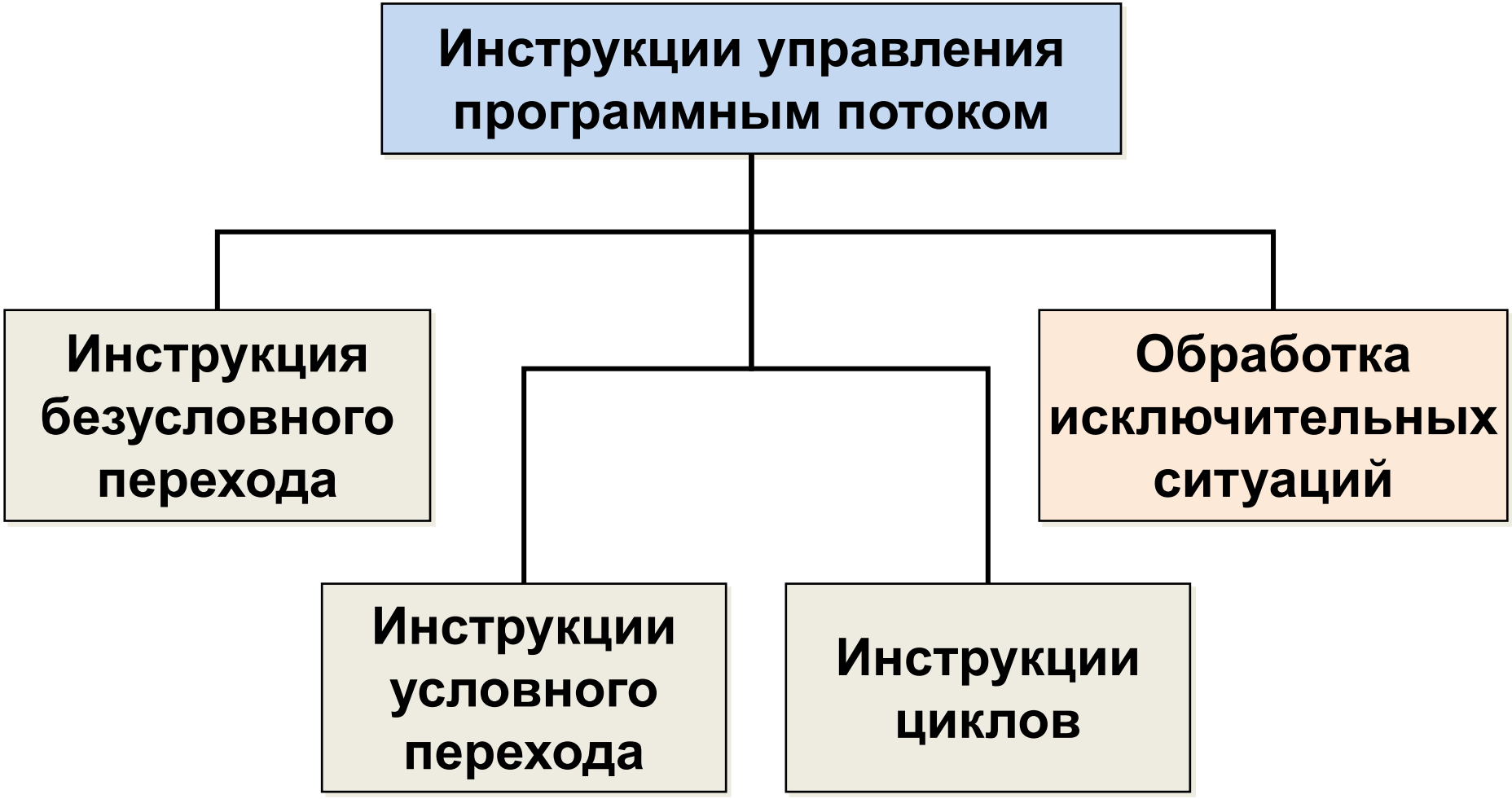
Тема 5.
Управление программным потоком

Инструкции управления программным потоком



Инструкция условного перехода if

Инструкция условного ветвления **if else** используется для разветвления процесса вычислений на два направления

```
if (условие) инструкция_1; [else инструкция_2;]
```

```
if(условие) //принимает значение true или false
{
    блок кода
}
else
{
    блок кода
}
```

Блок выполняется если условие равно «true»

Блок выполняется если условие равно «false»

Сначала вычисляется логическое условие в скобках. Если условие истинно, то выполняется следующая инструкция или блок кода. Если условие ложно, то выполняется инструкция или блок кода, следующий за **else**.

