I. Оператор switch

Кога го използваме? – когато имаме някаква променлива и предварително ни е известно множеството от стойностите, които тя може да приема. Според стойността на променливата искаме да изпълним различни действия в програмата си.

Пр.: а) Да се въведе цяло число number от ***1 до 12***. Да се изведе ***името на месеца***, на който съответства числото . – **Обяснение**: *известно ни е, че number приема стойности от 1 до 12. Искаме програмата да извърши различни действия спрямо това коя стойност наистина приема променливата ни – ако number == 1, искаме да изведем „Януари"; ако number == 2, искаме да изведем „Февруари" и т.н.*

б) Да се въведе символ symbol, за който знаем, че е измежду главните латински букви ***A, B, C, D*** и ***E***. Ако символът е A, да се изведе ***„symbol is A"***; ако символът е B, да се изведе ***„symbol is B"*** и аналогично за другите символи. - **Обяснение**: *известно ни е, че symbol приема стойности от множеството {A, B, C, D, E}. Искаме програмата да извърши различни действия спрямо това коя стойност наистина приема променливата ни – ако symbol == `A`, ще изведем A; ако symbol == `B`, ще изведем B и т.н.*

в) Да се въведе цяло число number. Ако то е ***0, 1, 2 или 3***, да се изведе ***„number is smaller than 4"***. Иначе ***(т.е. 4, 5, 6,…)***, да се изведе ***„number is bigger than 4"***. - **Обяснение**: *известно ни е, че number приема стойности от множеството {0, 1, 2, 3, 4, 5, 6,…}. Искаме програмата да извърши различни действия спрямо това коя стойност наистина приема променливата ни – ако number == 0, ще изведем "number is smaller than 4"; ако number == 1, ще изведем отново "number is smaller than 4", и т.н., докато ако number == 4, ще изведем "number is bigger than 4"; ако number == 5, ше изведем "number is bigger than 4", etc.*

Синтаксис:

- **switch (<израз_с_който_сравняваме>) {**
  case <израз$_1$> : <оператори$_1$>; break;$_{опц.}$
  case <израз$_2$> : <оператори$_2$>; break;$_{опц.}$
  …
  case <израз$_n$> : <оператори$_n$>; break;$_{опц.}$
  {default: <оператори$_{n+1}$>; }$_{опц.}$

}, където: < израз_с_който_сравняваме > е израз, според чиято стойност ще се изпълнят case-овете (напомняме, че сама променлива също е израз);

case <израз$_i$> - case е запазена дума; <израз$_i$> е константен израз (т.е. не съдържа променливи), който съответства на някоя от очакваните стойности за израза (напомняме, че сама константа също е израз);

<оператори$_i$> е последователност от операции, които искаме да се извършват в случай, че <израз_според_който_ще изменяме> == <стойност$_i$>;

break е запазена дума. Ако

default: <оператори$_{n+1}$> - default е запазена дума; <оператори$_{n+1}$> е последователност от операции, които ще се извършат в случай, че стойността на израз-а не е посочена в нито един от <израз$_i$>.

Семантика:

Напомняме, че семантика означава това как работи операторът.

1. Програмата влиза в switch-а. Изчислява стойността на <израз_с_който_сравняваме>.
2. Започва да върви ред по ред по case-вете. Влиза в първия case.
   - Ако стойността на <израз$_1$> == стойността на <израз_с_който_сравняваме >, се изпълняват <оператори$_1$>.
     - Ако на края на <оператори$_1$> има думата „break", изпълнението на switch-а приключва.
     - Ако на края на <оператори$_1$> няма думата „break", се изпълняват ВСИЧКИ ОПЕРАТОРИ ОТ КЕЙСОВЕТЕ ПОД ПЪРВИЯ КЕЙС, докато не

се срещне break. Ако никъде не се срещне break, ще се изпълнят абсолютно всички оператори под този кейс, включително тези на default.

