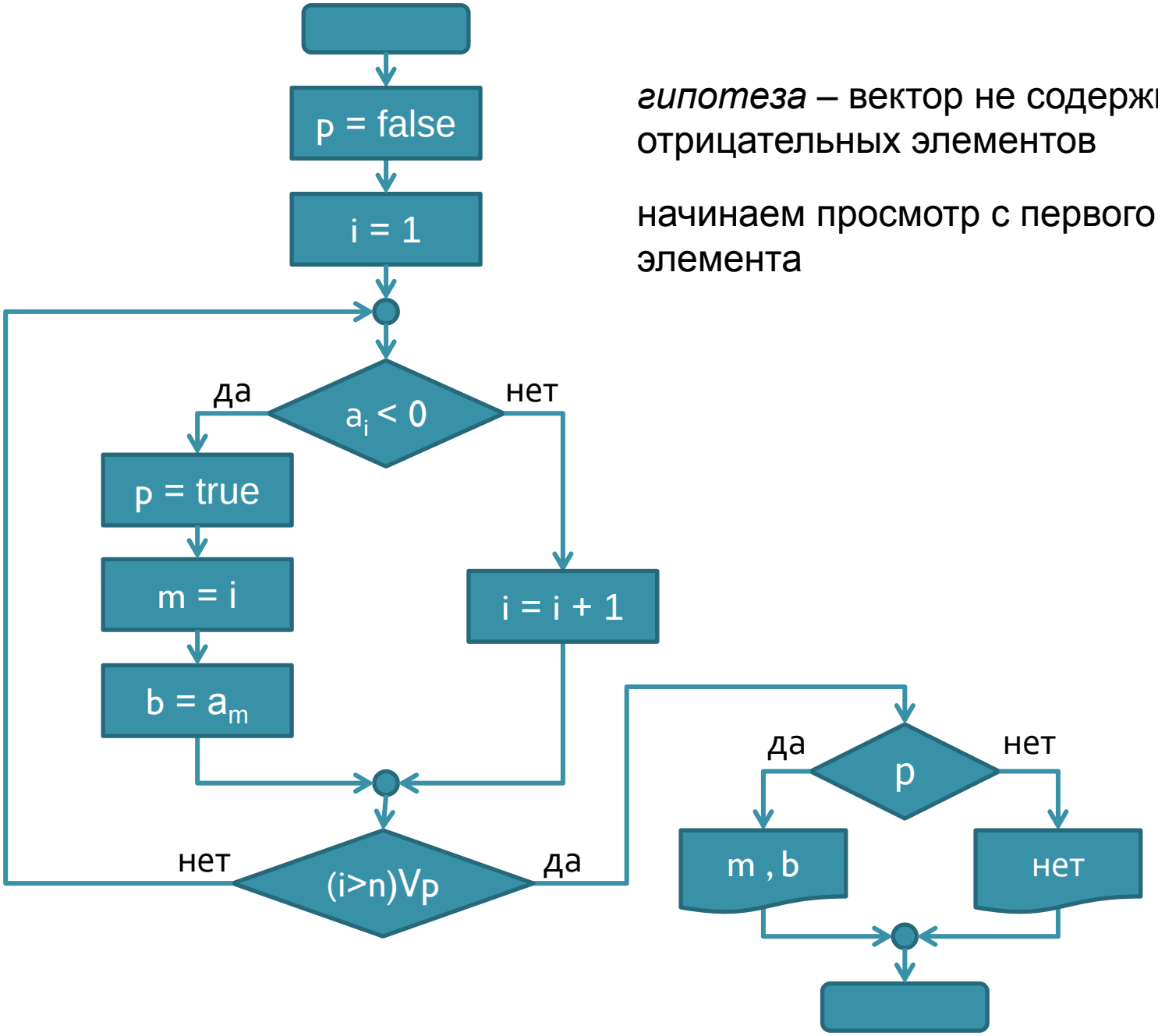


Ввод значений элементов вектора A

Детализируем процессы:
«решение задачи»
и «вывод результата»

