

**Algoritmus** – přesný postup / návod, jak vyřešit daný typ úlohy pomocí symbolů.

### Vlastnosti algoritmů

#### 5 kritérií

Konečnost – každý algoritmus musí skončit v konečném počtu kroků. Pro každý vstup musí být konečný

Obecnost – Neřeší jeden konkrétní problém, ale obecnou třídu podobných problémů

Determinovanost – každý krok musí být jednoznačně a přesně definován



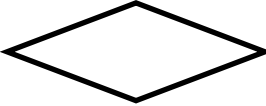




Výstup – má alespoň jeden výstup, veličinu, která je v požadovaném vztahu k zadaným vstupům

Elementárnost – skládá se z konečného počtu jednoduchých kroků

#### Metody návrhu

Shora dolů – postup řešení rozkládáme na jednodušší operace, až dospějeme k jednoduchým krokům

### Kreslení vývojových diagramů

	Konec a začátek algoritmu
	Běžný příkaz
	Podmínka
	Cyklus s počtem opakování
	Cyklus s podmínkou na začátku, cyklus s podmínkou na konci
	Zobrazení výstupů
	Podprogram
	Spojovací značka, spojovací čára

## Přepis algoritmu z vývojového diagramu do jazyka C

Deklarace proměnné/konstanty: `int numA = 1;`      `float numA = 1,5;`

začátek: `int main(void)` nebo pouze `int main()`

podmínky: `if (podmínka) {akce – podm. splněna} else {akce – podm. nesplněna}`

opakování: `for (int i = 0; i < 10; i++)`

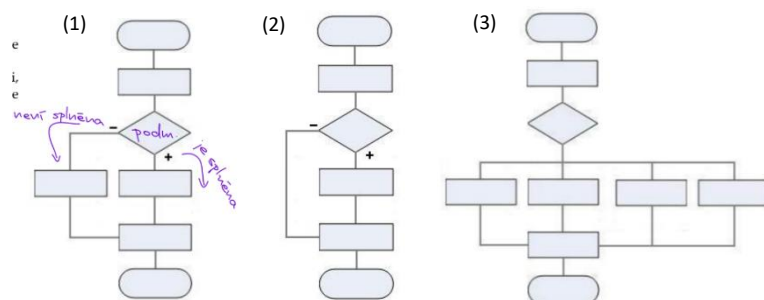
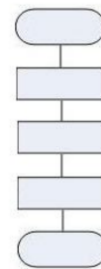
konec: `return 0;`

## **Základní programové struktury**

Sekvence: jednoduchý algoritmus, příkazy jdoucí po sobě

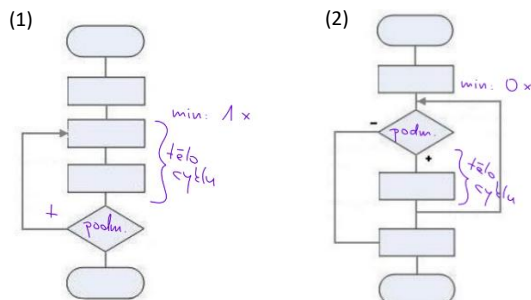
Větvení:

- (1) Úplná alternativa – příkazy v obou větvích
- (2) Neúplná alternativa – příkazy pouze v jedné větvi
- (3) Několikanásobná – podmínka + výběr



Cykly: neustálé opakování cyklů

- Se známým počtem opakování – for
- S neznámým počtem opakování – (1) s podmínkou na konci – do while
- (2) s podmínkou na začátku – while



## Programovací jazyk C

- Programovací jazyk, který může být jak vyšší (vysoká míra abstrakce) tak nižší („blízko k HW“).
- Je kompilovaný, to znamená, že zdrojový kód musí být nejprve kompilován, aby se z něj stal strojový kód.
- Case sensitive – rozlišuje malá a velká písmena v názvech proměnných, konstant, funkcí a hodnot
- Určuje: množinu hodnot (minimum a maximum)  
povolené operace  
velikost místa v paměti (kolik Bajtů)

### Praktický příklad – Replit C-25

```
1 //Nadeklaruj funkci f_najdiMaximum() datového typu int
2 //Funkce bude mít tři formální parametry "a, b, c" datového typu int
3 //Funkce vrátí návratovou hodnotu maximum z těchto tří čísel
4 //Napiš hlavní program, ve kterém uživatel zadá postupně tři celá čísla, zavolej tuto funkci a
  předej jí tyto skutečné parametry
5 //Funkce vrátí do hlavního programu hodnotu maxima, kterou v hlavním programu vypíšeš
6
7 #include <stdio.h>
8
9 int cislo1, cislo2, cislo3, maximum, soucet;
10
11 int f_najdiMaximum(int a, int b, int c)
12 {
13     if(a > maximum) maximum = a;
14     if(b > maximum) maximum = b;
15     if(c > maximum) maximum = c;
16     return maximum;
17 }
18
19 int main(void)
20 {
21     printf("Napiš tři čísla: \n");
22     scanf("%d %d %d", &cislo1, &cislo2, &cislo3);
23
24     printf("\n");
25
26     soucet = f_najdiMaximum(cislo1, cislo2, cislo3);
27     printf("Největší číslo je: %d", soucet);
28
29     return 0;
30 }
```