- Ако стойността на <израз$_1$> != стойността на <израз_с_който_сравняваме >, проверката преминава към следващия case.

В общия случай, при влизане в case <израз$_i$> се изпълнява:

- Ако стойността на <израз$_i$> == стойността на <израз_с_който_сравняваме >, се изпълняват <оператори$_i$>.
  - Ако на края на <оператори$_i$> има думата „break", изпълнението на switch-а приключва.
  - Ако на края на <оператори$_i$> няма думата „break", се изпълняват всички оператори от кейсовете под i-тия кейс, докато не се срещне break. Ако никъде не се срещне break, ще се изпълнят абсолютно всички оператори под този кейс, включително тези на default.
- Ако стойността на <израз$_i$> != стойността на <израз_с_който_сравняваме>, проверката преминава към следващия case.

3. Ако стойността на <израз_с_който_сравняваме> не е равна на нито една от стойностите на <израз$_i$>, се изпълнява операторът на default.

Пр.:

```
//а) Резултат на конзолата: Първи случай
int value = 1;
switch(value) {
        case 1: cout << "Първи случай"; break;
        case 2: cout << "Втори случай"; break;
        default: cout << "По подразбиране";
}
```

```
//б) Резултат на конзолата: Първи случайВтори случай
int value = 1;
switch(value) {
        case 1: cout << "Първи случай";
        case 2: cout << "Втори случай"; break;
        default: cout << "По подразбиране";
}
```

```
//в) Резултат на конзолата: Първи случайВтори случайПо подразбиране
int value = 1;
switch(value) {
        case 1: cout << "Първи случай";
        case 2: cout << "Втори случай";
        default: cout << "По подразбиране";
}
```

```
//г) Резултат на конзолата: По подразбиране
int value = 5;
switch(value) {
        case 1: cout << "Първи случай"; break;
        case 2: cout << "Втори случай"; break;
        default: cout << "По подразбиране";
}
```

Операторът switch много прилича на if…else if… else вариацията на if. Всъщност винаги когато можем да използваме if…else if…else, можем да използваме switch на негово място. Тогава, ако можем да използваме if…else if…else вместо switch, защо бихме прибягнали до switch?

Предимства на switch пред if…else if…else	Пример

Доста **по-четимо** и **кратко** от if…else if… else. Това се забелязва най-вече при по-дълги програми.	`//проверка за сезон според месеца:`
	`int month;`
	`cin >> month;`
	`switch (month) {`
	`  case 12, 1, 2: cout << "Winter is nasty."; break;`
	`  case 3: cout << "March is spring but is great."; break;`
	`  case 6, 7, 8: cout << "Summer is great."; break`
	`  default: cout << "Autumn and spring are so-so."; brek;`
	`}`
	
	`//същото, но с if:`
	`int month;`
	`cin >> month;`
	`if (month == 1 || month == 2 || month == 12) {`
	`  cout << "Winter is nasty.";`
	`} else if (month == 6 || month == 7 || month == 8) {`
	`  cout << "Summer is great.";`
	`} else if ( month == 3) {`
	`  cout << "March is spring but is great.";`
	`} else {`
	`  cout << "Autumn and spring are so-so.";`
	`}`

Удобен за използване на **enum**-и	
По-бързо изпълнение на програмата	

Недостатъци на switch пред if…else if…else	Пример

Не може да се използва за float променливи (нито в switch-а може да има float променлива, нито в case-а може да има float константа).	`//няма да се компилира`
	`float a = 3.4;`
	`switch (a) {`
	`  case 4.5: cout << "Larger than 4"; break;`
	`  default: cout << "Not larger than 4";`
	`}`

Не може да имаме **израз**, който **няма костантна стойност**, за case. Иначе казано, не може да има променлива в case-а.	`//това ще се компилира - в кейса имаме израз,`
	`//който съдържа само 2 константи (2 и 1):`
	`int a = 3;`
	`switch (a) {`
	`  case 2 + 1: cout << "Is 3."; break;`
	`  default: cout << "Is not 3.";`
	`}`
	