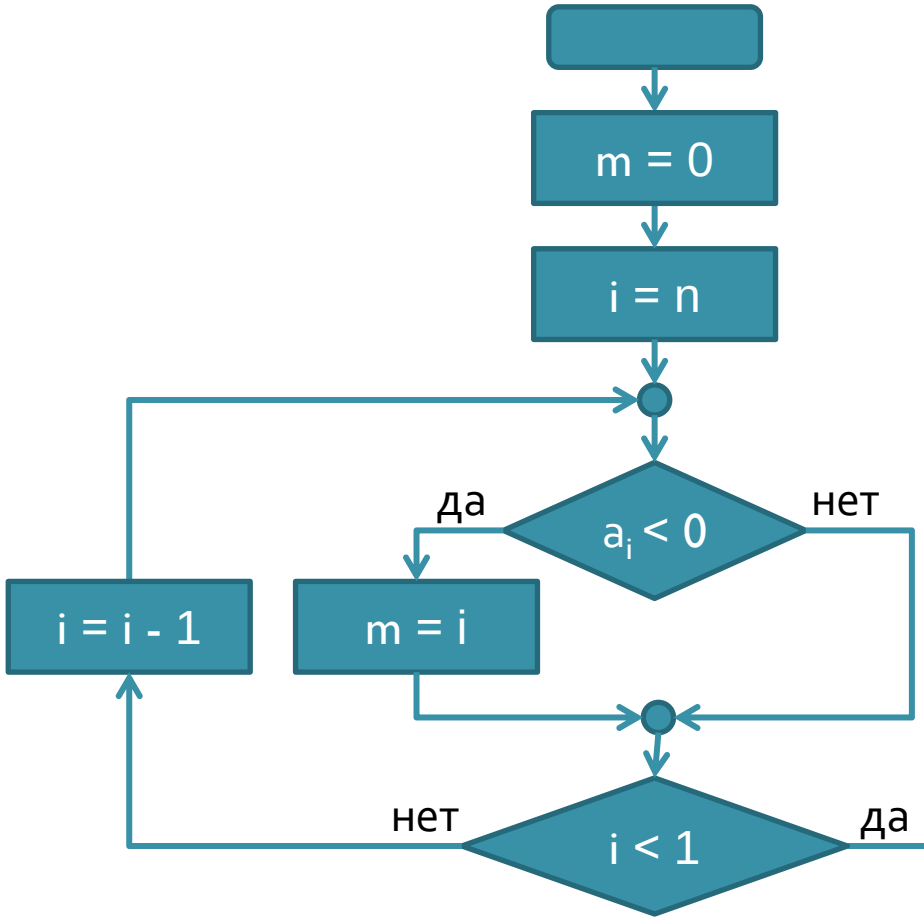
Введем булевскую переменную :

$$p = \begin{cases} \text{true} – \text{есть хотя бы один отрицательный} \\ \text{false} – \text{в векторе нет отрицательных элементов} \end{cases}$$

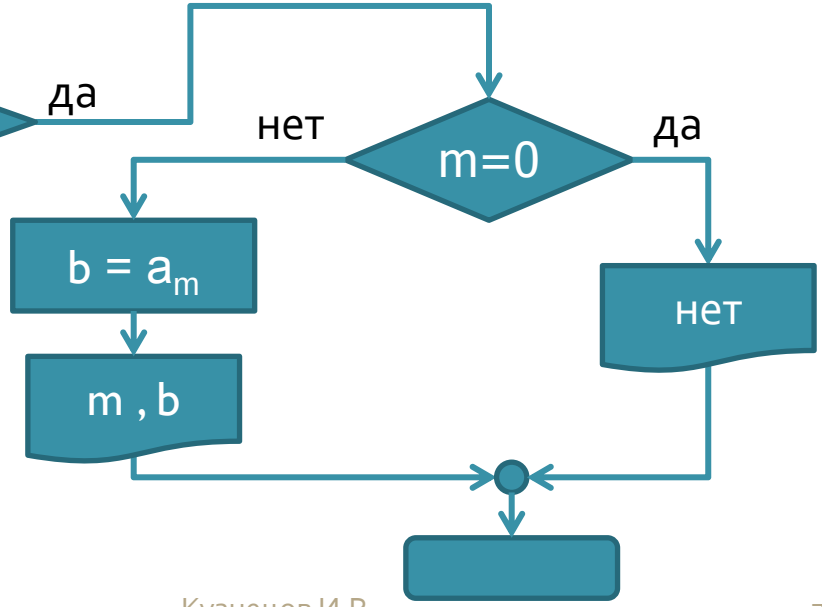


гипотеза – вектор не содержит отрицательных элементов

начинаем просмотр с первого элемента




```
begin
m = 0;
for i = n downto 1 do
      if (a[i] < 0) then m = i;
if (m=0) then
   write (“нет отрицательных)
else
   begin
      b = a[m];
      write (m , b)
   end;
end.
```



