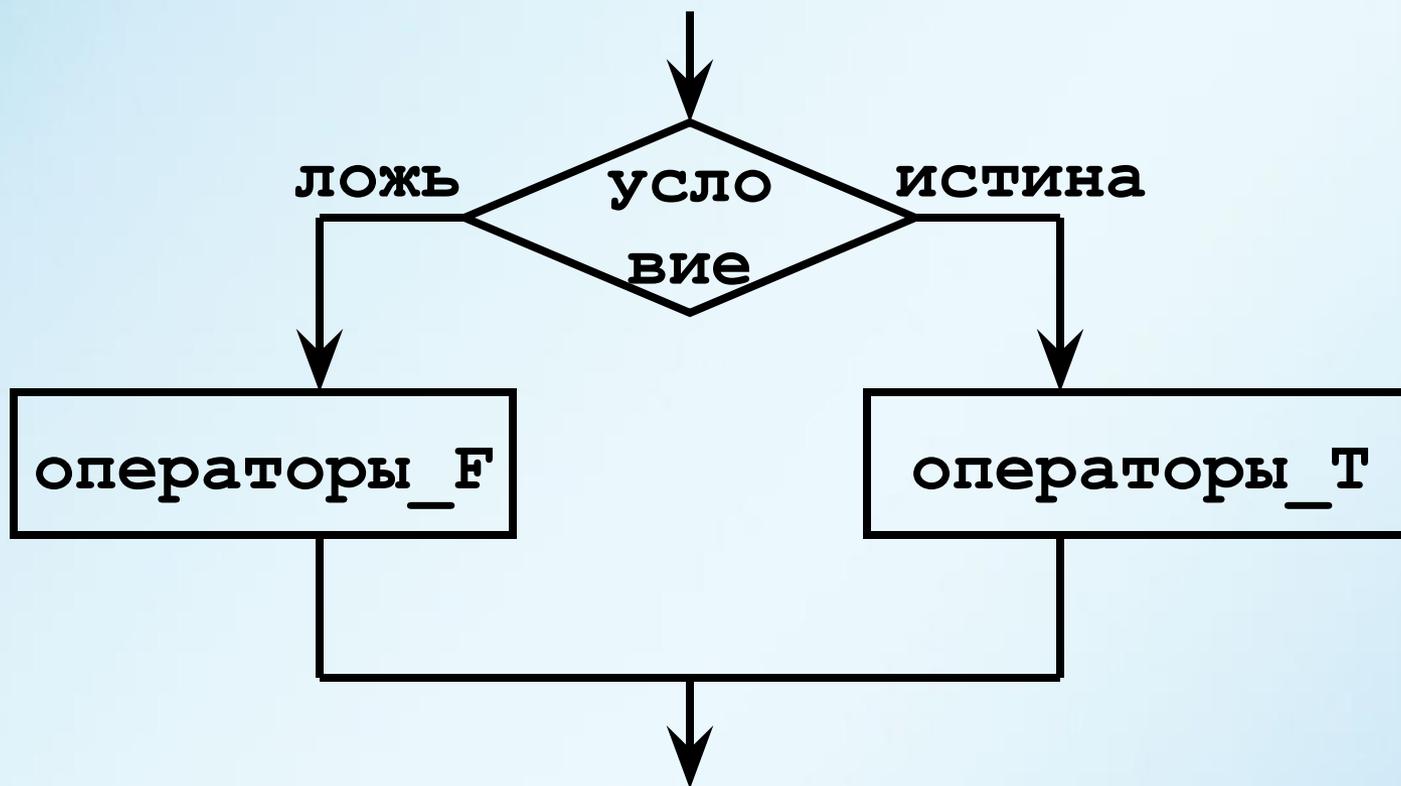


Лекция 3

Управляющие операторы

Условный оператор if



Создание разветвлений в программах, принятие решений, проверка условий, исключительные ситуации, проверка ошибок.

Условный оператор if

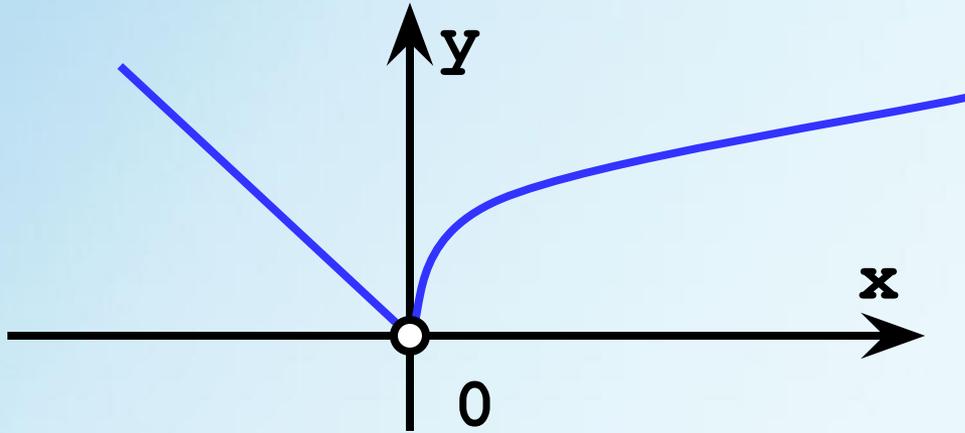
① имя: `if` (условие) `then`
 операторы_T
`else`
 операторы_F
`end if` имя

