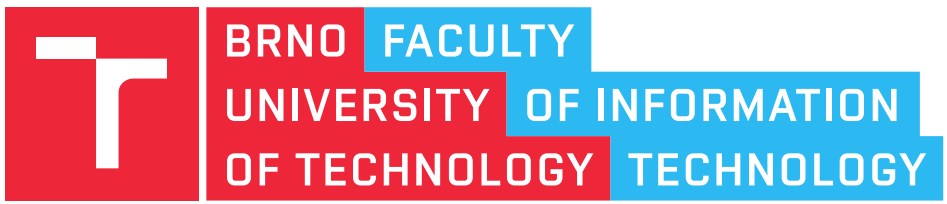
Základy programování (IZP)

2. počítačové cvičenie

Brno University of Technology, Faculty of Information Technology
Božetěchova 1/2. 612 66 Brno - Královo Pole
ilazur@fit.vut.cz



Celé čísla

```
int i = 42;
const int i = 42;
```

Desatinné čísla

- Preferovaným typom je **double** pred **float**
- Pri načítavaní aj výpise využívame %lf

```
double e = 4.2;
const double e = 2.7182818284;
printf("%.2lf ", e);
```

Znaky

- Pri načítavaní aj výpise využívame %c

```
char c = 'a';
const char z = 'A';
```

Výpočet koreňov kvadratickej rovnice

1. Načítajte zo STDIN 3 celé koeficienty kvadratickej rovnice a, b, c
2. Skontrolujte, či zadané čísla zodpovedajú koeficientom kvadratickej rovnice
3. Spočítajte hodnotu diskriminantu
4. Na základe hodnoty diskriminantu určite riešenie rovnice

```
    #include <math.h>
    sqrt(x); // 2th root
```

```
gcc -std=c99 -Wall -Werror -g -o quad quad.c -lm
```

```
    x = x * 2.0; // type cast
    x = (double)(x * 2); // type cast
```

Výpočet koreňov kvadratickej rovnice

```
#include <stdio.h>
#include <math.h>

int main(void) {
   int a, b, c, discriminant;
   double x1, x2;

   printf("Select three integer numbers: ");
   scanf("%d %d %d", &a, &b, &c);

   if (a == 0) {
      printf("Error: The supplied coefficient 'a' = 0!\n");
      return 1;
   }
```

Výpočet koreňov kvadratickej rovnice

```
    discriminant = b * b - 4 * a * c;
    if (discriminant < 0) {
        printf("The quadratic equation has no roots.\n");
    } else if (discriminant == 0) {
        x1 = -b / (2.0 * a); // Explicit type cast: (double)(2 * a)
        printf("The quadratic equation has only one root: x1=%lf\n", x1
);
    } else {
        x1 = (-b + sqrt(discriminant)) / (2 * a);
        x2 = (-b - sqrt(discriminant)) / (2 * a);
        printf("The quadratic equation has the following roots:
                    x1=%lf, x2=%lf\n", x1, x2);
    }

    return 0;
}
```

Detekcia písmena

1. Načítajte zo STDIN vstup
2. Rozhodnite, či je vstupom písmeno

```
char c = 'a';
if (c == 'a') {
    printf("c is letter a");
}
```

```
char c = 'a';
if (c == 97) {
    printf("c is letter a");
}
```

Detekcia písmena

```
#include <stdio.h>

int main(void) {
  char c;

  printf("Select a character: ");
  scanf("%c", &c);

  if ((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z')) {
    printf("Char '%c' is a letter!\n", c);
  } else {
    printf("Char '%c' is not a letter!\n", c);
  }

  // Show that characters can be compared to integers
  if ('a' == 97) {
    printf("Character '%c' has ordinal value of '%d'!\n", 97, 'a');
  }

  return 0;
}
```

Cyklus

- Blok programu, ktorý je vykonaný viacero ráz

Druhy cyklov

- S pevným počtom opakovaní
- S podmienkou na začiatku
- S podmienkou na konci

```
for (int i = 0; i < 10; i++) {
    printf("%d\n", i);
}
```

Detekcia párnych čísiel

1. Načítajte zo STDIN 3 celé čísla
2. Rozhodnite, pre každé číslo, či je párne, alebo nepárne

Detekcia párnych čísiel

```c
#include <stdio.h>

int main(void) {
  for (int i = 0; i < 3; i++) {
    int num;
    printf("Select the %d. number: ", i + 1);
    scanf("%d", &num);

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

  return 0;
}
```

Výpočet faktoriálu z daného čísla

1. Načítajte zo STDIN 1 celé číslo
2. Vypočítajte faktoriál z tohoto čísla

```
    i = i + 1;
    i += 1;
```

Výpočet faktoriálu z daného čísla

```c
#include <stdio.h>

int main(void) {
    int num;

    printf("Select an integer number: ");
    scanf("%d", &num);

    if (num < 0) {
        printf("Error: Factorial is defined only for non-negative numbers!\n");
        return 1;
    }

    int factorial = 1;
    for (int i = 0; i < num; i++) {
        factorial *= i + 1; // is similar to factorial = factorial * (i + 1);
    }

    printf("Factorial of %d = %d\n", num, factorial);

    return 0;
}
```

Pole

- štruktúra, ktorá združuje konečný počet prvkov
- Indexácia začína číslom **0**

```
int arr[5];
int arr[] = {0, 1, 2, 3, 4};
arr[0] = 33;
```

Ukladanie čísiel

1. Načítajte zo STDIN vopred daný počet desatinných čísiel do poľa
2. Následne zo STDIN získajte 1 celé číslo, ktoré bude udávať index
3. Ak je index korektný, vypíšte číslo na danom indexe

Ukladanie čísiel

```
#include <stdio.h>

int main(void) {
  const int arr_size = 5;
  double numbers(arr_size);
  printf("Select %d numbers: ", arr_size);
  for (int i = 0; i < arr_size; i++) {
    scanf("%lf", &numbers(i));
  }

  int index = 0;
  printf("Select index: ");
  scanf("%d", &index);

  if (index < 0 || index > (arr_size - 1)) {
    printf("Wrong index selected\n");
    return 1;
  }

  printf("Selected value on index %d is %lf\n", index, numbers(index));
  return 0;
}
```

Stačí vypracovať jednu variantu

- Načítajte 5 celých/desatinných čísiel a vypíšte ich v opačnom poradí
- Načítajte 5 celých/desatinných čísiel a vypíšte ich maximum a minimum
- Načítajte 5 celých/desatinných čísiel a vypíšte ich sumu a aritmetický priemer