Смесени оператори за присвояване

- int a=0;
- += събиране a+=8 // a = a + 8
- -= изваждане a-=2 // a = a - 2
- *=умножение a*= 15 // a = a * 15
- /=деление a/=5 // a = a / 5
- %= остатък при деление a%=3 // a = a % 3

Логически оператори

- Работят само с булеви стойности
- && - дали и двете страни са истина
- || - дали поне една от двете страни е истина
- ! - Обратното на булевата стойност вдясно

Таблица на логическите операции

A	B	A&&B	A||B	!A	!B
True	True	True	True	False	False
False	True	False	True	True	False
False	False	False	False	True	True
True	False	False	True	False	True

Начин на работа на && и ||

```
bool error(){
        abort(); //функция, която гърми
        return true;
}
int main(){
false && error();

std::cout << "Did it explode?\n";
}
```

Начин на работа на && и ||

```
bool error(){
        abort(); //функция, която гърми
        return true;
}
int main(){
true && error();

std::cout << "Did it explode?\n";
}
```

Начин на работа на && и ||

```
bool error(){
        abort(); //функция, която гърми
        return true;
}
int main(){
true || error();

std::cout << "Did it explode?\n";
}
```

Начин на работа на && и ||

```
bool error(){
        abort(); //функция, която гърми
        return true;
}
int main(){
false || error();

std::cout << "Did it explode?\n";
}
```

Начин на работа на && и ||

- Ако левият bool на && е false, && връща false, без да проверява десния
- Ало левият bool на || е true, || връща true, беаз да проверява десния

Релационни оператори

- Служат за сравняване на 2 стойности
- == дали двете са равни
- != дали двете са различни
- >   дали лявата е по-голяма от дясната
- <   дали лявата е по-малка от дясната
- >= дали лявата е по-голяма или равна на дясната
- <= дали лявата е по-малка или равна на дясната

- При сравняване на литерал, ще се вземе числената им стойност (виж ASCII таблицата)

Единични оператори

- Работят само с 1 променлива
- Примери:

  ++

  --

  +

  -

- Други, които днес няма да разглеждаме, но скоро ще ви вкарат в ада: &, *, new, delete

++ и --

- Ще разгледаме само ++, всичко при -- е аналогично

- Има два различни оператора ++
  - Prefix (пред името на променливата)
  - Postfix (след името на променливата)

- И двата увеличават стойността на променливата с 1, но връщат различни резултати

Разлика между prefix и postfix

- Prefix връща стойността на променливата след инкрементиране
- Postfix връща стойността на променливата преди инкрементиране

- Prefix връща lvalue
- Postfix връща rvalue

Оператори + и – (отново)

- + и – могат да се използват и само върху един елемент

- Единичните оператори +/- влияят върху знака на числовите стойности

- int a = 50;
- a = -a; //a ще придобие стойност -50
- a = +a; //a ще си запази стойността

Оператор ? :

- Основен синтаксис:

<булева стойност> ? <израз 1> : <израз 2>

- Ако <булева стойност> е true, операторът ще върне <израз 1>, иначе ще върне <израз 2>

- !!!<израз 1> и <израз 2> трябва да са от един и същ вид, иначе ще се получи грешка при компилация

Примери с оператор ? :

- int a = 1 ? 2 : 3; // a ще получи стойност 2, зле написан код
- int a = (0 ? 2 : 3); // a ще получи стойност 3, четлив код
- int a = 5 + (5<4 ? 2 : 3); //a ще получи стойност 8
- Принтиране на по-малка променлива

```
std::cout<<( a < b ? a  : b );
```