Organizace, výuka, body, témata

- Garant: doc. RNDr Jitka Kreslíková, CSc.
  náplň přednášek, zkoušek.
- Zástupce garanta: Ing. Aleš Smrčka, PhD.
  náplň cvičení, seminářů, projektů.
- Vedení cvičení: cca 16 cvičících v cca 40 skupinách.
  Naše skupina: cvičící RNDr Libor Škarvada
  Konsultace s L. Š. ke cvičení: čtvrtky a pátky v C235 po dohodě mailem.
- Základní informace na IS VUT: `https://www.fit.vut.cz/study/course/IZP/`

Výuka a hodnocení

2 přednášky (2 h + 1 h týdně)

1 seminář (1 h týdně)

1 laboratorní cvičení (2 h týdně)	1 bod týdně, nutných aspoň 6 bodů
První projekt (do 30. 10.)	10 bodů, nutný aspoň 1 bod
Druhý projekt (do 1. 12.) s obhajobou ve 49. týdnu 2023	14 bodů, nutný aspoň 1 bod
Polosemestrální test 10. 11. 2023 v 18:00	12 bodů
Zápočet	nutné 23 body ze cvičení, projektů a testu
Závěrečná zkouška leden 2024	54 body, nutné aspoň 23 body
	Celkem je nutno získat od 50 do 100 bodů.

Cvičení

	téma	pozn.
21. 9.	první programy	
5. 10.	cykly, typy	
12. 10.	řetězce	
19. 10.	funkce	
26. 10.	struktury	
2. 11.	ukazatele	*odevzdání 1. projektu 30. 10. 8:00*
9. 11.	funkce	*polosemestrální test 10. 11. 18:00*
23. 11.	alokace paměti, dbg	
30. 11.	iterace a rekurse	*odevzdání 2. projektu 1. 12. 23:59*
7. 12.	*obhajoba*	*obhajoba 2. projektu*
14. 12.	datové struktury	

(Aktuální verzi viz `https://moodle.vut.cz/course/view.php?id=268296`)

Programovací jazyk C

- *univerzální* – lze jím zapsat každý algoritmus (tzv. turingovská úplnost)
- *imperativní* – program se skládá z *příkazů*, které mění *stav výpočtu*
- *typovaný* – pracuje s daty různých *typů*, typy se kontrolují *při překladu*
- relativně *nízké úrovně* – chudý typový systém, základní datové struktury
- doplněný *makro-jazykem* CPP se základními direktivami
- typicky je *kompilovaný* – programy se překládají do spustitelného kódu

Vývoj programu v C

```
do {
  do {
    vim program.c ;                                    // editace
    gcc -o program program.c ;                         // překlad
  } while (syntaktické chyby) ;
  ./program test.data ;                                // testování
  if (logické chyby) {
    while (nevidíme příčinu chyb) { gdb program ; }    // ladění
  }
} while (logické chyby) ;
```

- Co znamenají jednotlivé kroky vývojového cyklu programu,
- co dělají zmíněné nástroje,
- význam souborů `.c`, `.h`, `.o`, `.so`, `.a`.
- textové editory: vim, gedit, atom, nano, pico, emacs, ne, pspad ...
- překlad: `gcc -std=c11 -Wall -Wextra -Werror -o` *jméno* *jméno*`.c`

Vývojová prostředí

Zkušenější programátoři používají ke zrychlení práce.

Vývojové prostředí může být

- buď samostatný programový systém – Code::Blocks, Visual Studio, Eclipse,
- anebo rozšíření textového editoru – vim-mode, emacs-mode...

Funkce vývojového prostředí

1. Propojení kroků vývojového cyklu – spouštění překladu, ladění...,
2. označení míst se syntaktickými chybami,
3. barevné zvýraznění kódu,
4. automatické odsazování,
5. doplňování syntaktických prvků
6. napovídání i doplňování jmen,
7. vyhledávání definic hodnot a typů,
8. zobrazení závislostí.

Instalace a používání Code::Blocks:

- wiki stránky IZP `wis.fit.vutbr.cz/FIT/st/cwk.php?title=IZP:Lab1&csid=569324&id=10033#CodeBlocks`

Poznámka: Pokud někomu nevyhovuje Code::Blocks nebo jiný vývojový nástroj, má potíže s jeho správným nastavením nebo nevyužije jeho pokročilé funkce, je vždy možné vrátit se k příkazovému řádku. I tam lze jeden průchod vývojovým cyklem provést stiskem cca čtyř kláves (uložení souboru v okně editoru, přechod do vedlejšího okna se shellem, krok zpět v historii shellu (šipka nahoru) a Enter).

