

C# pro začátečnice

Martin Černil, Jiří Hudec

Filip Eckstein, Jan Kratochvíla, Filip Pindej

Jiří Michalčík, Josef Trbušek, Martina Nemethová

21.1. 2025

Použití materiálů

Toto dílo je licencováno pod

Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License



LEKCE

Co nás čeká a jak to bude probíhat 😊



OSNOVA

- 21. 1. Organizace, úvod, Visual Studio, Hello World, konzole
- 28. 1. Proměnné, datové typy, podmíněný příkaz if
- 4. 2. Debugging, switch a parsování
- 11. 2. **Opakování, cykly**
- 18. 2. Metody
- 25. 2. Objektově orientované programování (OOP)
- 4. 3. OOP pokračování
- 11. 3. **Opakování OOP**
- 18. 3. Pole
- 25. 3. **Opakování**
- 1. 4. Grafická kalkulačka - základy
- 8. 4. Grafická kalkulačka - pokračování, konzultace, rady a tipy, co dál

STRUKTURA LEKCE

Časově

- Začínáme v 18:00, končíme ve 20:30
- Plánujeme vždy dvě přestávky – zhruba v 18:50 a 19:50

Obsahově

- Rekapitulace úkolů z minulé lekce
- Výklad nové látky
- Breakout rooms - samostatná práce na úkolech s kouči

ÚKOLY

Ale budou samé jednoduché, můžete nám
věřit 😊



ÚKOLY

- Úkoly jsou **povinné/nepovinné**, nepovinné jsou silně doporučené.
- Odevzdávají se do portálu moje.czechitas.cz formou odkazu na dotnetfiddle.net, dostaneš od nás **individuální zpětnou vazbu**.
- Snažíme se opravit opravdu vše a odpovědět na všechny dotazy. Využij toho.
- Pro získání včasné odpovědi **odevzdávej úkoly během víkendu**.

BUDE TO BOLET?

- To, že jsi na úkolu strávila xx hodin, vůbec nevypovídá o tom, jestli na to máš nebo ne, jestli jsi chytrá, nebo ne, atd. Je to běžná součást učení, znamená to, že pracuješ opravdu intenzivně na tom, aby ses posunula dál a jde ti to!
- Neboj se tomu věnovat čas, je to úplně nová věc a na její vstřebání je potřeba zažít i “nepříjemné chvíle zoufalství”.
- Nezapomeň ale na pravidelný odpočinek.
- Odměnou ti bude funkční program, pochopení, jak ho vytvořit a hlavně skvělý pocit!

DOTAZY

The background is a solid pink color. In the top-left corner, there is a rectangular area with a light pink dot grid pattern. In the bottom-left corner, there is a circular area with a light pink dot grid pattern. In the bottom-right corner, there is a circular area with a light pink dot grid pattern. In the top-right corner, there is a circular area with a light pink dot grid pattern. In the center-right area, there is a large, light pink circular shape. Inside this shape, there is a line-art illustration of a woman's head and shoulders, facing right. The woman has short, wavy hair and is wearing a top with a large, pointed collar. The overall design is modern and minimalist.

ŽÁDNÝ DOTAZ NENÍ HLOUPÝ

- Vždy se najde v místnosti někdo, kdo se na to chce také zeptat, ale nemá odvahu.
- Nezapomeň, že jsou tu i **koučové**, kteří sem přišli, aby ti byli k dispozici.
- Vyplňuj prosím zpětnou vazbu.

HOSPODA?

Aneb zpětnou vazbu si rádi vyslechneme i ~~u piva~~ ústně.



ÚVOD DO PROGRAMOVÁNÍ

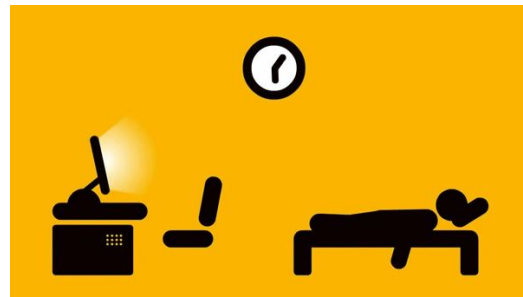


ÚVOD DO PROGRAMOVÁNÍ

- Mýty o programování
- Proč se učit programovat?
- IT pozice
- O čem je programování
- Programovací jazyk
- Jak to funguje
- Programovací hra
- Nastavení prostředí