② имя: `if` (условие) `then`
 операторы_T
`end if` имя

③ `if` (условие) оператор_T

Условие - Логическое выражение

Условный оператор if

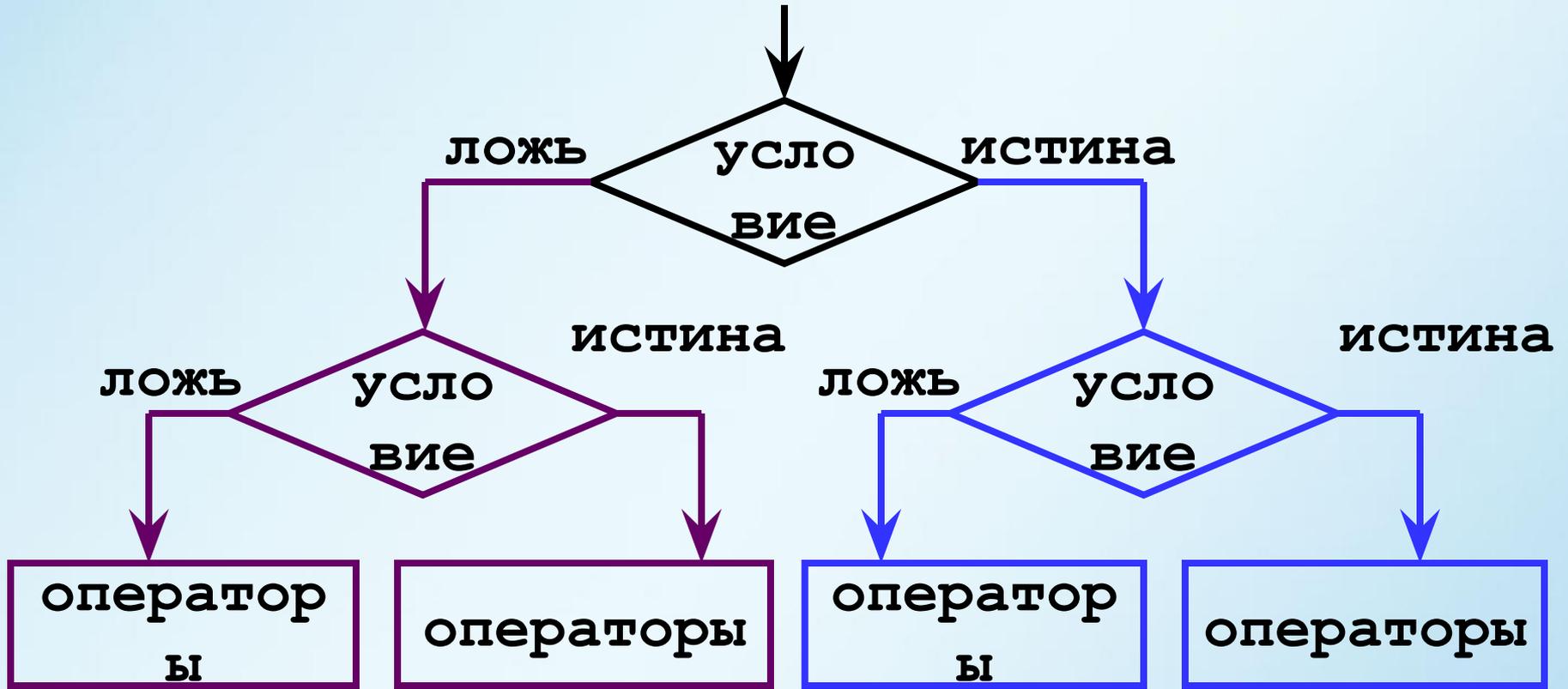


$$f(x) = \begin{cases} -x, & x < 0 \\ \sqrt{x}, & x \geq 0 \end{cases}$$

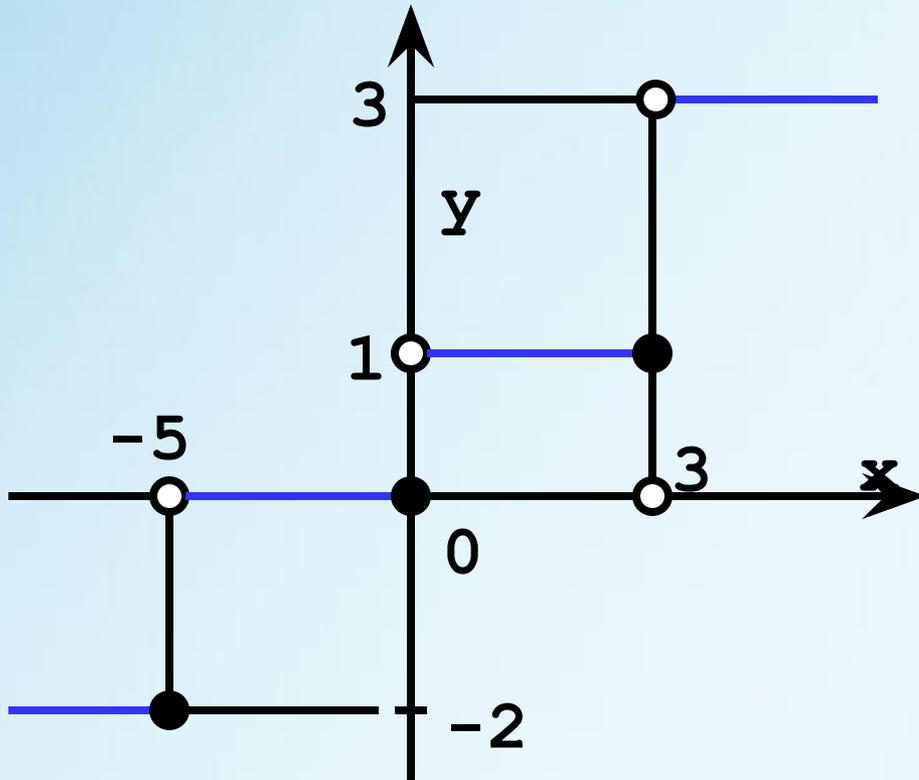
```
program func
  real x,fx
  write(*,"(A,\)") "x = "; read(*,*) x
  if (x < 0) then
    fx = -x
  else
    fx = sqrt(x)
  end if
  write(*,*) "F(x) = ", fx
end
```

Условный оператор if

Вложенные операторы **if** – множественное ветвление.



