1. Vstup a výstup

Příkazy vstupu a výstupu. Parsování. Ošetření výjimky.

* **vstup**:

Console.ReadLine();

* **výstup**

Console.WriteLine()

**Console** je statická třída, která obsahuje metody k obsluze konzole. Metodu na třídě voláme pomocí operátoru tečka (.). Každá metoda může obsahovat nějaké vstupní parametry, které se zadávají do závorky a jsou oddělené čárkou. V případě metody WriteLine() je parametrem text k vypsání.

**Parsováním** se myslí převod z textové podoby na nějaký specifický typ, např. číslo. Mnoho datových typů má v C# již připraveny metody k parsování.

string vek\_retezec = Console.ReadLine();

int vek = int.Parse(vek\_retezec);

opak k parsování - převést cokoli do textové podoby

soucet.ToString()

**Ošetření vstupu**

metoda TryParse(). Metoda vrací true/false podle toho, jestli se parsování podařilo či nikoli.

Console.WriteLine("Zadejte první číslo:");

while (!float.TryParse(Console.ReadLine(), out a))

Console.WriteLine("Neplatné číslo, zadejte prosím znovu:");

**Výjimky, tzv. pasivní ošetření chyb.**

Zejména, když je operace složitější a bylo by příliš náročné ošetřovat všechny možné chybové stavy, nastupují výjimky,

C# .NET obsahuje vestavěné třídy výjimek pro každou možnou chybu. Třída Exception je základní třídou všech tříd výjimek.

Pokusíme se nebezpečnou část kódu spustit v "chráněném režimu". Tento režim je nepatrně pomalejší a liší se tím, že pokud dojde k chybě, máme možnost ji odchytit a zabránit pádu programu. O chybě zde hovoříme jako o výjimce. Využíváme k tomu tzv. try-catch bloky.

**try block:** Jakýkoli podezřelý kód, který může vyvolávat výjimky, by měl být vložen do try{ }bloku.

Pokud během provádění dojde k výjimce, tok řízení skočí na první odpovídající catchblok.

**catch block:** Blok catch je blok obsluhy výjimek. Blok catch přebírá parametr typu výjimky, pomocí kterého můžeme získat podrobnosti o výjimce.

**finally block:** Blok finally bude vždy proveden bez ohledu na to, zda došlo k výjimce nebo ne. Obvykle finally by měl být blok použit k uvolnění zdrojů, např. k uzavření jakéhokoli streamu nebo objektů souboru, které byly otevřeny v try bloku.

Rozpoznání typu výjimky

Do catch bloku vložíme jako parametr proměnnou e typu Exception. Třída Exception reprezentuje výjimku a všechny výjimky jsou z ní odděděné. Obsahuje vlastnost Message, kde je uložena vlastní chybová hláška.

Výjimku lze vyvolat ručně pomocí klíčového slova throw.

# běhové chyby:

* dělení nulou:

int/0 pak chyba dělení nulou

float / 0 výsledek nekonečno

* nesprávný formát vstupu
* práce se soubory
* index mimo rozsah
* přetečení RAM

Console.WriteLine("Zadejte číslo, které bude dělit číslo 1");

//dělení nulou => běhová chyba jen pro celočíselný typ

//dělení nulou pro float double není chyba výsledek

try

{

int num = int.Parse(Console.ReadLine());

//if (num==0) throw new DivideByZeroException("dělení nulou ");

int result = 1 / num;

Console.WriteLine("1 / {0} = {1}", num, result);

}

catch (Exception ex)

{

Console.WriteLine("Nastala chyba! "+ex.Message);

}

Console.ReadKey();

if (double.IsInfinity(d)) // test preteceni v pohyblive radove carce //

Console.WriteLine("NEKONECNO! ");

celociselne preteceni (exception) je v C# standardne vypnute