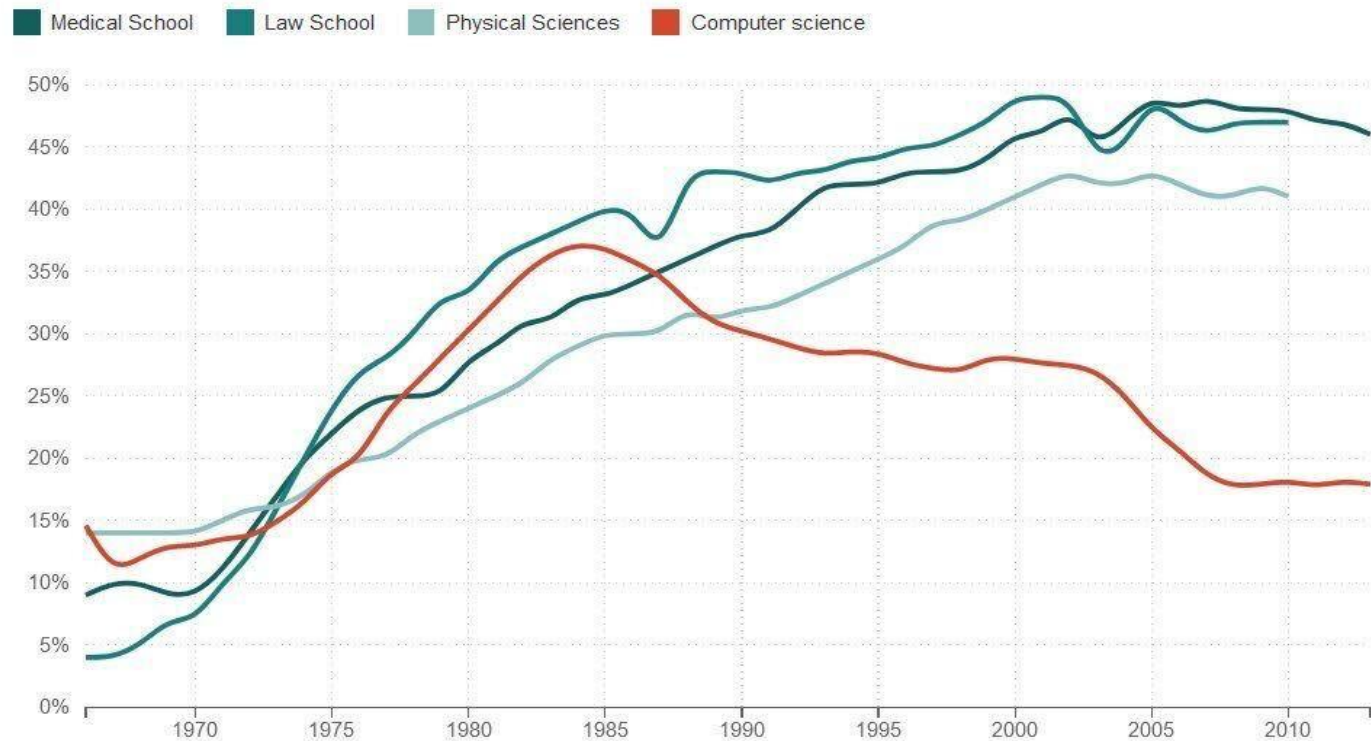
MÝTY O PROGRAMOVÁNÍ

- Ženy se na programování nehodí
- S programováním se musí začít v mládí
- K programování je potřeba mít vysokou školu
- K programování je potřeba matematika
- Programátoři jsou asociálové zavření někde ve sklepě
- IT je jen o programování



What Happened To Women In Computer Science?

% Of Women Majors, By Field



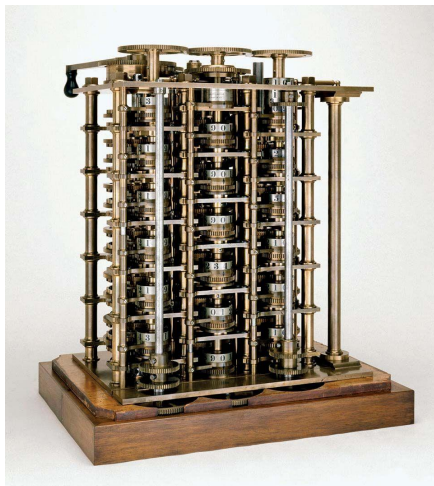
Source: National Science Foundation, American Bar Association, American Association of Medical Colleges

