Условен оператор if. Тернарен оператор

I. Условен оператор if

Кога го използваме? – когато искаме даден фрагмент от програмата ни да се изпълни само ако конкретно условие е удовлетворено.

Пр.: а) Да се въведе цялото число X. Ако то е четно, да се изведе неговият адрес.
б) Да се въведат две десетични числа X и Y и да се намери сумата им. Ако сумата им се дели на 3, да се изведе „Sum is divisible by 3.". Ако сумата им не се дели на 3, да се изведе „Sum is not divisible by 3."
в) Да се въведе символ. Ако той е латинска буква, да се изведе „Latin Letter", а ако е цифра, да се изведе „Digit". Иначе да се изведе „Not latin letter or digit".

Под синтаксис на оператор се има предвид начинът, по който се пише, а под семантика - начинът, по който операторът работи.

Синтаксис: условният оператор if има 3 вариации в C++:

- **if (<условие>) {**
  <оператори>;
  }, където: <условие> е булев израз,
  <оператори> е последователност от операции, които искаме да се извършват в случай, че условието е изпълнено.

Пр.:
```
//а)
int a = 2;
if(a < 3) {
        cout << "a < 3" << endl; //a < 3
}

//б)
int a = 3;
if(a == 3) {
        cout << "a == 3" << endl; // a == 3
        a = a + 2;
        cout << "a == " << a << endl; // a == 5
}

//в)
int a = 5;
if(a > 5) {
        cout << "a < 5" << endl; //на конзолата не се изписва нищо, защото условието a > 5 не е изпълнено;
}

//г)
int a = 1;
int b = 4;
if(a == 2 || b == 4) {
        cout << a + b; // 5
}
```

- **if (<условие>) {**
  <оператори$_1$>;
  } else {
  <оператори$_2$>;
  }, където: <условие> е булев израз,
  <оператори$_1$> е последователност от операции, които искаме да се извършват, в случай, че условието е изпълнено,
  <оператори$_2$> е последователност от операции, които искаме да се извършват, в случай, че условието НЕ Е изпълнено;

Пр.:
```
//а)
int a = 3;
if(a < 3) {
        cout << "a < 3" << endl; //a < 3
} else {
        cout << "a >= 3" << endl;
}

//б)
int a = 5;
if(a > 5) {
        cout << "a < 5" << endl;
} else {                        //за разлика от в) на предишния пример, тук на конзолата се изписва
        cout << "a >= 5" << endl; // a >= 5;
}

//в)
int a = 1;
int b = 4;
if(a == 2 && b == 4) { // false, т.к. a == 1
        cout << a + b;
} else {
        cout << a - b; // -3
}
```

- **if (<условие$_1$>) {**
  <оператори$_1$>;
  } else if (<условие$_2$>) {
  <оператори$_2$>;
  } else if … {
  …
  } else {
  <оператори$_n$>;
  }, където : <условие$_i$> е булев израз;

<оператори$_i$>, i != n, е последователност от операции, които искаме да се извършват, в случай, че <условие$_i$> е изпълнено; в момента, в който едно от условията е изпълнено и се изпълнят съответните му оператори, проверката на if-а приключва (вж. д))

<оператори$_n$> е последователност от операции, която се изпълнява, ако нито едно от условията преди else-а не е било изпълнено.

Пр.:
```
//а)
int num = 1;
if (num < 2) {
        cout << "num < 2";
} else if (num == 2) {
        cout << " num == 2";
} else if (num == 3) {
        cout << "num == 3";
} else {
        cout << "else";
}

//б)
int num = 2;
if (num < 2) {
        cout << "num < 2";
} else if (num == 2) {
        cout << " num == 2"; // num == 2
} else if (num == 3) {
        cout << "num == 3";
} else {
        cout << "else";
}
```

```
//в)
int num = 3;
if (num < 2) {
        cout << "num < 2";
} else if (num == 2) {
        cout << " num == 2";
} else if (num == 3) {
        cout << "num == 3"; // num
} else {
        cout << "else";
}

//г)
int num = 4;
if (num < 2) {
        cout << "num < 2";
} else if (num == 2) {
        cout << " num == 2";
} else if (num == 3) {
        cout << "num == 3";
} else {
        cout << "else"; //else
}
```

```
//д)
int num = 2;
if (num == 2) {
        cout << "num == 2!!!"; // num == 2!!!
} else if (num == 2) {          // не влиза тук
        cout << "num == 2???";
} else if (num == 3) {
        cout << "num == 3";
} else {
        cout << "else";
}
```

Забележка. Операторите при if заграждаме в „{}“ задължително, ако са повече от един и по желание, ако са точно 1. Добрият стил изисква все пак винаги да използваме „{}“.