Условный оператор if

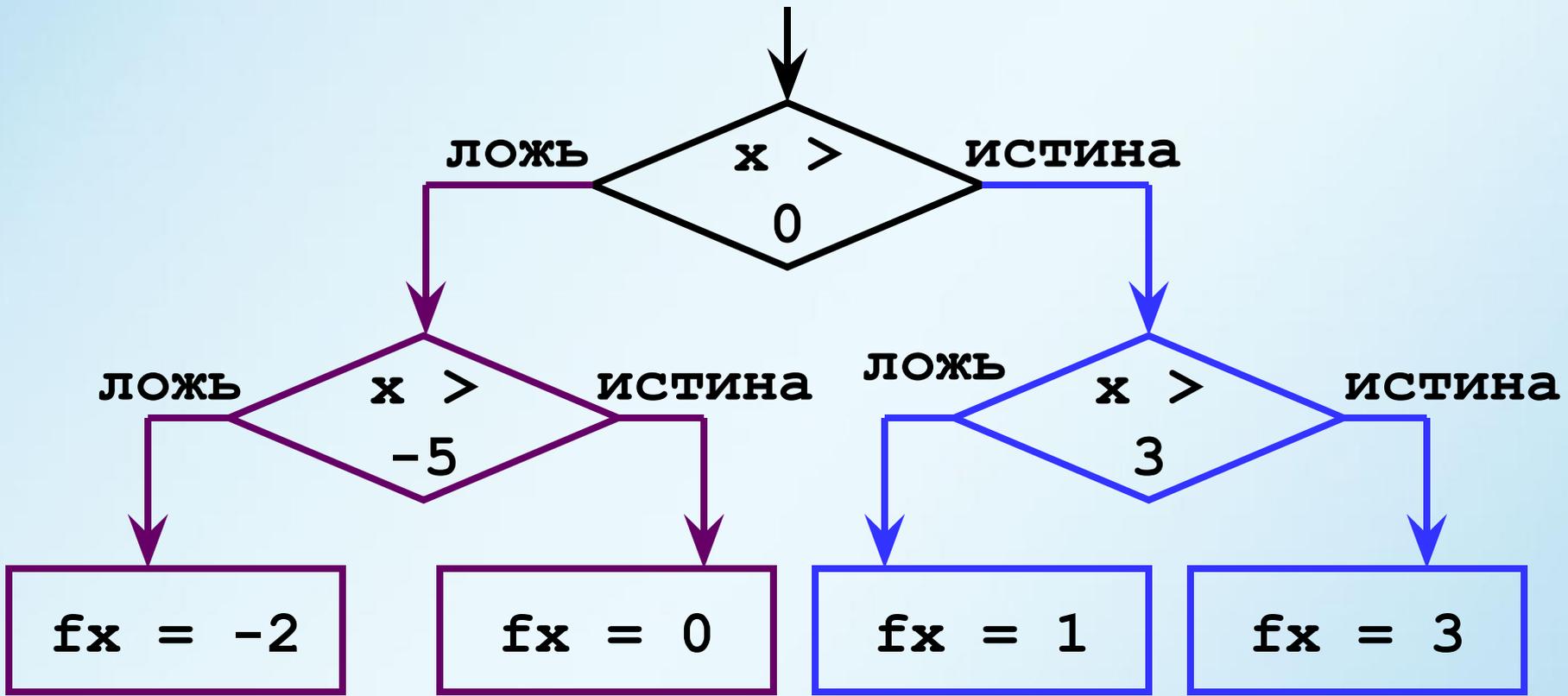


$$f(x) = \begin{cases} -2, & x \leq -5 \\ 0, & -5 < x \leq 0 \\ 1, & 0 < x \leq 3 \\ 3, & x > 3 \end{cases}$$

Возможны несколько вариантов использования **if**.

Условный оператор if

Вариант № 1

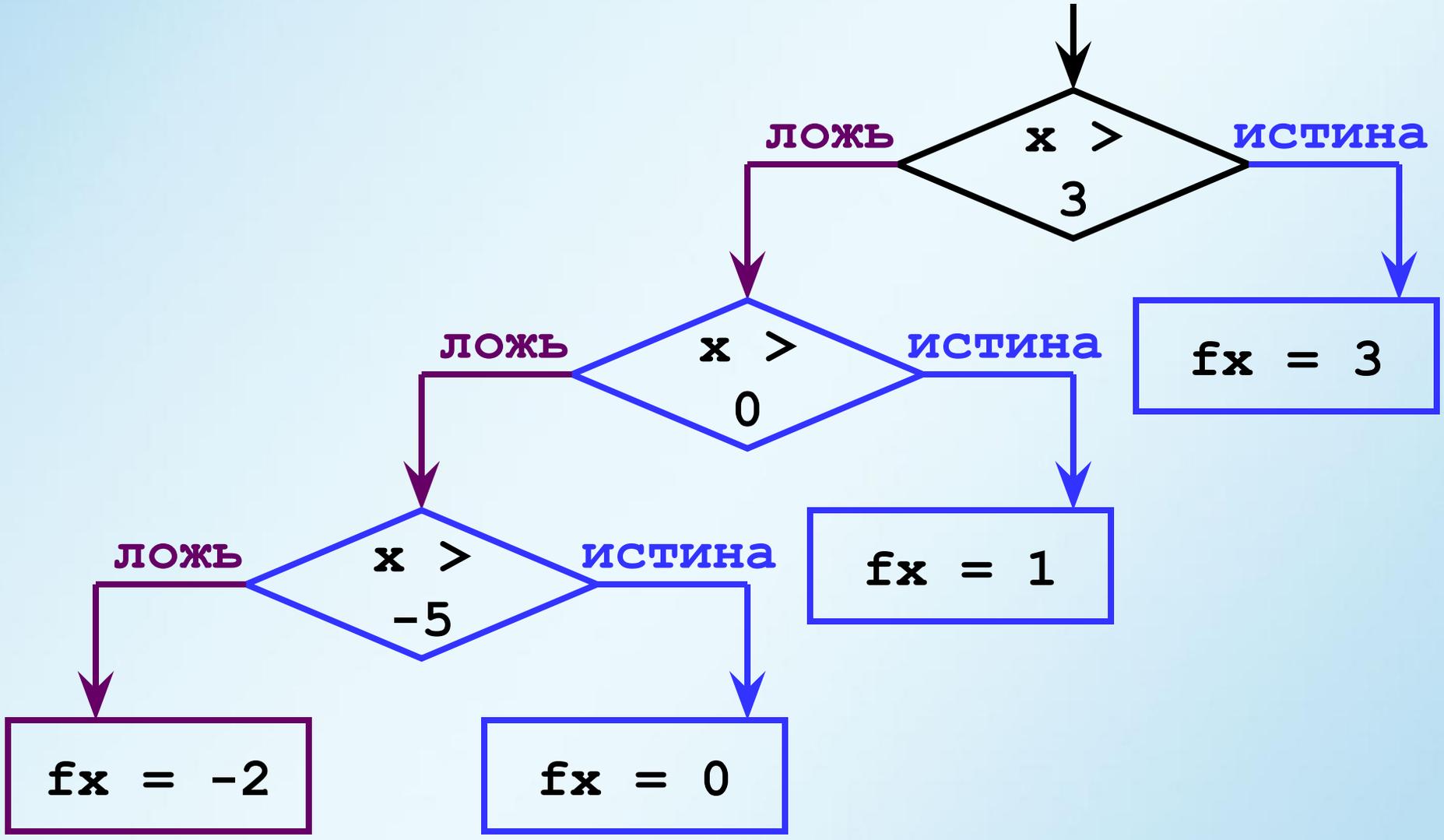


Условный оператор if

```
! -----Вариант № 1
if (x > 0) then
  if (x > -5) then ! -----
    fx = 0          !
  else             !
    fx = -2        !
  end if          ! -----
else
  if (x > 3) then ! -----
    fx = 3        !
  else             !
    fx = 1        !
  end if          ! -----
end if
```

Условный оператор if

Вариант № 2



Условный оператор if

! -----Вариант № 2

```
if (x > 3) then
```

```
    fx = 3
```

```
else
```

```
    if (x > 0) then
```

```
        fx = 1
```

```
    else
```

```
        if (x > -5) then
```

```
            fx = 0
```

```
        else
```

```
            fx = -2
```

```
        end if
```

```
    end if
```

```
end if
```

Можно упростить, используя `elseif`.

