Операторы Операторы break, continue, goto. Пустой оператор. (ОК)
План
1. break
2. continue
3. goto
4. Пустой оператор.

1. break
Оператор break может использоваться для принудительного выхода из циклов while, do-while и for.
Выход выполняется из ближайшего цикла или оператора switch.

```
for (d = 2; d < n; d++)
    if (n % d == 0)
        break;
```

2. continue
Оператор continue передает управление в конец цикла.

В циклах while и do-while это означает переход к проверке управляющего выражения,
а в цикле for — выполнение expr_3 и последующую проверку expr_2.
Оператор continue может использоваться только внутри циклов.

```
sum = 0;        i = 0;
while (i < 10)
{
    scanf(“%d”, &num);
    if (num < 0)
        continue;
    sum += num; i++;
}
```

3. goto
Оператор goto способен передать управление на любой оператор функции, помеченный меткой.
Метка — это идентификатор, расположенный вначале оператора:
идентификатор : оператор

Оператор может иметь более одной метки. Сам оператор goto записывается в форме

```
goto индетификатор;
// Определение "простоты" числа
int main(void)
{
    int d, n = 17;
    for (d = 2; d < n; d++)
        if (n % d == 0)
            goto done;

        done:
        printf("%d ", n);
    return 0;
}
```

Считается, что оператор goto источник потенциальных неприятностей.
Этот оператор на практике практически никогда не бывает необходим и почти всегда легче обходится без него.

Есть несколько ситуаций, в которых без goto удобно использовать.
Например, когда необходимо сразу выйти из двух и более вложенных циклов.

4. Пустой оператор.
Пустой оператор состоит только из символа «;». Основная «специализация» пустого оператора — реализация циклов с пустым телом:
Пустой оператор легко может стать источником ошибки:

```
if (d == 0);            // <-
```

```
printf(“ERROR: division by zero!\n”);
```