

# Тема: Операторы ветвления и циклы в программировании.



Операторы ветвления позволяют программе принимать решения на основе заданных условий.

# Тема: Операторы ветвления и циклы в программировании.



## Операторы ветвления

позволяют программе принимать решения на основе заданных условий.



# Тема: Операторы ветвления и циклы в программировании.

**if**

Оператор if используется для выполнения кода, если определенное условие истинно.



```
if условие:
```

```
    # код, который выполнится, если условие истинно
```

else

elif

# Тема: Операторы ветвления и циклы в программировании.

If

Пример.



```
возраст = 18  
if возраст >= 18:  
    print("Вы совершеннолетний")
```

else

elif

# Тема: Операторы ветвления и циклы в программировании.

**else**

Оператор else используется вместе с if для выполнения кода, когда условие ложно.



```
if условие:  
    # код, если условие истинно  
else:  
    # код, если условие ложно
```

If

elif

# Тема: Операторы ветвления и циклы в программировании.

else

Пример.



```
возраст = 16
if возраст >= 18:
    print("Вы совершеннолетний")
else:
    print("Вы несовершеннолетний")
```

If

elif

# Тема: Операторы ветвления и циклы в программировании.

**elif**

Оператор elif (сокращение от "else if") используется для проверки множества условий.



```
if условие1:
    # код, если условие1 истинно
elif условие2:
    # код, если условие2 истинно
elif условие3:
    # код, если условие3 истинно
else:
    # код, если все условия ложны
```


if

else

# Тема: Операторы ветвления и циклы в программировании.

elif

Пример.



```
оценка = 4
if оценка == 5:
    print("Отлично")
elif оценка == 4:
    print("Хорошо")
elif оценка == 3:
    print("Удовлетворительно")
else:
    print("Неудовлетворительно")
```

if

else



# Тема: Операторы ветвления и циклы в программировании.



Циклы позволяют выполнять необходимые участки кода множество раз.



```
for элемент in последовательность:
```

```
    # код, который выполняется для каждого элемента
```

# Тема: Операторы ветвления и циклы в программировании.



Циклы позволяют выполнять необходимые участки кода множество раз.



```
сумма = 0
for число in range(1, 101):
    сумма += число
print(f"Сумма чисел от 1 до 100: {сумма}")
```

# Тема: Операторы ветвления и циклы в программировании.



Циклы позволяют выполнять необходимые участки кода множество раз.



```
while условие:  
    # выполняется этот код
```

# Тема: Операторы ветвления и циклы в программировании.



Циклы позволяют выполнять необходимые участки кода множество раз.



```
a = 5
while a > 0:
    print(a)
    a -= 1
```

# Тема: Операторы ветвления и циклы в программировании.



break используется для немедленного выхода из цикла  
continue используется для перехода к следующей итерации цикла



```
for i in range(10):  
    if i == 5:  
        break  
    print(i)
```



```
for i in range(10):  
    if i % 2 == 0:  
        continue  
    print(i)
```

# Тема: Операторы ветвления и циклы в программировании.



Условные операторы и циклы могут быть вложенными друг в друга для создания более сложной логики.



```
возраст = 25
имеет_лицензию = True

if возраст >= 18:
    if имеет_лицензию:
        print("Вы можете водить автомобиль")
    else:
        print("Вам нужно получить водительские права")
else:
    print("Вы слишком молоды для вождения")
```

# Тема: Операторы ветвления и циклы в программировании.



Условные операторы и циклы могут быть вложенными друг в друга для создания более сложной логики.



```
for i in range(1, 6):  
    for j in range(1, 6):  
        print(f"{i} * {j} = {i*j}", end="\t")  
    print()
```