Условный оператор if

! -----Вариант № 2а, elseif

```
if (x > 3) then
    fx = 3

elseif (x > 0) then
    fx = 1

elseif (x >= 5) then
    fx = 0

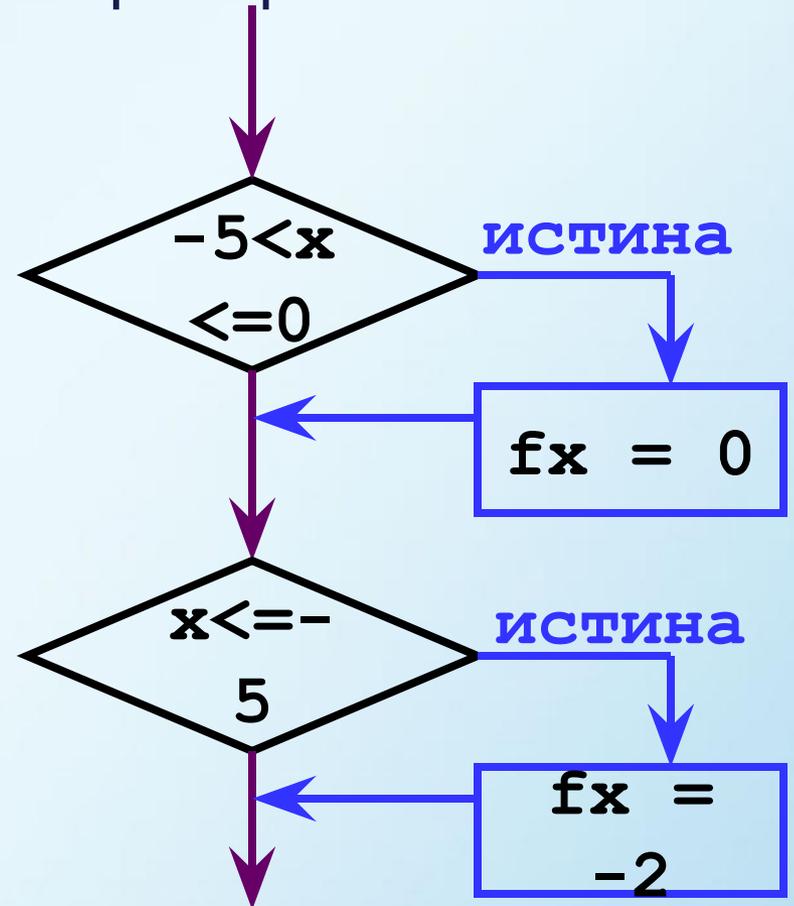
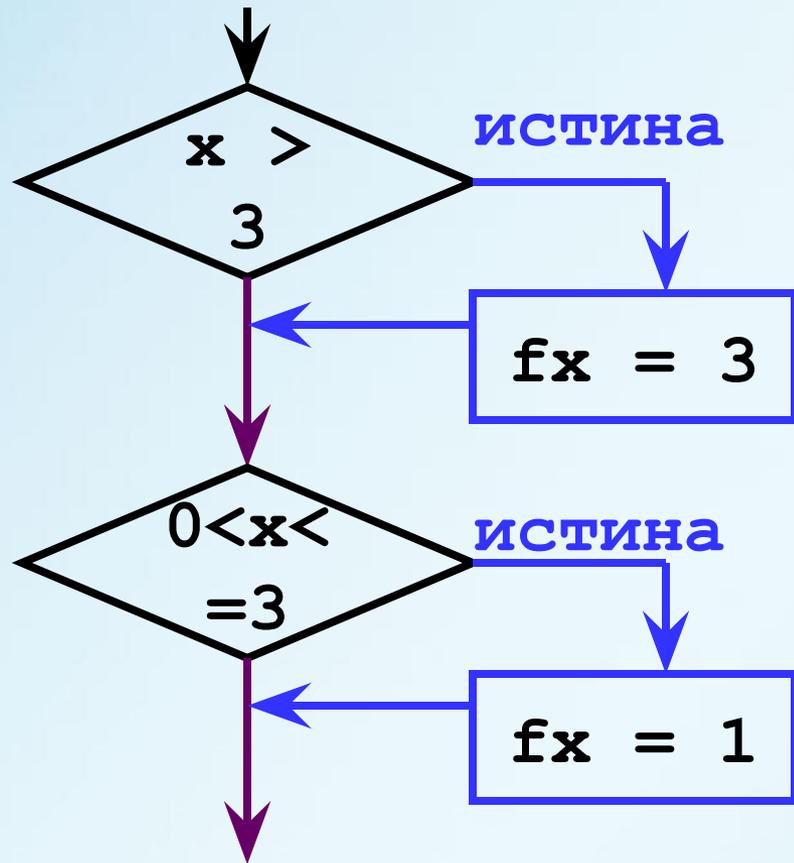
else
    fx = -2
end if
```

Один общий **endif** + легкая читаемость.

Условный оператор if

Вариант № 3

последовательные операторы `if`



Проверка на равенство, проверка каждого условия.

Условный оператор if

Всегда ли следует использовать оператор `if`,
когда произносим "если" ?

*если $A < 0$ тогда $M = A$
иначе $M = 0$*

```
if ( A < 0 ) then  
  M = A  
else  
  M = 0  
end if
```



```
M = min(A, 0)
```

```
if ( A > 0 ) then  
  M = A  
else  
  M = 0  
end if
```



```
M = max(A, 0)
```

Условный оператор if

Переменные-флаги – хранят результаты проверок.

```
logical status
```

```
...
```

```
status = логическое выражение
```

```
...
```

```
if ( status ) then
```

```
    операторы_T
```

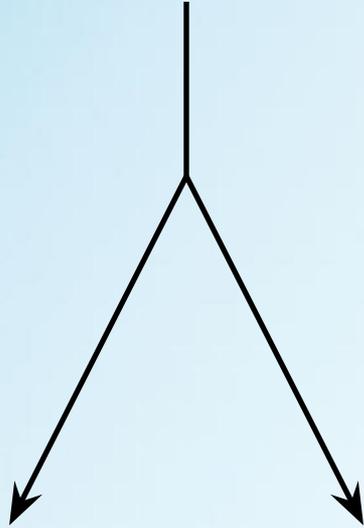
```
else
```

```
    операторы_F
```

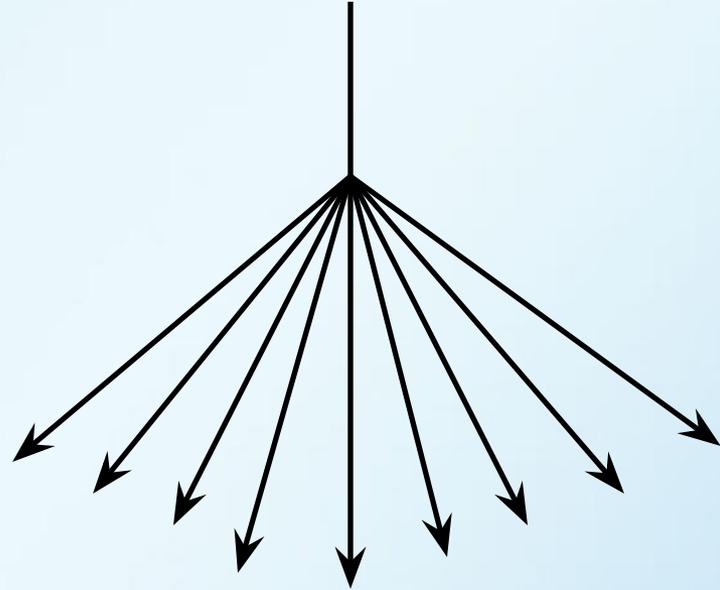
```
end if
```

```
...
```

Оператор выбора `select case`



`if`



`select case`

Обработка клавиш, сообщений, событий,
диапазонов целых или символьных данных.

Оператор выбора `select case`

```
select case (выражение)
  case (множество_значений_1)
    операторы
  case (множество_значений_2)
    операторы
  ...
  case default
    операторы
end select
```

Выражение должно быть
целого, символьного или логического типа.