Credit: Quoc Trung Bui/NPR

MÝTY – ŽENY SE NA PROGRAMOVÁNÍ NEHODÍ

Ada Lovelace

- 1815 - 1852 Londýn
- Babbage, analytical engine
- Zavedla pojmy např.
 - podmíněný skok
 - cyklus
 - podprogram



MÝTY – ŽENY SE NA PROGRAMOVÁNÍ NEHODÍ

Barbara Liskov

- 1939 -
- Teoretické práce vedoucí k definici **objektového programování**
- https://en.wikipedia.org/wiki/Barbara_Liskov
- <https://www.quantamagazine.org/barbara-liskov-is-the-architect-of-modern-algorithms-20191120/>



MÝTY – ŽENY SE NA PROGRAMOVÁNÍ NEHODÍ

Margaret Hamilton

- 17.8.1936, USA
- vedoucí vývoje navigačního software pro let a přistání na Měsíci
- ocenění NASA (2003), Prezidentská medaile svobody (2016)

„V počátcích nebyli programátoři bráni úplně vážně, nebyla to svébytná disciplína, byla to spíš taková vedlejší odnož, hlavní roli hrál hardware. Programování se také považovalo spíše za umění a kouzlení, nikoli za vědu.“

- Originál zdrojového kódu Apollo 11 je k nahlédnutí na [GitHubu](#)



CHCI DO IT, MUSÍM UMĚT PROGRAMOVAT?

- Ne nutně, ale ...
- Sebevědomí
- Lepší pocit v týmu s programátory
- **Znalosti se opravdu hodí!**
- Znat základy programování patří k “základní gramotnosti” ve světě IT, která tě posune z pozice “běžného uživatele”

IT POZICE



IT POZICE

Computing-Core Disciplines	Computing-Intensive Fields	Computing-Infrastructure Occupations
Artificial intelligence	Aerospace engineering	Blockchain administrator
Cloud computing	Autonomous systems	Computer technician
Computer science	Bioinformatics	Data analyst
Computer engineering	Cognitive science	Data engineer
Computational science	Cryptography	Database administrator
Database engineering	Computational science	Help desk technician
Computer graphics	Data science	Identity theft recovery agent
Cyber security	Digital library science	Network technician
Human-computer interaction	E-commerce	Professional IT trainer
Network engineering	Genetic engineering	Reputation manager
Programming languages	Information science	Security specialist
Programming methods	Information systems	System administrator
Operating systems	Public Policy and Privacy	Web identity designer
Performance engineering	Instructional design	Web programmer
Robotics	Knowledge engineering	Web services designer
Scientific computing	Management information systems	
Software architecture	Network science	
Software engineering	Multimedia design	
	Telecommunications	

ZPĚT K PROGRAMOVÁNÍ 😊



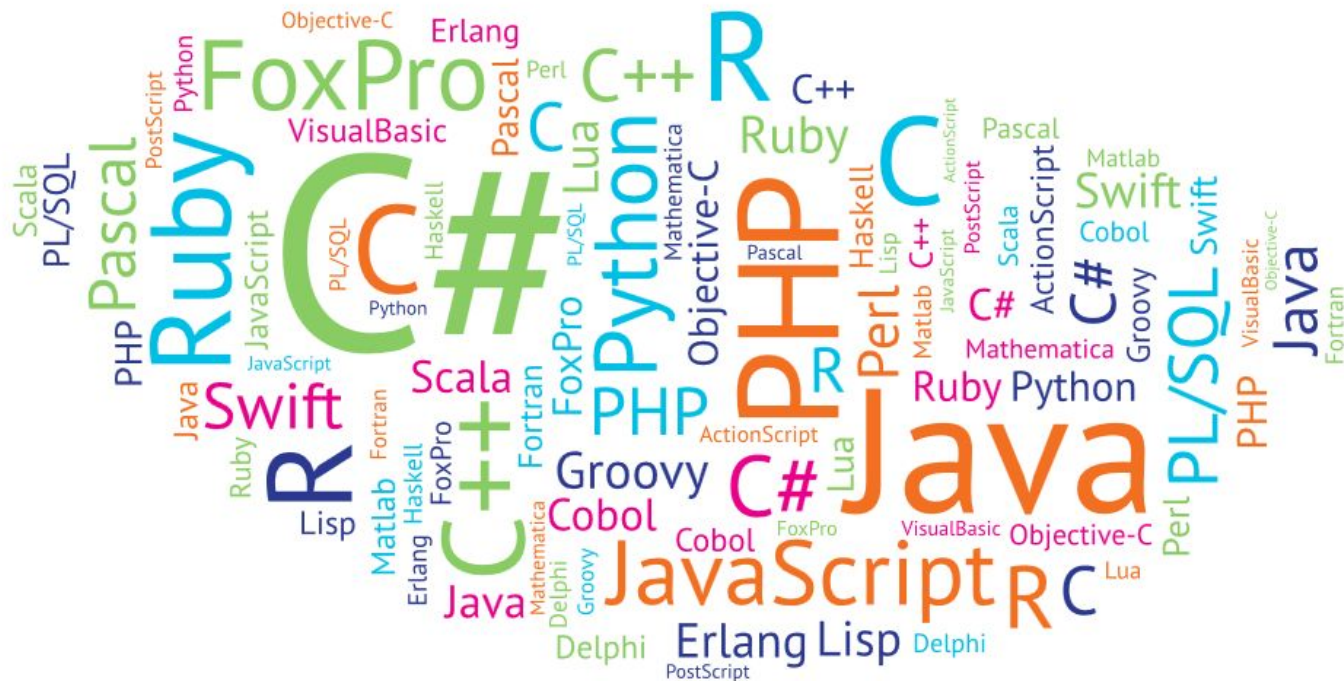
O ČEM JE PROGRAMOVÁNÍ?

