**for cyklus** int i, j; (1)Nejprve se provede inicializace, potom

for (i = 1; i <= 5; i++) (2) ověří se podmínka.nepravda cyklus se ukončí. V opačném případě se

for (j = 1; j <= 5; j++) (3)provede příkaz/blok,

printf("[%i, %i]%s", i, j, j == 5 ? "\n" : " "); (4)provede krok a pokračuje se zpět k podmínce.

**do - while cyklus**

int i = 1;

do {

i++;

printf("%i\n", i);

} while (i <= 10);

Rozdíl oproti while cyklu je jen v tom, že se nejprve vykoná tělo cyklu a teprve potom se zkontroluje podmínka. To znamená, že tělo se vždy alespoň jednou provede.

**while cyklus** int i = 1; while (i <= 10) {

printf("%i\n", i); i++; }

**Podmínka**

Podmínka má syntaxi

if (výraz) příkaz1 nebo blok1 [ else příkaz2 nebo blok2 ]

Příklad: unsigned synacku = 5, jablicek = 11; if (synacku > 0) printf("Každý synáček dostane %u jablíček " "a %u jich zbude na tatínka.\n", jablicek / synacku, jablicek % synacku); else { puts("Tatínek má 0 synů."); puts("Raději nebudeme dělit nulou.");

**Logické operátory**

**Vykřičník (!)** je negace (not/ne)

**&&** má význam konjunkce (and/a)

**||** je alternativa (or/nebo). • Logické xor C nemá

Operátory && a || zaručují pořadí vyhodnocování. Nejprve se vyhodnotí levý operand a vyhodnocení pravého následuje pouze pokud je to nezbytné pro vypočtení hodnoty výrazu.

**Funkce printf**

**printf("Ahoj světe\n");** vypíše ahoj světe s odřádkováním

**puts("Ahoj světe");** rovnou odřádkuje bez potřeby /n

**printf("Minus sto dvacet pět je %i\n", i);** výpis čísla typu int

**printf("Jedeme na %i%%\n", 100);** vypsání procenta

**printf("Znak '%c' má ASCII kód %i, šestnáctkově %x\n", 'm', 'm', (unsigned) 'm');** ASCII kódu ve dvou soustavách

**printf("10 pi je asi %f, což se dá napsat i jako %e\n", d, d);**

Výpis reálných čísel

**printf("Vnořená printf vypsala %i znaků\n", printf("12345\n"));** Funkce printf vrací, kolik toho vytiskla

Constanta se nemění const float PI = 3.14

**Printf – tabulka parametrů**

**i -** int, znaménkové číslo v desítkové soustavě

**o, u, x, X -** unsigned, nezáporné celé číslo v osmičkové, desítkové, šestnáctkové

**e -** double, v exponenciální formě

f **–** double, ve formě s desetinou tečkou

**c -** int, zkonvertuje se na unsigned char reprezentující jeden znak

**s** **-** const char \*, nulou ukončený řetězec

**p -** void \*, adresa paměti, číslo v šestnáctkové soustavě

**Základní datové typy**

Char – celé číslo

Short – krátké celé číslo

Int – celé číslo

Long – dlouhé celé číslo

Long Long – ještě dělší

Float – racionální číslo

Double - s dvojitou přesností

Long double – velmi dlouh´raconální č.

PRG C

**Konstanty**

Konstanty jsou objekty, jejichž hodnota se za chodu programu nemění.

**Konstanty - řetězce**

**"Ahoj světe!"** - Ahoj světe

**"Znak ' lze zapsat"** - Znak ' lze zapsat

**"Znak \" nikoli"** - Znak " nikoli

**"Nový\nřádek"** – Nový

řádek

**"Hrozně" "Dlouhý"** **"Řetězec"** –

HrozněDlouhýŘetězec

**"Nula\000ukončuje"** - Nula

**Konstanty – znaky**

**'\\'** - Zpětné lomítko

**'\''** – Apostrof

**'\a'** – Zvonek

**'\b'** – Backspace

**´\f´** - Nová Stránka

**'\n'** – Odřádkování

**'\r'** - Na začátek řádky

**'\t'** – Tabelátor

**´%i´** - datový typ