Оператор выбора select case

```
program calculator  ! программа "Калькулятор"
  real a,b,res
  character op
  logical :: flagERR = .FALSE.

  write(*,"(A,\)") "a = "; read(*,*) a
  write(*,"(A,\)") "b = "; read(*,*) b
  write(*,"(A,\)") "Operation = "; read(*,*) op

  select case (op)
    case ('+'); res = a+b
    case ('-'); res = a-b
    case ('*'); res = a*b
    case ('/'); res = a/b
    case default; flagERR = .TRUE. ; write(*,*) "ERROR!"
  end select

  if (.NOT.flagERR) write(*,*) "Result...", res
end
```

Оператор выбора select case

```
program interval ! попадание в целочисленный интервал
```

```
integer k
```

```
write(*,"(A,\)") "Enter number >= 0 ..."  
read(*,*) k
```

```
select case(k)
```

```
  case (0);          write(*,*) "0"
```

```
  case (1:9);       write(*,*) "1..9"
```

```
  case (10:99);    write(*,*) "10..99"
```

```
  case (100:999);  write(*,*) "100..999"
```

```
  case default
```

```
    write(*,*) "----- OVERFLOW -----"
```

```
end select
```

```
end
```

Операторы `goto` и `continue`

Передача управления по метке

`goto` метка

Оператор `continue` – пустой оператор, не выполняет никаких действий и не оказывает влияния на программу

```
...  
if (ошибка_1) goto 100 ! реакция на ошибки  
...! в одном месте  
read(*,*,ERR = 100) ....  
...  
  
100 continue  
...
```

Оператор цикла do

имя **do** переменная = начало, конец, шаг

операторы ! ----- тело цикла

end do имя

Переменная может быть
целого или вещественного типов.

Организация вычислений,
итерационные алгоритмы,
вычисление сумм, произведений,
подсчёт и перебор значений.

Оператор цикла do

Схема выполнения

- ① Переменной цикла присваивается начальное значение.
- ② Выполнение тела цикла.
- ③ Переменная цикла увеличивается на шаг цикла.
- ④ Если переменная цикла больше конечного значения, то цикл завершает работу, иначе переход на шаг 2.

Оператор цикла do

```
do k = 1, 15      ! цикл выполнится 15 раз
                  ! на последней итерации k=15
                  ! после выполнения цикла k=16

do k = 1, 50, 2   ! цикл выполнится 25 раз
                  ! на последней итерации k=49
                  ! после выполнения цикла k=51

do k = 30, -10, -4 ! цикл выполнится 11 раз
                  ! на последней итерации k=-10
                  ! после выполнения цикла k=-14

do s = 8.0, 10.0 , 0.1
```

Чему равно s на последней итерации цикла
и после выполнения цикла ?

Оператор цикла do

Значение переменной s во время работы цикла

8.400002

8.500002

8.600002

8.700003

8.800003

8.900003

9.000004

9.100004

9.200005

9.300005

9.400005

9.500006

9.600006

9.700006

9.800007

9.900007

после работы цикла

10.00001

Исправление

```
do s = 8.0, 10.0 + 0.1/2, 0.1
```

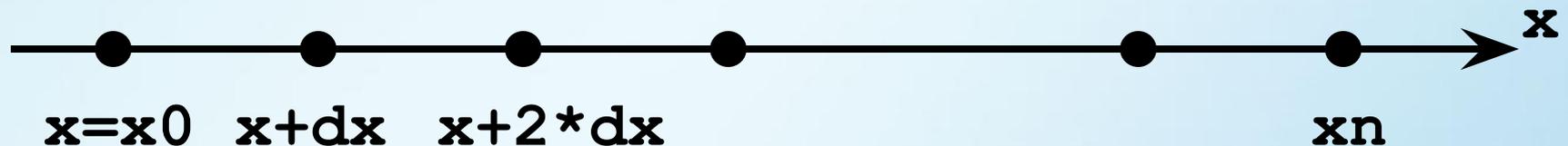
Оператор цикла do

Табулирование функции (1-й способ)

```
program table_1
real :: x0 = 0.0, xn = 1.0, dx = 0.1
real x, fx

do x = x0, xn + dx/2, dx
  fx = sin(x)
  write(*, "(2(a,f8.3,10x))") "x = ", x, "f(x) = ", fx
end do

end
```



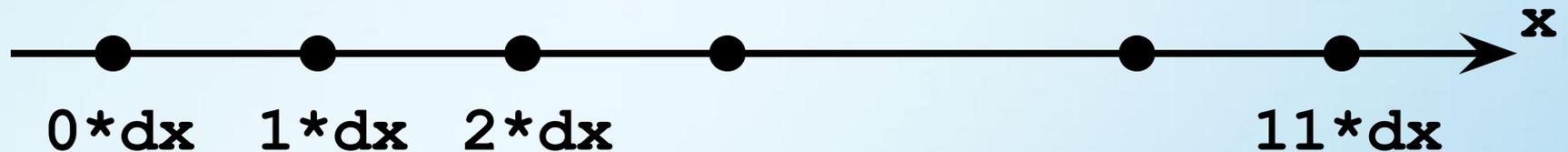
Оператор цикла do

Табулирование функции (2-й способ)

```
program table_2
integer i
real :: dx = 0.1
real x, fx

do i = 0,10
  x = i*dx
  f = sin(x)
  write(*,"(2(a,f8.3,10x))") "x = ",x, "f(x) = ",fx
end do

end
```

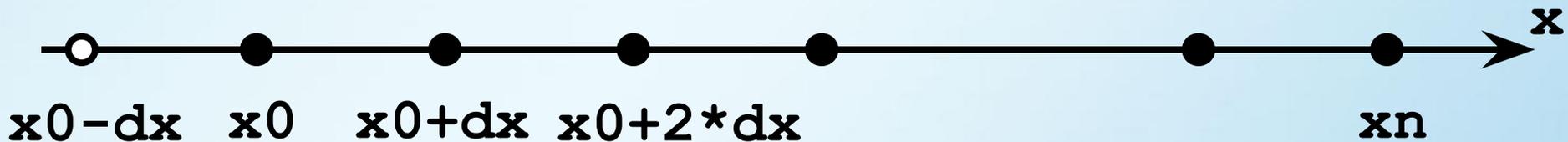


Оператор цикла do

Табулирование функции (3-й способ)

```
program table_3
integer i
real :: dx = 0.1, x = 0, f

  x = x-dx      ! начальное значение x = -0.1
  do i = 0,10
    x = x+dx    ! x накапливает сумму
    f = sin(x)
    write(*,"(2(a,f8.3,10x))") "x = ",x, "f(x) = ",fx
  end do
end
```



Оператор цикла do

Вычисление суммы

$$\sum_{k=1}^{10} \frac{k^2}{k+1}$$

```
program summa
integer k
real s

s = 0.0
do k = 1,10
    s = s + k*k/(k+1.0)
end do
write(*,*) s! 47.01988

end
```

Оператор цикла do

Вычисление произведения

$n!$

```
program summa
integer(8) fact
integer, parameter :: N = 10

fact = 1
do k = 1,N
    fact = fact*k
end do

write(*,*) fact    ! 3628800
end
```

Оператор цикла `do while`

имя цикла: `do while` (логическое условие)

