Maturitní témata 2020/21

Otázka č. 25 – Počítačové sítě a programování

Podmínky a cykly v různých jazycích:

- cykly - něco co se bude opakovat, když chceme provést část kódu víckrát, máme dva druhy cyklů - cyklus while a cyklus for

- podmínka - podmínka se zapisuje příkazem if, jako podmínku můžeme použít libovolný logický výraz, podmíněně můžeme vykonat také více příkazů

While:

- cyklus, na rozdíl od cyklu for kde předem známe počet opakování se while používá když cyklus závisí na nějaké podmínce, tělo cyklu se opakuje dokud je podmínka splněna

- pomocí while lze udělat i cyklus for protože for je speciální případ while, cyklus while se ale používá na trochu jiné věci, často máme v jeho podmínce např. funkci vracející logickou hodnotu true/false

- např. v pythonu

```
odpoved = input('Řekni Ááá! ')
while odpoved != 'Ááá':
    print('Špatně, zkus to znovu')
    odpoved = input('Řekni Ááá! ')
```

- v programech jazyku C se nachází ještě cyklus do-while, je velmi podobný jako while ale je tu jeden rozdíl → příkaz (nebo blok příkazů) se provede vždy alespoň jednou, podmínka se kontroluje až poté

```
do
{
    // příkazy
} while ( /* podmínka */ );
```

For:

- cyklus, řídící struktura počítačového programu a je svou činností podobný cyklu while-do s testováním podmínky na začátku cyklu

- typicky se cyklus skládá z inicializátoru, podmínky, inkrementu a těla cyklu, v různých programovacích jazycích existují různé modifikace for cyklu kde je např. místo inicializátoru, podmínky a inkrementu uveden výčet hodnot které se budou přiřazovat proměnné

- tento cyklus má stanovený pevný počet opakování a hlavně obsahuje tzv. řídící proměnnou (celočíselnou) ve které se postupně během běhu cyklu mění hodnoty, syntaxe (zápis) for může být např. v C#:

```
for (promenna; podminka; prikaz)
```

→ promenna je řídící proměnná cyklu, které nastavíme počáteční hodnotu (nejčastěji 0), např. int i = 0

→ podminka je podmínka vykonání dalšího kroku cyklu, jakmile nebude platit cyklus se ukončí, např. (i<10)

→ prikaz nám říká co se má v každém kroku s řídící proměnnou stát, tedy zda se má zvýšit nebo snížit, k tomu využijeme speciálních operátorů ++ a -- (slouží k zvýšení nebo snížení proměnné o 1)

Jazyk C / C++ / Java / C# používá konstrukce *inicializátor*, *podmínka*, *inkrement* a *tělo cyklu*:

```
for ( i=0; i<N; i++ ) { // inicializátor; podmínka; inkrement
  // tělo cyklu
}
```

V jazyku PHP můžeme použít i více variant:

- podobně jako v jazyku C (inicializátor, podmínka, inkrement):

```
for ( $i=0; $i<$N; $i++ ) {
   // tělo cyklu
}
```

- nebo užitím `foreach` pro všechny prvky určitého pole:

```
$a = array(1,2,5,8);
foreach ( $a as $index=>$hodnota ) {
  echo "index=$index, hodnota=$hodnota\n";
}
```

If:

- podmínka, vykoná nějakou funkci jen pokud je splněná určitá podmínka, podmínky slouží k větvení programu

- podmínka if je nejjednodušší z podmínek, pokud je splněná podmínka něco se vykoná (pokud podmínka splněná není nevykoná se nic)

```
if (podminka) {
        //provedou se příkazy, které se provedou když platí podmínka
        prikazy;
}
```

- dalším typem podmínky if je podmínka s alternativou (if - else), pokud je splněná podmínka něco se vykoná, pokud není splněna vykoná se něco jiného

```
if (podminka) {
        //provedou se příkazy, které se provedou když platí podmínka
        prikazy;
} else {
        //provedou se alternativní příkazy, které se provedou když neplatí podmínka
        alternativniPrikazy;
}
```

- další možností je podmínka s více alternativami

```
if (Podmínka1) {
// co se má vykonat pokud je podmínka 1 splněna
}
else if (Podmínka2) {
// co se má vykonat pokud je podmínka 2 splněna
}
else {
// co se má vykonat pokud podmínka 1 ani podmínka 2 neplatí

}
```

- v podmínkách se setkáváme s různými operátory:

```
> //je větší než
=> //je rovno nebo větší než
< //je menší než
=< //je rovno nebo menší než
== //je rovno
!= //je různé, nerovná se
```

- podmínka může mít v sobě jakýkoliv kód i další podmínku, podmínky mohou být takto hierarchicky vložené

- podmínky je možné skládat a to pomocí dvou základních operátorů:

A zároveň	&&
Nebo	||

Pass:

- klíčové slovo Pythonu pass se používá v podmínkách (nebo kdekoliv jinde), přičemž znamená, že daná část kódu nic nedělá

```
slovo = input("\nNapište Python pro ukončení: ")
if (slovo == "Python" or slovo == "python"):
    print("\nSlovo zadáno!")
    pokracovat = False
else:
    pass # nic se nestane
```

Continue:

- příkaz continue je podobný break, používá se však k ukončení pouze aktuálního průběhu cyklu a ne celého cyklu, cyklus poté rovnou přechází na další průběh, použití continue můžeme najít např. při validování (kontrolování) položek při procházení nějaké kolekce

Break:

- příkaz break ukončuje aktuální cyklus, používá se nejčastěji pokud pomocí cyklu nalezneme nějakou položku v kolekci a dále již v jejím procházení nechceme pokračovat, nebudeme tak dále zbytečně prohledávat zbytek kolekce když již máme to co jsme hledali