UNIVERZITA PALACKÉHO V OLOMOUCI
KATEDRA INFORMATIKY

ZÁKLADY PROGRAMOVÁNÍ 1

KONZULTACE

- V pracovně 5.070
- Každé úterý 15:00 - 16:30
- Email: roman.vyjidacek@upol.cz
- Web: https://vyjidacek.cz

DOPORUČENÁ LITERATURA

- Pavel Herout: Učebnice Jazyka C. Kopp, 2007.
- Kernighan, Ritchie: Programovací jazyk C, 2006.
- Reek Kenneth: Pointers on C. Addison Wesley, 1997.
- Robert Sedgewick: Algorithms in C. Addison-Wesley Professional, 2001.
- Jeri R. Hanly, Elliot B. Koffman: Problem Solving and Program Design in C. Addison Wesley, 2006.
- Eric S. Roberts: Programming Abstractions in C. Addison Wesley, 1997.
- Eric S. Roberts: The Art and Science of C. Addison Wesley, 1994.

PODMÍNKY ZÍSKÁNÍ ZÁPOČTU

- Získání alespoň 25 bodů, kde
  - 30 bodů lze získat za vypracování 2 domácích úkolů (každý za 15 bodů).
  - Dalších 10 bodů je možné získat za vypracování bonusových úkolů.
  - Na vypracování domácích úloh budete mít 3 týdny.
  - Bonusové úlohy můžete odevzdat na semináři, na kterém jsou zadány, nebo do začátku příštího semináře.
- Povinná 75% účast.

V ČEM BUDEME PROGRAMOVAT?

JAZYK C

- Nízkoúrovňový programovací jazyk
- Platformově nezávislý
- Kompilovaný (překládaný)
- Minimalistický

V ČEM PSÁT PROGRAM?

- Code::Blocks (multiplatformní).
  [český manuál: http://www.sallyx.org/sally/c/codeblocks/]
- MS Visual Studio (MS Windows).
- GNU Emacs (GNU/Linux).
- Xcode (OS X).
- Sublime Text (multiplatformní).
- Libovolný textový editor.

PRVNÍ PROGRAM

```c
#include <stdio.h>

int main() {

    /* Tohle je komentar */
    printf("Hello, World!\n");
    return 0;
}
```

PŘEKLAD PROGRAMU

- Překlad probíhá příkazem: `gcc main.c -o main`
- V grafických prostředích kliknutím na příšlušné tlačítko (Build, Compile, případně Run, Debug).

SYNTAXE JAZYKA

SYNTAXE

- Rozlišuje velikost písmen (case-sensitive):
  - cislo ≠ Cislo ≠ CISLO
- Většinou ignoruje bílé znaky (odřádkování, tabulátor, mezera):
  - a=b+c je stejné jako a = b +c
  - “Ahoj svete!” ≠ “Ahoj svete!”
  - int cislo; ≠ intcislo;

PROGRAM JE POSLOUPNOST PŘÍKAZŮ

- Každý příkaz je ukončen středníkem.

- Příklady:

```
int cislo = 47;
int obvod = 2 * PI * r;
printf("Ahoj svete!\n");
printf("obvod=%i\n", obvod);
```

PROMĚNNÁ

- Pomocí proměnné si “pojmenujeme” hodnotu.
- Musíme definovat jakého typu proměnná je:
  - typ nazev_promenne = hodnota;
  - typ nazev_jine_promenne;

PROMĚNNÁ

- Název se může skládat z písmen (a-zA-Z), číslic (0-9) a podtržítka (_). Měl by začínat písmenem. Příklady:

```
int vyska = 180;
double obvod_kruznice = 6.283;
float tan2PI;
int a, b = 10, c;
```

- **Používejte smysluplné názvy proměnných!** Váš kód bude srozumitelnější jak pro Vás tak pro ostatní.
- Příklad `2 * PI * r` vs `2 * wtf * btw`

ZÁKLADNÍ DATOVÉ TYPY

DATOVÝ TYP	VELIKOST	ROZSAH
char	1B	0 – 255
int	4B	-2147483648 – 2147483647
float	4B	$1.2 * 10^{-38}$ – $3.4 * 10^{+38}$
double	8B	$2.3 * 10^{-308}$ – $1.7 * 10^{+308}$

KVANTIFIKÁTORY LONG A SHORT

DATOVÝ TYP	VELIKOST	ROZSAH
short int	2B	-32768 – 32767
long int	8B	$-(2^{63})$ – $(2^{63}) - 1$
long	8B	-32768 – 32767
short	2B	-2147483648 – 2147483647

KVANTIFIKÁTORY SIGNED A UNSIGNED

- Na celočíselné typy (int, char) lze aplikovat kvalifikátory signed a unsigned.