- Způsob, jak říci stroji, co má dělat.
- Sada instrukcí jdoucích logicky po sobě tak, aby mohl stroj vyřešit zadaný úkol.

HRA NA PROGRAMOVÁNÍ



PROGRAMOVACÍ JAZYKY



PROGRAMOVACÍ JAZYK

- Formalizovaný jazyk pro komunikaci s počítačem
- Obrovské množství (a další vznikají)
- Různé specializace: desktopové aplikace, weby, databáze, mobilní aplikace
- Pro často používané funkcionality vznikají **knihovny** a **frameworky**.

PROGRAMOVACÍ JAZYK

- Programovací jazyk je mnohem jednodušší než přirozený jazyk → menší slovní zásoba v řádu desítek slov
- Důležité je porozumět principům, ty se nemění, programovací jazyky ano

PROČ C#

- Jazyk vyšší úrovně - snadněji se učí
 - Objektově orientovaný
 - Intuitivní vývojové prostředí Visual Studio
 - Žádaný na trhu práce
 - Pravidla jazyka jsou méně benevolentní než např. u PHP pro tvorbu webu, začátečník lépe získá správné návyky psaní kódu
-
- BONUS: syntaxe je v základu téměř identická s Javou

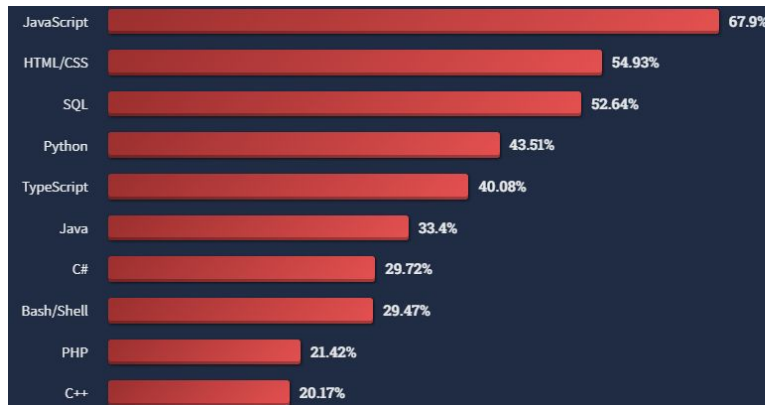
PROČ C#

Popularity of Programming Language

Worldwide, Sept 2021 compared to a year ago:

Rank	Change	Language	Share	Trend
1		Python	29.48 %	-2.4 %
2		Java	17.18 %	+0.7 %
3		JavaScript	9.14 %	+0.8 %
4		C#	6.94 %	+0.6 %
5		PHP	6.49 %	+0.4 %
6		C/C++	6.49 %	+0.9 %
7		R	3.59 %	-0.5 %
8	↑↑↑	TypeScript	2.18 %	+0.3 %
9		Swift	2.1 %	-0.4 %
10	↓↓	Objective-C	2.06 %	-0.6 %