можно обойтись без булевского признака

Программирование

Общие понятия

- Программирование –
  - процесс составления плана действий
  - дисциплина, изучающая методы и приемы составления программ
- Программа –
  - план действий, подлежащих выполнению некоторым исполнителем, обычно автоматическим устройством
- Оператор –
  - грамматическая конструкция языка программирования, выражающая некоторое законченное действие при выполнении программы
- Объект –
  - элемент данных, который доступен и может быть обработан некоторой программой (действием, оператором)

Общие понятия (2)

- Программа – алгоритм, записанный на некотором формальном языке, основанном на конкретных представлениях и структурах данных
- Триада – <вход><алгоритм><выход>
  - <объект><оператор><объект>
- Переменная – триада (имя переменной, обозначение текущего значения, само текущее значение), причем обозначение текущего значения понимается как адрес поля памяти, в котором оно помещается
  - <имя><ссылка><значение>

Пример

Вычисление частичной суммы ряда

- **Содержательная постановка:**
  - Дан ряд (сходящийся)

$$S = 1 + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} \ldots,$$

  - Найти частичную сумму *n* слагаемых

S = 1 + 1 + 0,5 + 0,166666 + 0,0416666 = 2,708332 (5 сл.)

S = 2,708332 + 0,0083333 + 0,0013888 = 2,7180553 (7 сл.)

S = 2,7182785 (9 сл)

S = 2б7182814 (11 сл.) e = 2,718281828459

Вычисление факториала

На 8-разрядном микрокалькуляторе:

0! = 1 (по определению)	1/(0!) = 1
1! = 1	1/(1!) = 1
2! = 2	1/(2!) = 0,5
3! = 6	1/(3!) = 0,1666666
4! = 24	1/(4!) = 0,0416666
5! = 120	1/(5!) = 0,0083333
6! = 720	1/(6!) = 0,0013888
7! = 5040	1/(7!) = 0,0001984
8! = 40320	1/(8!) = 0,0000248
9! = 362880	1/(9!) = 0,0000027
10! = 3628800	1/(10!) = 0,0000002
11! = 39916800	1/(11!) = 0,0000000

Формальная постановка

- Дано n $\in$ N (n=1, 2, …)
  найти s $\in$ R, где $$S^{(n)} = 1 + \sum_{i=1}^{n-1} \frac{1}{i!}$$

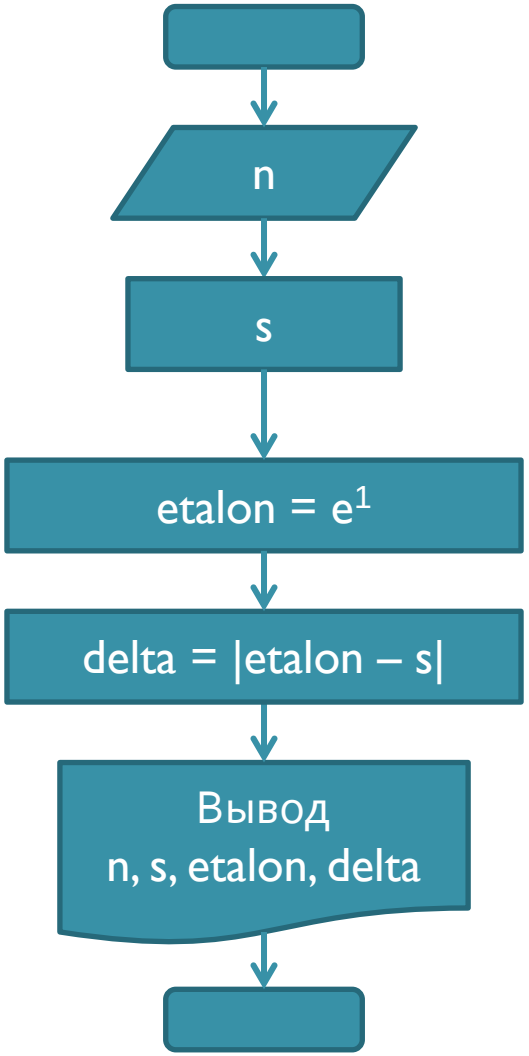
- В цикле следует использовать результаты предшествующего шага, т.е.

$$f(i) = \frac{1}{i!} \qquad f(i-1) = \frac{1}{(i-1)!}$$

$$\frac{f(i)}{f(i-1)} = \frac{(i-1)!}{i!} = \frac{1 * 2 * .. * (i-1)}{1 * 2 * .. * i} = \frac{1}{i}$$