	`//това няма да се компилира - в кейса имаме израз,`
	`//който съдържа променлива (b), освен константа (0):`
	`int b = 2;`
	`switch (b) {`
	`  case b + 0: cout << "Is 2."; break;`
	`  default: cout << "Is not 2.";`
	`}`

Не може да имаме една и съща константа в 2 case-а.	`//няма да се компилира - има 2 кейса с константата 2;`
	`int b = 2;`
	`switch (b) {`
	`  case 2: cout << "Is not 1."; break;`
	`  case 2: cout << "It is not 3."; breakl`
	`  default: cout << "Is not 2.";`
	`}`

Трябва да следим за разклонението на програмата по-внимателно, заради нуждата от поставяне на **break**-ове и **default**-и.	

Полезно за switch:
https://www.youtube.com/watch?v=_CelY_ZBXb4&ab_channel=thenewboston
https://www.youtube.com/watch?v=sQkgGd7PEfM&ab_channel=TutorialsPoint%28India%29Ltd. – за готин индийски акцент. ☺

Забележка. На практика switch statement-а се използва като условен оператор, затова можем да го причислим към групата на if и ternary:

II. 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>) {**
	<оператори$_1$>;
	} else {
	<оператори$_2$>;
	}
		Ако условието е изпълнено, се изпълняват <оператори$_1$>, иначе се изпълняват <оператори$_2$>.	int a = 4;
			if (a < 3) {
			cout << "a < 3" << endl;
			} else {
			cout << "a >= 3" << endl; // a>=3
			}

if…else if…else	**if (<условие$_1$>) {**
	<оператори$_1$>;
	} else if (<условие$_2$>) {
	<оператори$_2$>;
	} else if … {
	…
	} else {
	<оператори$_n$>;
	}
		Ако <условие$_i$> е изпълнено, се изпълнява <оператори$_i$> и изпълнението на if-а приключва. Ако нито едно от условията не е изпълнено, се изпълняват <оператори$_n$>.	int a = 4, b;
			if (a > 4) {
			b = 5;
			} else if (a < 4) {
			b = -5;
			} else {
			b = 0; // b == 0
			}

ternary	**(<условие>) ? <оператор$_1$> : <оператор$_2$>**	Ако условието е изпълнено, се изпълнява <оператор1>, иначе се изпълнява	

		<оператор2>.	

	<променлива> = (<условие>) ? <стойност$_1$> : <стойност$_2$>
	(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

switch	**switch (<израз_с_който_сравняваме>) {**
	case <израз$_1$> : <оператори$_1$>; break;$_{опц.}$
	case <израз$_2$> : <оператори$_2$>; break;$_{опц.}$
	…
	case <израз$_n$> : <оператори$_n$>; break;$_{опц.}$
	{default: <оператори$_{n+1}$>; }$_{опц.}$
	}
		При влизане в case <изрази>, ако <изрази> ==<израз_с_който_сравняваме >, се изпълняват <операториі>. Ако на края на <операториі> има думата „break", изпълнението на switch-а приключва. Ако на края на <операториі> няма думата „break", се изпълняват всички оператори от кейсовете под i-тия кейс, докато не се срещне break. Ако никъде не се срещне break, ще се изпълнят абсолютно всички оператори под този кейс, включително тези на default. Ако стойността на <изрази> != стойността на <израз_с_който_сравняваме>, проверката преминава към следващия case. Ако стойността на <израз_с_който_сравняваме> не е равна на нито една от стойностите на <изрази>, се изпълнява операторът на default.	int value = 1;
			int counter = 1;
			switch(value) {
			case 2: counter += 2;
			case 1: counter++;
			case 3: counter += 5;
			default: counter++;
			}
			cout << counter << endl; //counter==8