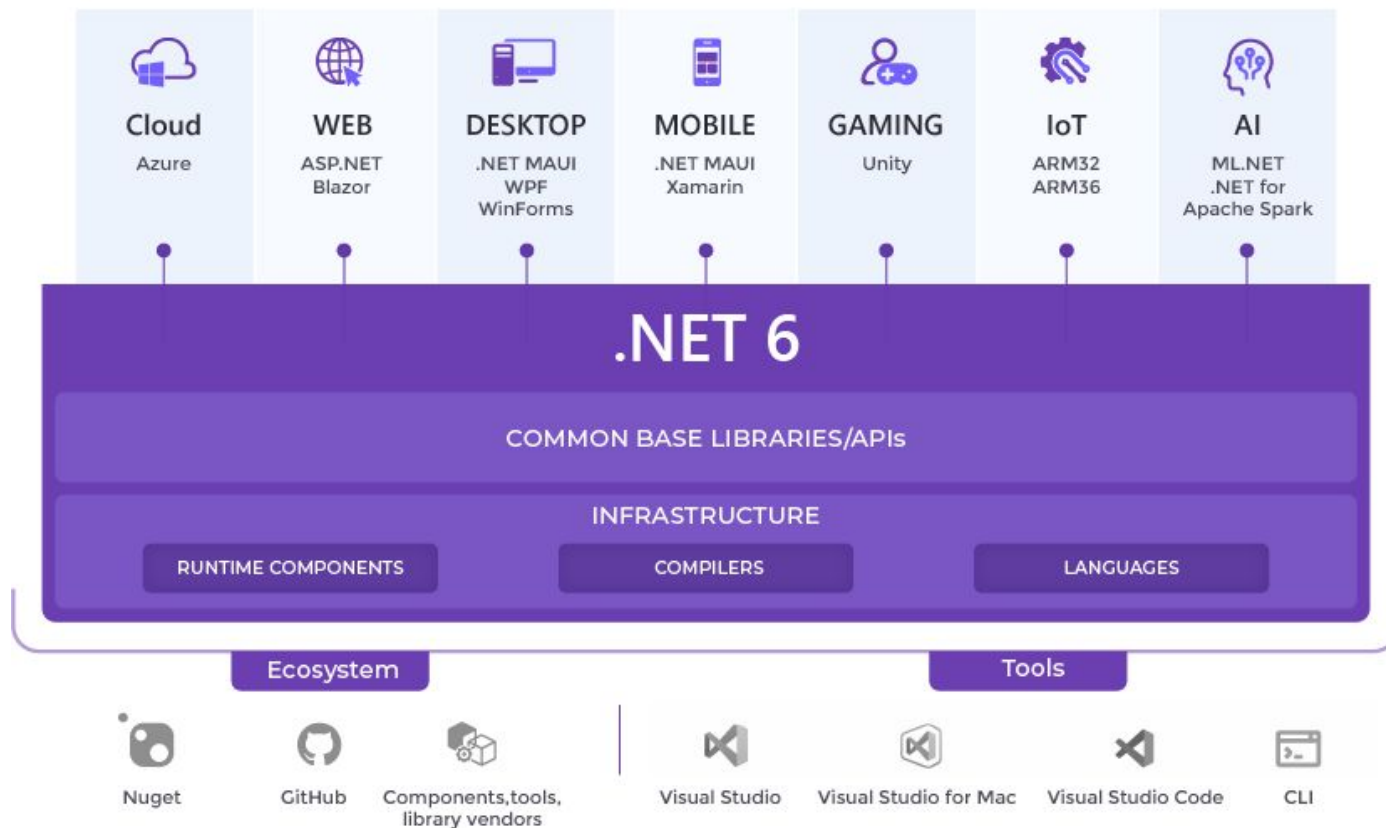
StackOverflow 2022 Most Popular Technologies



Top 10: Most In-Demand Programming Languages 2021

1. **JavaScript** (62%)
2. **Java** (59%)
3. **Python** (48%)
4. **C#** (40%)
5. **PHP** (32%)
6. **C++** (27%)
7. **Typescript** (24%)
8. **C** (15%)
9. **Kotlin** (15%)
10. **Swift** (14%)

Na co vše se dá .NET použít



JAK TO FUNGUJE



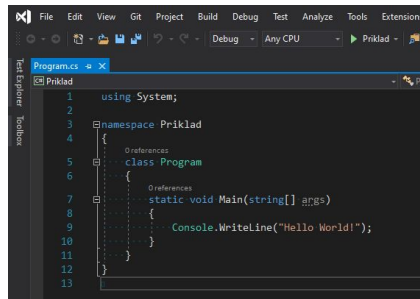
JAK TO FUNGUJE

- **CPU** = processor
 - Provádí výpočty
 - Zpracovává strojový kód
- **Paměť RAM** = krátkodobé úložiště dat
 - Drží v paměti data a procesy dokud je PC zapnuté
 - Rychlý přístup, malá kapacita
- **Harddisk** = dlouhodobé úložiště dat
 - Data zůstanou zachována i po vypnutí PC
 - Pomalejší přístup, velká kapacita