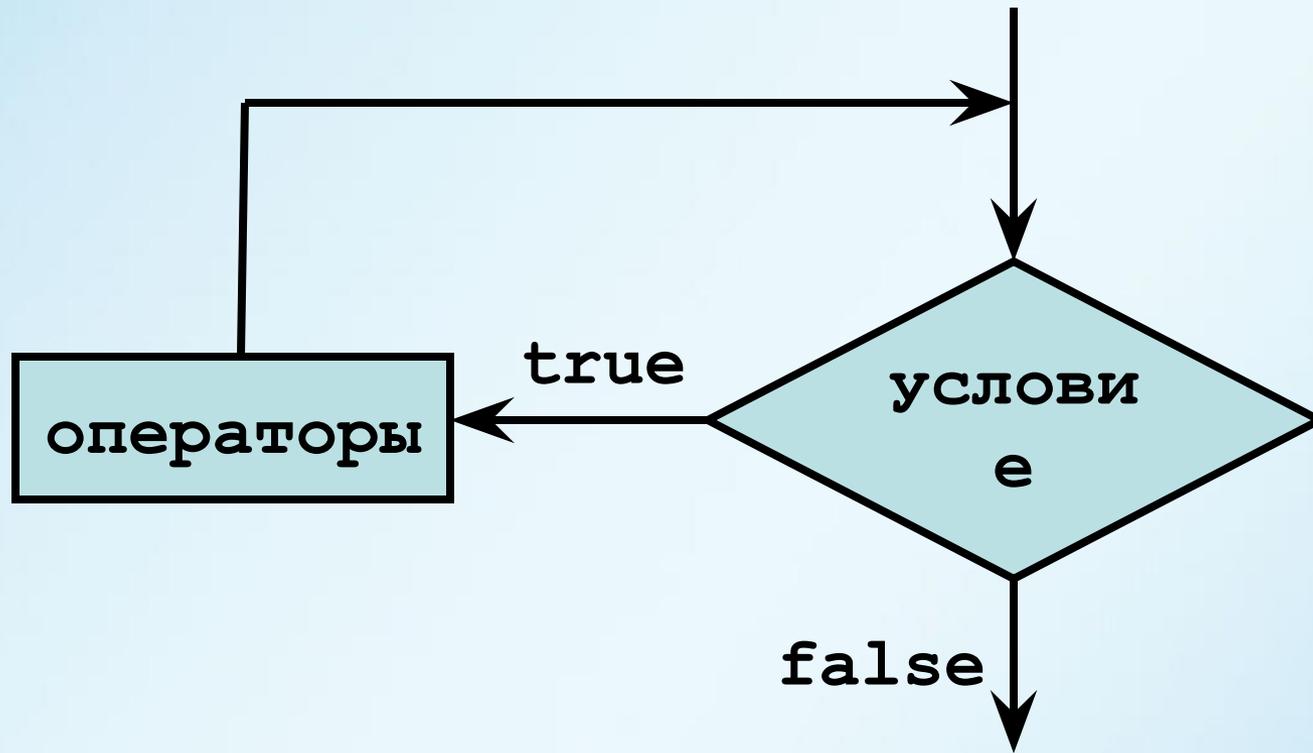
операторы

`end do` имя цикла

Циклы выполняющиеся неопределенное число раз.

Выполнять операторы пока условие истинно.

Оператор цикла do while



Оператор цикла do while

Суммировать ряд пока слагаемое > 0.0005

$$\sum_{k=1} \frac{1}{k^3 + k^2 + 1}$$

```
program summa
real :: sum = 0, slag

  slag = 1.0/(k**3+k**2+1) ! первое слагаемое
do while (slag>0.0005)
  sum = sum + slag
  slag = 1.0/(k**3+k**2+1)
end do

write(*,*) "summa = ", sum, "slag = ", slag
end
```