Инструкция условного перехода if

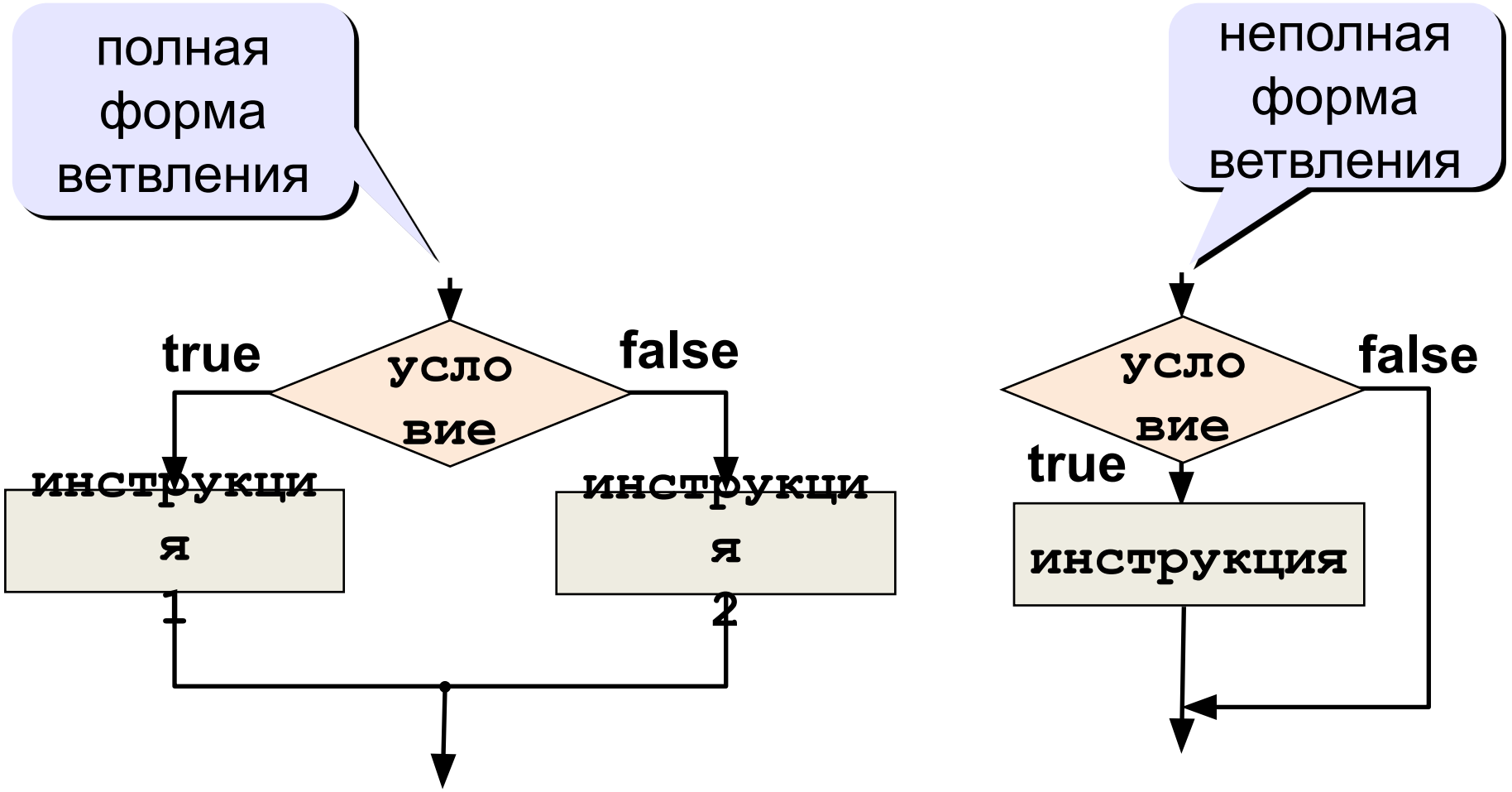
Одна из ветвей может отсутствовать, как правило, опускают вторую ветвь с ключевым словом **else**.

```
if(условие) //принимает значение true или false
{
    блок кода
}
```

Блок выполняется если условие равно «true»

Если условие равно «false» управление передается на инструкцию следующую за **if**

Блок схемы для условного перехода



Примеры инструкции if-else

Примеры кода

```
int a;
int b;
...
int c;

if(a >= b)
   c = a;
else
   c = b;

int d = c;
```

```
int a;
int b;
...
if(a >= b)
{
   a /= 2;   //a=a/2
   b *= 2;   //b=b*2
}
else
{
   a *= 2;   //a=a*2
   b /= 2;   //b=b/2
}
```

Примеры инструкции if

Примеры кода

```
int a;
int b;
...

if(a < b)
{
   int c = a;
   a = b;
   b = c;
}

int d = a;
```

```
bool a;
bool b;
...
bool c = false;

if(a == true && b == false
|| b == true && a == false)

   c = true;


bool d = c;
```

Вложение if-else-if

С помощью ключевых слов **if** и **else** можно составлять так называемые **if-else-if** конструкции, которые могут осуществить проверку сразу нескольких условий.