DATOVÝ TYP	VELIKOST	ROZSAH
signed char	2B	-128 – 127
unsigned char	2B	0 – 255

LOGICKÉ HODNOTY

- Logické hodnoty “pravda” a “nepravda” reprezentujeme čísly: •
- nula = nepravda.
- nenulové číslo = pravda.

VÝSTUP NA OBRAZOVKU

- K výpisu na obrazovku používáme funkci printf.

```
printf(ridici_retezec, hodnota1, hodnota2, ...);

int a = 10, b = 27;
printf(“Velikost typu int je %i", sizeof(int)); // Velikost typu int je 4
printf("Soucet 15 + 10 je %i", 15 + 10); // Soucet 15 + 10 je 25
printf("Soucet a + b: %i", a + b); // Soucet a + b: 37
printf("Soucet: %i, soucin: %i, rozdil: %i”, a + b, a * b, a - b);
printf("Dek: %d, okt: %o, hexa: %x", b, b, b); // Dek: 27, okt: 33, hexa: 1b
```

ZNAKY

- K uložení jednoho znaku používáme datový typ char.
- Znak je interně reprezentován číslem
- Podle ASCII tabulky se zjistí jaký znak se má vytisknout.

```
char znak_a = 'a';
printf("znak: %c, cislo: %i\n", znak_a, znak_a); // znak: a, cislo: 97
char tajny_znak = 98;
printf("znak: %c, cislo: %i\n", tajny_znak, tajny_znak); // znak: b, cislo: 98
```

VSTUP Z KLÁVESNICE

- Pro načítání hodnot z klávesnice používáme funkci scanf.

```
scanf(ridici_retezec, &promenna1, &promenna2, …);
```

```
int cislo, hexa_cislo, cislo1, cislo2, den, mesic, rok;
scanf("%i", &cislo);
scanf("%x", &hexa_cislo);
scanf("%d %d", &cislo1, &cislo2);
scanf("%d.%d.%d", &den, &mesic, &rok);
```

ZÁKLADNÍ MOŽNOSTI ŘÍDÍCÍHO ŘETĚZCE

- %c – výpis nebo načtení znaku.
- %d nebo %i – celé číslo desítkově znaménkově.
- %u – celé číslo desítkově neznaménkově.
- %o – celé číslo osmičkově.
- %x nebo %X – celé číslo šestnáctkově (malá/velká písmena).
- %f – desetinné číslo.
- %e nebo %E – desetinné číslo semilogaritmicky.
- %s – textový řetězec.
- %.3f – vypíše desetinné číslo s přesností na tři des. čísla.

ARITMETICKÉ OPERACE

Operátor	Argumenty	Popis
-	1	Mění znaménko argumentu
+	2	součet argumentů
-	2	rozdíl prvního a druhé argumentu
*	2	součin argumentů
/	2	podíl argumentů
%	2	zbytek po celočíselném dělení prvního argumentu druhým

TYPY VS ARITMETICKÉ OPERÁTORY

- Mějme následující kód:

```
int int_number = 11;
double double_number = 2.5;

int int_result = int_number / double_number;

double double_result = int_number / double_number;
```

- Jaké hodnoty budou obsahovat proměnné int_result a double_result?

TYPY VS ARITMETICKÉ OPERÁTORY

- Typ výsledku aritmetického operátoru je většinou určen typem jeho argumentů s nejvyšší prioritou.
- Typy s plovoucí řádovou čárkou mají vyšší prioritu než celočíselné.
- V rámci jedné skupiny typů mají větší typy vyšší prioritu.
- U operátoru přiřazení je výsledný typ roven typu proměnné do které hodnotu přiřazujeme.

ÚKOLY

- Úkoly nalezne na adrese https://vyjidacek.cz/zpc1/lecture1.html.
- Nemusíte je odevzdávat. Slouží pouze k procvičení.