Вложенные циклы do

имя_1 do i1 = начало, конец, шаг

имя_2 do i2 = начало, конец, шаг

имя_3 do i3 = начало, конец, шаг

операторы

end do имя_3

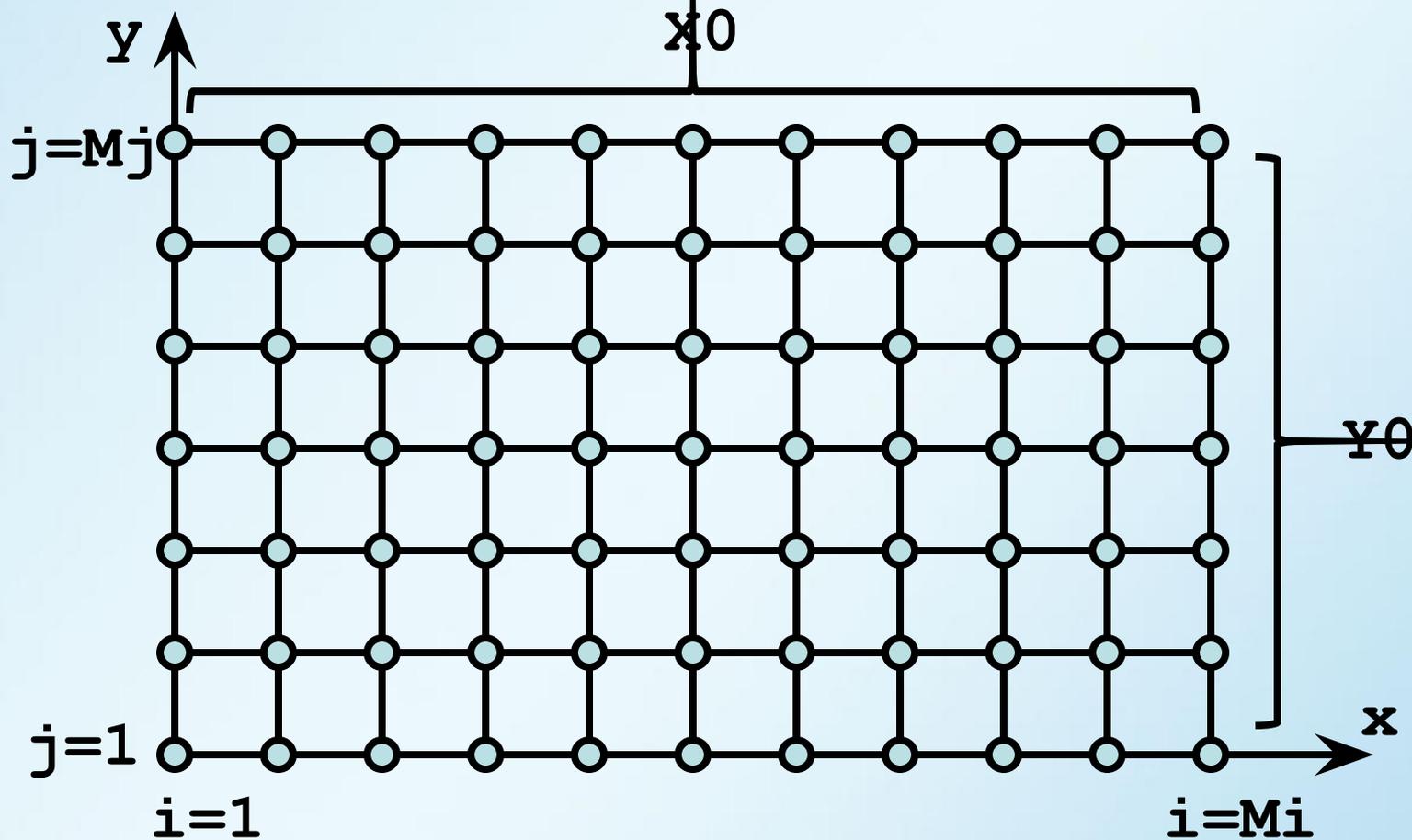
end do имя_2

end do имя_1

Вложенные циклы do

Протабулировать функцию двух переменных

$$f(x, y) = x^2 + y$$



Вложенные циклы do

```
program func_table
integer, parameter :: Mi = 5, Mj = 7 ! сетка
real x, y, fxy, dx, dy
integer i, j
real :: Y0 = 1.0; X0 = 2.0 ! размеры области

dx = X0/(Mj-1); dy = Y0/(Mi-1)
do i = Mi, 1, -1 ! сверху вниз
  y = (i-1)*dy
  do j = 1, Mj
    x = (j-1)*dx
    fxy = x**2+y
    write(*, '(f7.2, \)') fxy
  end do
  write(*, *); ! переход на следующую строку
end do

end
```

Управление циклами

Оператор `exit` – прекращение выполнения цикла.

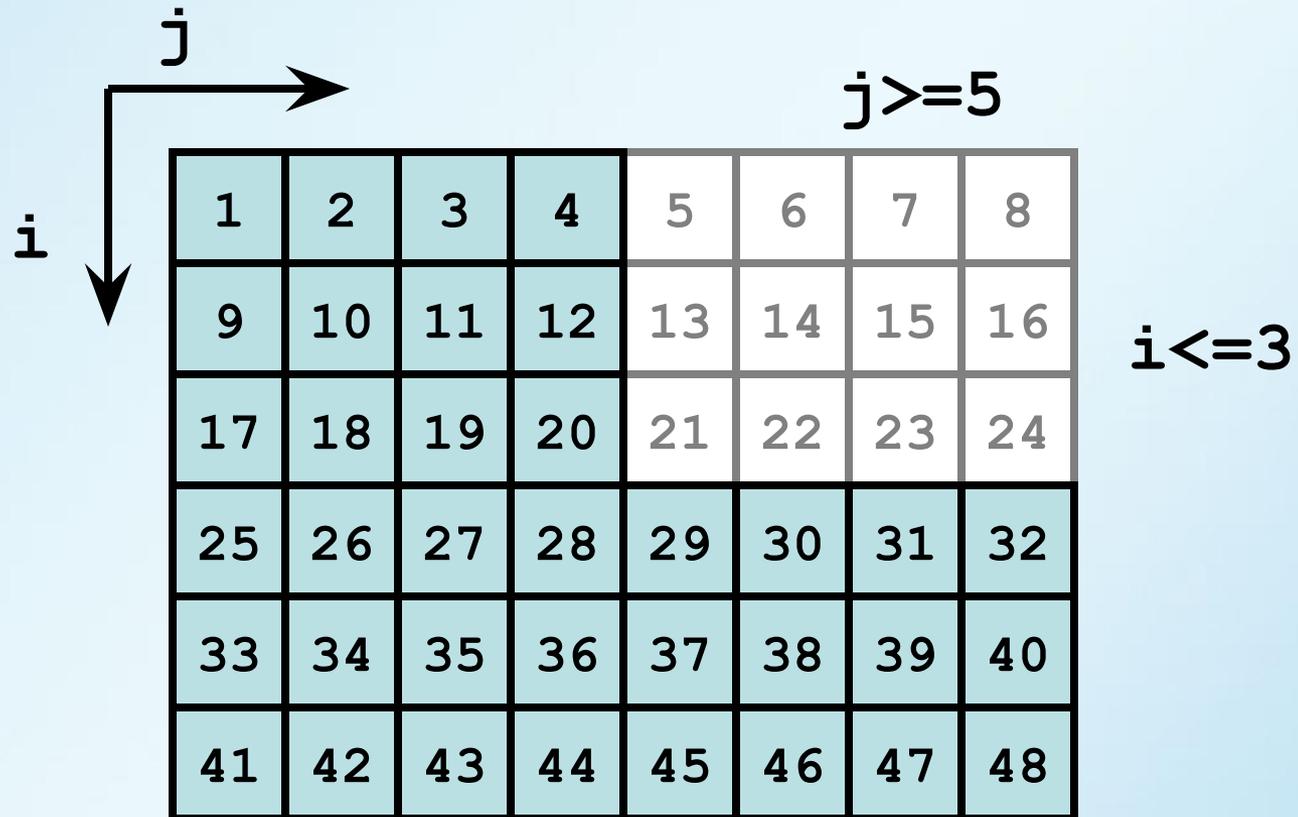
```
C1: do k = 1,100 ! внешний цикл с именем C1
    do i = 1,200
        do j = 1,300
            do n = 1,400

                if (условие) exit C1
                    ! выход из C1

            end do
        end do
    end do
end do C1
```

Управление циклами

Оператор `cycle` – прекращение текущей итерации.



Вывести на экран элементы закрашенной области.

Управление циклами

```
program region
integer :: i,j,s = 0

do i = 1,5
  do j = 1,8
    s = s+1
    if ((i <= 3) .AND. (j >= 5)) cycle ! обход
    write(*,"(i4,\)") s
  end do
  write(*,*)
end do

end
```

Бесконечные циклы

do

операторы

end do

do while (.TRUE.)

операторы

end do

```
program region  
use iflib
```

```
do ! бесконечный вывод псевдослучайных чисел  
  call random(x)  
  write(*,"(i1,\)") floor(x*10)  
end do
```

```
end
```