Informace o syntaxi, sémantice a používání jazyka C, o programových knihovnách:

- knihy, např. P. Herout, Učebnice jazyka C, Kopp 2009
- Unix: `man 3` *příkaz*
- `https://cplusplus.com/`
- `https://www.learnc.net/c-tutorial/`
- `https://stackoverflow.com/questions/tagged/c`

Jednoduché programy

Cvičení: Napište program, který na standardní výstup napíše krátký text.

```
#include <stdio.h>

int main(void) {
    printf("Hello, Kitty!\n");
    return 0;
}
```

Cvičení: Program, který přečte ze vstupu celé číslo a napíše, jakou má paritu (sudé/liché).

```
#include <stdio.h>
int main (void) {
    int num;

    // Read a user-supplied number
    printf ("integer number: ");
    scanf ("%d", &num);

    if ( num % 2 == 0 )
        printf ("Number %d is even.\n", num);
      else
        printf ("Number %d is odd.\n", num);

    return 0;
}
```

Cvičení: Program, který ze vstupu ze tří čísel a, b, c vypočte a vypíše hodnotu $D = b^2 - 4ac$. (Je-li $a \neq 0$, pak se D nazývá *diskriminant* kvadratické rovnice $ax^2 + bx + c = 0$.)

```
#include <stdio.h>

int main() {

    int a, b, c, D;           // declare (mutable) variables

    a = -7;  b = 2;  c = 3;   // assign values to the variables

    D = b * b - 4 * a * c;    // evaluate an expression

    printf("a = %d, b = %d, c = %d, D = %d\n", a, b, c, D);
                              // write all values to stdout

    return 0;
}
```

Cvičení: Program, který

- načte tři čísla a, b, c,
- ověří, zda $a < b$; pokud ne, skončí s chybou „$[a, b\,]$ není interval“,
- načte číslo x,
- vypíše, zda $x \in [a, b\,]$.

```
#include <stdio.h>

int main(void) {
    int a, b;  // interval bounds (lower, upper)
    int num;   // inspected number

    printf ("endpoints: ");  scanf("%d %d", &a, &b);
    if (a >= b) { printf("Not an interval.\n"); return 1; }
    printf ("integer number: ");  scanf("%d", &num);

    printf ("Number %d lies ", num};
    if ( num >= a && num <= b )
            { printf ("within"); }
      else  { printf ("outside"); }
    printf (" the interval [%d, %d].\n", a, b);
    return 0;
}
```

Poznámka: Chybové ukončení jsme řešili výpisem hlášky

```
printf("Not an interval.\n");
```

kterou funkce `printf` zapisuje do znakového proudu zvaného *standardní výstup*, `stdout`.

V praxi se chybové výstupy píší do odděleného znakového proudu, který se nazývá *standardní chybový výstup*, `stderr`.

```
fprintf(stderr, "Not an interval.\n");
```

Cvičení: Program, který načte tři čísla a vypíše největší z nich.

```
#include <stdio.h>

int main(void) {
  int a, b, c, m;

  printf ("Three integers: ");  scanf ("%d %d %d", &a, &b, &c);

  if ( a > b )
          { m = a; }
    else  { m = b; }
  if ( c > m )
     { m = c; }

  printf ("The maximum of the three numbers: %d.\n", m);

  return 0;
}
```

Lze některé složené závorky vynechat? Které? Proč?

Cvičení: Program počítající aritmetický průměr tří čísel.

```
#include <stdio.h>

int main(void) {
  float a, b, c, mean;

  printf ("Three decimal numbers: ");
  scanf ("%f %f %f", &a, &b, &c);

  mean = (a+b+c) / 3.0;

  printf ("Arithmetic mean: %6.2f.\n", mean);

  return 0;
}
```

Zjistěte, co dělá funkce `pow` z knihovny `math`.

Změňte program tak, aby počítal *geometrický* průměr.

Cvičení: Program, který načte letopočet a vypíše, zda je daný rok přestupný. Uvažujte pouze gregoriánský kalendář, tj. letopočty po roce 1582.

```
#include <stdio.h>
int main(void) {
  int year;
  printf ("Year: ");  scanf ("%d", &year);

  if (year < 1582) {
    printf("Year %d predates Greg. calendar.\n", year);
    return 1;
  }

  if ((year % 4 == 0 && year % 100 != 0) || year % 400)
          { l = "is"; }
    else  { l = "is not"; }

  printf ("Year %d %s a leap year.\n", year, l);

  return 0;
}
```