тогда $$f(i) = \frac{f(i-1)}{i}$$

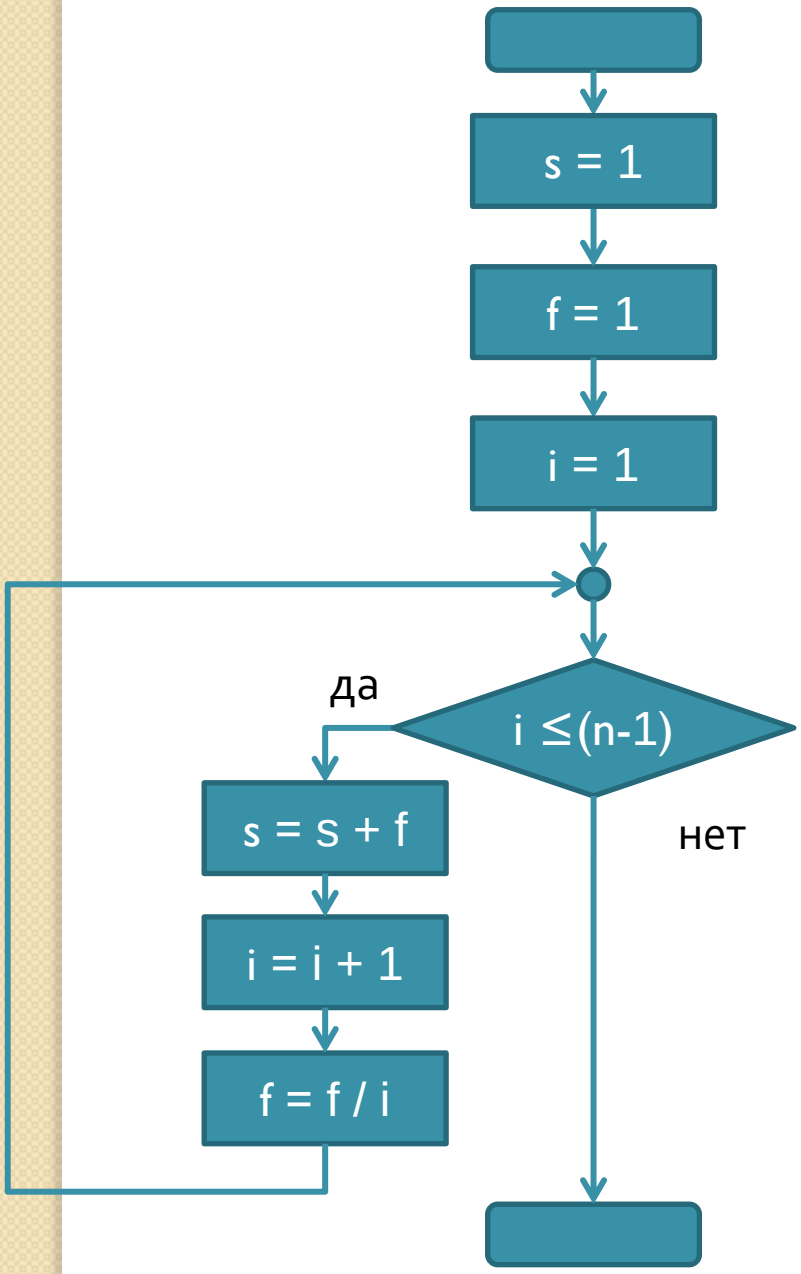
Первый уровень алгоритма



Ввод числа членов ряда

Вычисление частичной суммы

Раскроем процесс вычислений частичной суммы



Раскроем процесс вычислений частичной суммы на основе цикла с известным числом повторений . Вводим счетчик i от 1 до n-1 и на каждом шаге цикла рассчитываем следующее слагаемое ряда согласно рекуррентной формуле f = f / i

Текст программы

```
void_fastcall TForm1::Button1Click(Tobject*Sender)
{ int i, n ;
  float s, f, etalon, delta ;
   s = 1; f = 1;
   n = StrToInt (RichEdit1→Lines → Strings[0]);
   i = 1;
   while ( i <= (n – 1) )
   { s = s + f; i++; f = f / i; }
   etalon = exp( 1 ); delta = abs( etalon – s );
   RichEdit2 →Lines →Add (“частичная сумма “+
   IntToStr (n) + ” слагаемых равна ”+ FloatToStr (s));
   RichEdit2 →Lines →Add (“погрешность равна “ +
   FloatToStr (delta));
}
```

Цикл лекций подготовлен в 2013/2014уч. году Кузнецовым Игорем Ростиславовичем, доцентом кафедры радиоэлектронных средств Санкт-Петербургского Государственного электротехнического университета «ЛЭТИ»

Прочитан в дисциплине
«Информатика»