Синтаксис вложенных инструкций

```
if(условие 1)
   инструкция 1;
else
   if(условие 2)
      инструкция 2;
   else
      if(условие 3)
         инструкция 3;
      else
         инструкция 4;
```

```
if(условие 1) {
   if(условие 2)
      инструкция 1;
   else
      инструкция 2; }
else {
   if(условие 3)
      инструкция 3;
   else
      инструкция 4; }
```

Пример вложения if-else-if

Пример кода

```
int a;
...
if(a >= 1000)
   printf("Очень много!\n");
else
   if(a >= 100 && a < 1000)
      printf("Много!\n");
   else
      if(a >= 10 && a < 100)
         printf("Мало!\n");
      else
         printf("Очень мало!\n");
```

Тернарный оператор ?:

Тернарный условный оператор **?:** имеет 3 аргумента и возвращает результат первого выражения если условие равно **true** или результат второго выражения если условие равно **false**.

```
Условие ? Выражение1 : Выражение2;
```

Пример кода

```
int b, c;
scanf("%d %d", &b, &c);

bool a = (b > c) ? true : false;

printf("a = %s\n",(a) ? "true" : "false");

((a) ? b : c) = 10;
```

Инструкция выбора switch

Управление передается в точку программы где целочисленное выражение совпало с **case**-константой и выполняется до оператора **break**. Если ни с одной из **case**-констант совпадения нет, то управление передается на конструкцию с **default**-меткой,
при условии ее наличия, в противном случае ни один из операторов не выполняется.

Синтаксис

```
switch(выражение)
{
case CONST1:  ...
      break;
case CONST2:  ...
      break;
default:  ...
}
```

Целочисленное выражение

Блок выполняется если выражение равно «CONST1»

Блок выполняется если выражение равно «CONST2»

Блок «по умолчанию»

Примеры инструкции switch

Пример кода

```
int note;
scanf("%d", &note);

switch(note) //выражение арифметического типа
{
case 1:
case 2: printf("Неудовлетворительно\n");break;
case 3: printf("Удовлетворительно\n"); break;
case 4: printf("Хорошо\n");  break;
case 5: printf("Отлично\n"); break;

default: printf("Ошибка\n");
}
```

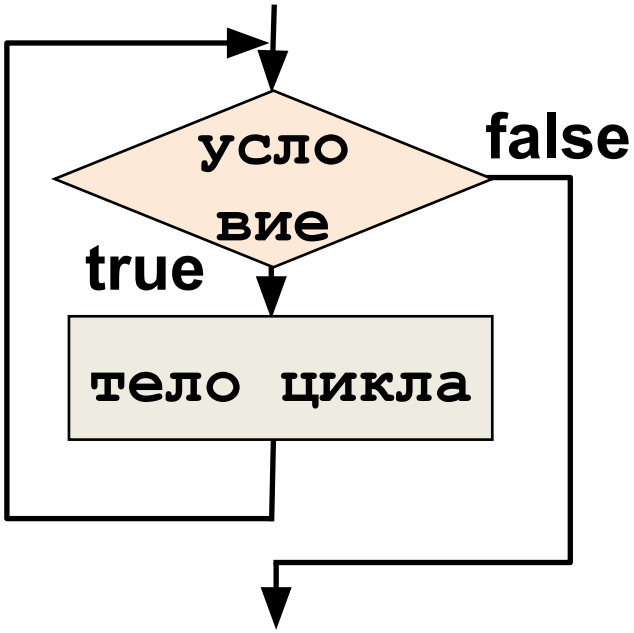
Цикл while

Цикл **while** является циклом с предпроверкой условия. Тело цикла будет многократно выполняться, пока условие будет иметь отличное от **false** значение.

Синтаксис

```
while(условие)
{
    тело цикла
}
```

принимает значение либо **true**, либо **false**



Пример с циклом while

Пример кода

```
int a = 10;

while(a--)
{
   printf("Сколько можно повторять!\n");
}
```

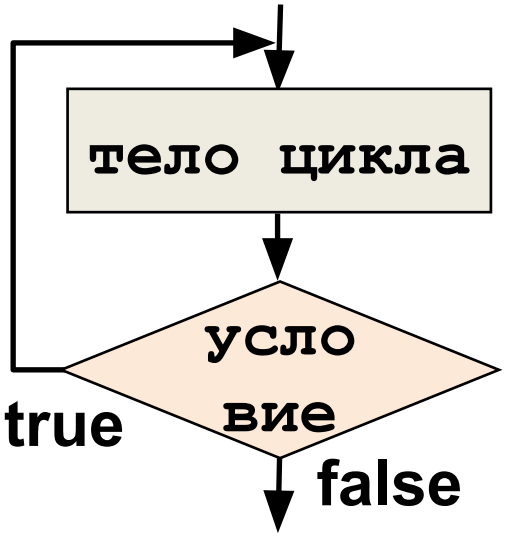
Цикл do-while

Цикл **do-while** является циклом с постпроверкой, условие проверяется после выполнения тела цикла. Следовательно, блок кода в цикле **do-while** выполниться хотя бы один раз. Цикл завершится, когда условие будет иметь значение равное **false**.