Бесконечные циклы

Аналог цикла `do while`

`do`

`if` (логическое условие) `exit`
операторы

`end do`

Цикл выполняющийся хотя бы один раз

`do`

операторы
`if` (логическое условие) `exit`

`end do`

Псевдослучайные числа

Получить одно псевдослучайное число

```
call random(x)    0.0 <= x < 1.0
```

Получить одно псевдослучайное число или массив

```
call random_number(x)
```

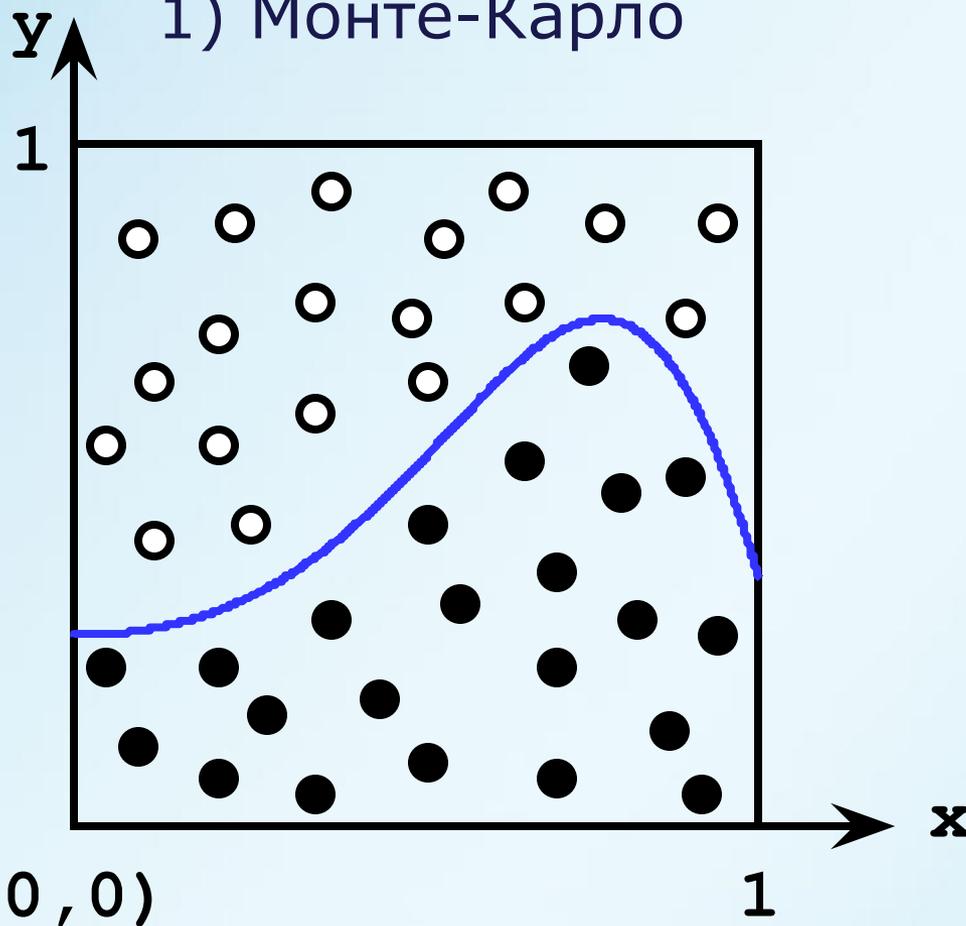
```
real x  
integer R
```

```
call random(x) ; R = int(x*10)      ! [ 0; 9]  
call random(x) ; R = int(x*11)-5   ! [-5; 5]  
call random(x) ; R = int(x*6)*10   ! 0,10,20,30,40,50
```

* З а д а н и е *

Вычислить интеграл $\int_0^1 \frac{\sin(3x^2)}{4} \cdot e^x dx$ методом

1) Монте-Карло



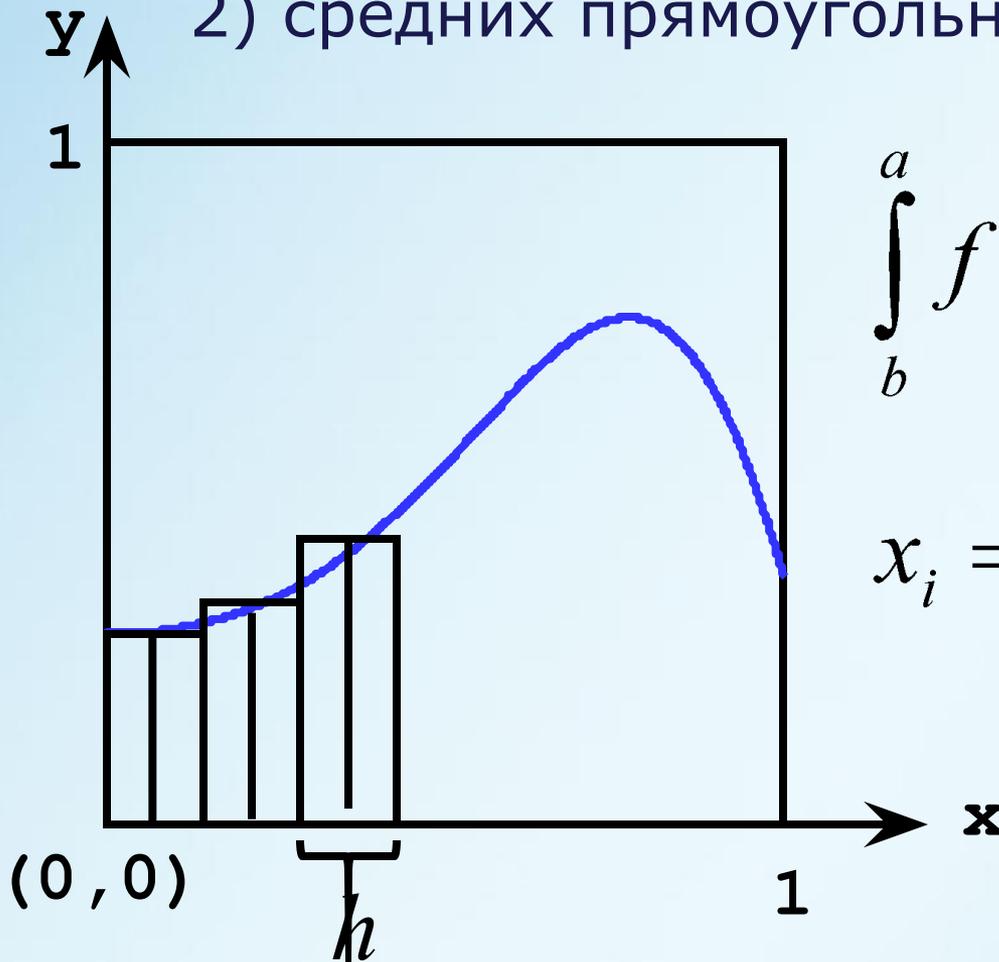
$$\int_0^1 f(x) dx \approx \frac{P}{N}$$

P - черные точки

N - белые точки

* Задание *

2) средних прямоугольников



$$\int_b^a f(x) dx \approx \sum_{i=0}^{N-1} (h \cdot f(x_i))$$

$$x_i = a + i \cdot h + \frac{h}{2}$$

$$h = \frac{(b-a)}{N}$$

$a = 0, \quad b = 1, \quad N$ - число отрезков.

* З а д а н и е *

```
program Monte_Karlo
integer k          ! переменная цикла
real x, y         ! случайные координаты точки
integer :: p=0    ! число попаданий
integer :: n=50000 ! число испытаний
real integral     ! вычисленное значение интеграла

do k=1,n
  call random_number(x)
  call random_number(y)
  if (y<sin(3*x*x)*exp(x)/4) p=p+1
end do

integral=REAL(p)/REAL(n)
write(*,"(A,f7.3)") "Integral = ", integral
end
```

Результат работы программы.

```
Integral = 0.244
Для продолжения нажмите любую клавишу . . .
```

```
program Rectangle
integer i          ! переменная цикла
real x, a, b      ! границы интервала, текущий x
integer :: N=500  ! число прямоугольников
real h           ! ширина прямоугольника
real :: integral=0.0 ! вычисленное значение интеграла

a=0
b=1
h=(b-a)/N

do i=0,N-1
  x=a+i*h+h/2
  integral=integral+h*sin(3*x*x)*exp(x)/4
end do

write(*,"(A,f7.3)") "Integral = ", integral
end
```

Результат работы программы.

```
Integral = 0.250
Для продолжения нажмите любую клавишу . . .
```