Пр.:

```
//Good practice
if (a < 5) {
        cout << a;
}

//Also O.K. practice
if (a < 5)
{
        cout << a;
}
```

```
//Bad practice
if (a < 5) { cout << a; }

//Also bad practice
if (a < 5)
        cout << a;

//Bad practice №3
if (a < 5) cout << a;
```

Препоръките на Google Style Guide за форматиране на условния оператор:
https://google.github.io/styleguide/cppguide.html#Conditionals

Полезно за условния оператор if:
https://www.youtube.com/watch?v=Zfm-138maOE&ab_channel=CalebCurry, 1:22 – C++ if operator for beginners
https://www.youtube.com/watch?v=qEgCT87KOfc&ab_channel=TheCherno – branching in C++ explained on a deeper level

II. Тернарен оператор

Кога го използваме? – когато имаме една кратка операция, която искаме да се извърши в случай, че условието е изпълнено и още една кратка операция, която да се извърши в случай, че условието не е изпълнено. Използваме го, когато имаме нужда от if...else, но операторите на if и на else едноредови и кратки.
Важно! Тернарният оператор връща резултат, т.е. можем да присвоим резултата от тернарен оператор на променлива (вж. пр.).

Синтаксис:

- **(<условие>) ? <оператор$_1$> : <оператор$_2$>**
  Ако условието е изпълнено, се изпълнява <оператор$_1$>, ако условието не е изпълнено, се изпълнява <оператор$_2$>.

Най-често го използваме във варианта:

- **<променлива> = (<условие>) ? <стойност$_1$> : <стойност$_2$>**
  Ако условието е изпълнено, на променливата се присвоява <стойност$_1$> , ако условието не е изпълнено, се на променливата присвоява <стойност$_2$>.

Пр.:

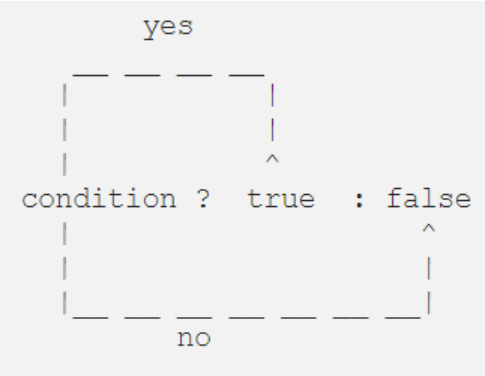
```
//а)
int x = 42;
bool y = (x == 42) ? true : false;
cout << "y == " << y; //y == 1

//б)
int x = 44;
bool y = (x == 42) ? true : false;
cout << "y == " << y; //y == 0

//в)
int a = 5;
int b = 3;
int larger = (a > b) ? a : b;
cout << "larger == " << larger; // larger == 5
```

```
//в) е еквивалентно на записа:
int a = 5;
int b = 3;
int larger;
if ( a > b) {
        larger = a;
} else {
        larger = b;
}
cout << "larger == " << larger; // larger == 5
```

Нагледно как работи тернарният оператор:



Името на тернарния (ternary) оператор (от латински "ternarius" = „съставен от три елемента") произлиза от броя на операндите, върху които се прилага – булево условие (първи операнд), оператор₁ (втори операнд) и оператор₂ (трети операнд). Аналогично в C++ имаме унарни оператори (--, ++ -> прилагат се върху един операнд; пр. i++, --i) и бинарни оператори (+, - и др. -> прилагат се върху два операнда; пр. a + b, a – b, etc.).Тях, естествено, ги наричаме с интуитивните им имена, вместо с „бинарен плюс!"/ "бинарен минус!"/„бинарно деление!" и пр.

Полезно за тернарния оператор:
https://medium.com/@jraleman/ternary-operators-vs-if-else-statements-6c26f7d034f7

III. Таблица с условните оператори

Оператор	Синтаксис	Семантика	Пример

if	**if (<условие>) {**
	<оператори>;
	}
		Ако условието е изпълнено, се изпълняват <оператори>.	int a = 2;
			if (a < 3) {
			cout << "a < 3" << endl; // a < 3
			}

if...else	**if (<условие>) {**
	<оператори₁>;
	} else {
	<оператори₂>;
	}
		Ако условието е изпълнено, се изпълняват <оператори₁>, иначе се изпълняват <оператори₂>.	int a = 4;
			if (a < 3) {
			cout << "a < 3" << endl;
			} else {
			cout << "a >= 3" << endl; // a>=3
			}

if...else if...else	**if (<условие₁>) {**
	<оператори₁>;
	} else if (<условие₂>) {
	<оператори₂>;
	} else if ... {
	...
	} else {
	<операториₙ>;
	}
		Ако <условиеᵢ> е изпълнено, се изпълнява <операториᵢ> и изпълнението на if-а приключва. Ако нито едно от условията не е изпълнено, се изпълняват <операториₙ>.	int a = 4, b;
			if (a > 4) {
			b = 5;
			} else if (a < 4) {
			b = -5;
			} else {
			b = 0; // b == 0
			}

ternary	**(<условие>) ? <оператор₁> : <оператор₂>**	Ако условието е изпълнено, се изпълнява <оператор1>, иначе се изпълнява <оператор2>.	

	<променлива> = (<условие>) ? <стойност₁> : <стойност₂>
	(частен случай на горния синтаксис)
		Ако условието е изпълнено, на променливата се присвоява <стойност1>, иначе на променливата се присвоява <стойност2>.	int a = 5;
			int b = 3;
			int larger = (a > b) ? a : b; // larger == 5