Синтаксис

```
do
{
    тело цикла
} while(условие);
```

принимает значение либо **true**, либо **false**



Пример с циклом do-while

Пример кода

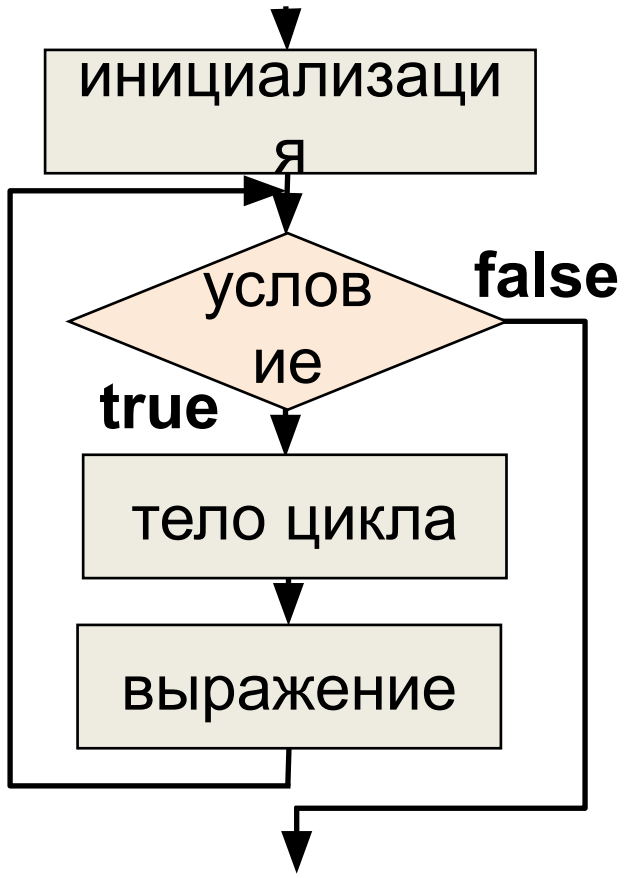
```
bool a = false;

do
{
   printf("Цикл выполняется хотя бы один раз!");
}
while(a);
```

Цикл for

Цикл **for** является циклом с предпроверкой условия. В цикле могут **инициализироваться** переменные до начала его выполнения, проверяться **условие** и выполняться **выражение** после каждого выполнения тела цикла.

Синтаксис

```
for(инициал.;условие;выражение)
{
    тело цикла
}
```



Пример с циклом for

Пример кода

```
int N = 10;
long f = 1;

for(int i = 1; i <= N; i++)
    f *= i;

printf("f = %ld\n", f);
```

Сравнение двух циклов

Пример с циклом while

```
double x = 5;
double y = 1;
int n = 10;

int i = 0;

while(i < n)
{
    y *= x;
    i++;
}
printf("y = %lg\n", y);
```

Пример с циклом for

```
double x = 5;
double y = 1;
int n = 10;

for(int i = 0; i < n;
i++)
    y *= x;

printf("y = %lg\n", y);
```

Инструкция continue

Инструкция **continue** пропускает все инструкции, оставшиеся до конца тела цикла, и начинает новую итерацию.

Пример кода

```
int N = 10;
long S = 0;

for(int i = 1; i <= N; i++)
{
   if(i%2) continue;

   S += i;
}
printf("Значение S : %ld\n", S);
```

Инструкция break

Инструкция **break** завершает выполнение цикла и передает управление на следующую за циклом инструкцию.

Пример кода

```
int N = 10;
long S = 0;

for(int i = 1; i <= N; i++)
{
   if(S > 10) break;

   S += i;
}
printf("Значение S : %ld\n", S);
```

Инструкция безусловного перехода goto

Инструкция **goto** позволяет реализовать передачу программного управления из одной точки программы в другую, отмеченную меткой. Метка состоит из идентификатора и завершающего двоеточия.

Пример кода

```
    ...
    a = b+c;
    goto M5;

    ...

M5:
    d = e-a;
    ...
```

«Обоснованное» применение инструкции goto

Пример кода

```
    while(true)
    {
     int N;
     long S = 0;
     scanf("%d", N);

     for(int i = 0; i <= N; i++)
     {
        S += i;
        if(S > 100) goto L1;
     }
     printf("Значение S : %ld\n", S);
    }
L1: // выход из двойного цикла
```