JAK TO FUNGUJE

Překladač

- Přeloží programátorem napsaný kód tak, aby mu „rozuměl“ procesor



01001000
01100101
01101100
01101100
01101111



Kompilátor

2104
1105
3106
7001
0053
FFFE
0000

ORG 100
LDA A
ADD B
STA C
HLT
DEC 83
DEC -2
DEC 0
END

```
static void Main(string[] args)
{
    int a = 83;
    int b = -2;
    int c = a + b;
}
```

BLOKOVÉ PROGRAMOVÁNÍ

<https://studio.code.org/hoc/1>



NASTAVENÍ PROSTŘEDÍ

- Visual Studio Code



- <https://code.visualstudio.com/download>

- .NET Fiddle



- dotnetfiddle.net

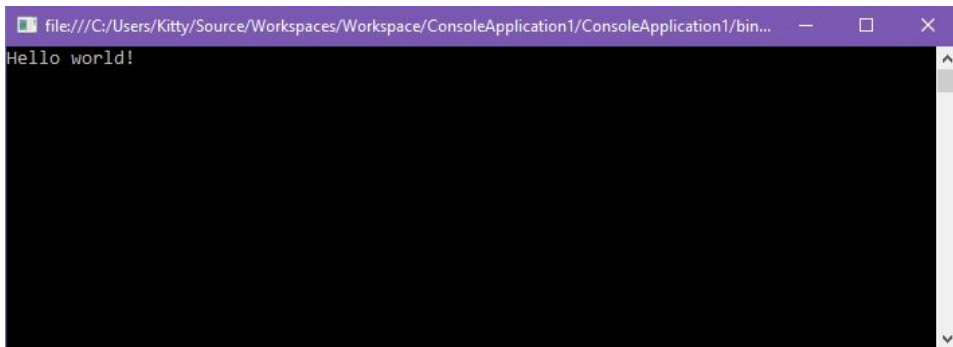
PRVNÍ PROGRAM

Hello World 😊



KONZOLE

- Slouží pro zadávání vstupních hodnot a zobrazování výsledků (výpis hodnot) u konzolových aplikací
- Konzole je aktivní po celou dobu běhu programu



KOMENTÁŘE V KÓDU

- Super na psaní poznámek přímo do kódu
- Jednořádkový komentář

```
// tohle je jednořádkový komentář
```

- Víceřádkový komentář

```
/* tohle je víceřádkový komentář  
aniž by bylo potřeba na každém řádku  
psát lomítka */
```

LEKCE 2

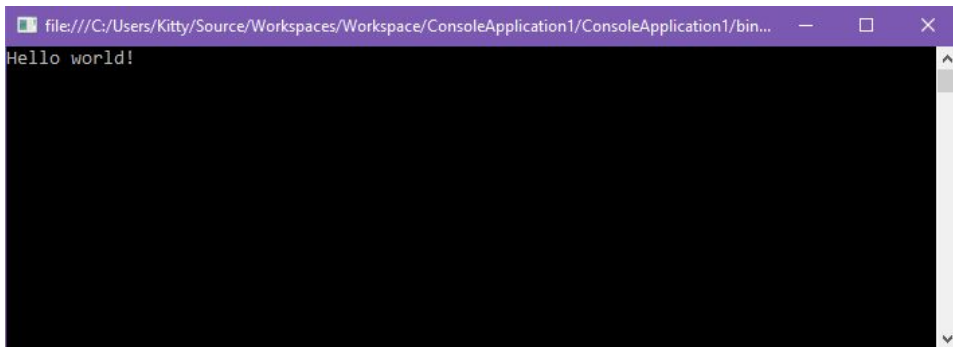


OPAKOVÁNÍ



KONZOLE

- Slouží pro zadávání vstupních hodnot a zobrazování výsledků (výpis hodnot) u konzolových aplikací
- Konzole je aktivní po celou dobu běhu programu



KOMENTÁŘE V KÓDU

- Super na psaní poznámek přímo do kódu
- Jednořádkový komentář

```
// tohle je jednořádkový komentář
```

- Víceřádkový komentář

```
/* tohle je víceřádkový komentář  
aniž by bylo potřeba na každém řádku  
psát lomítka */
```

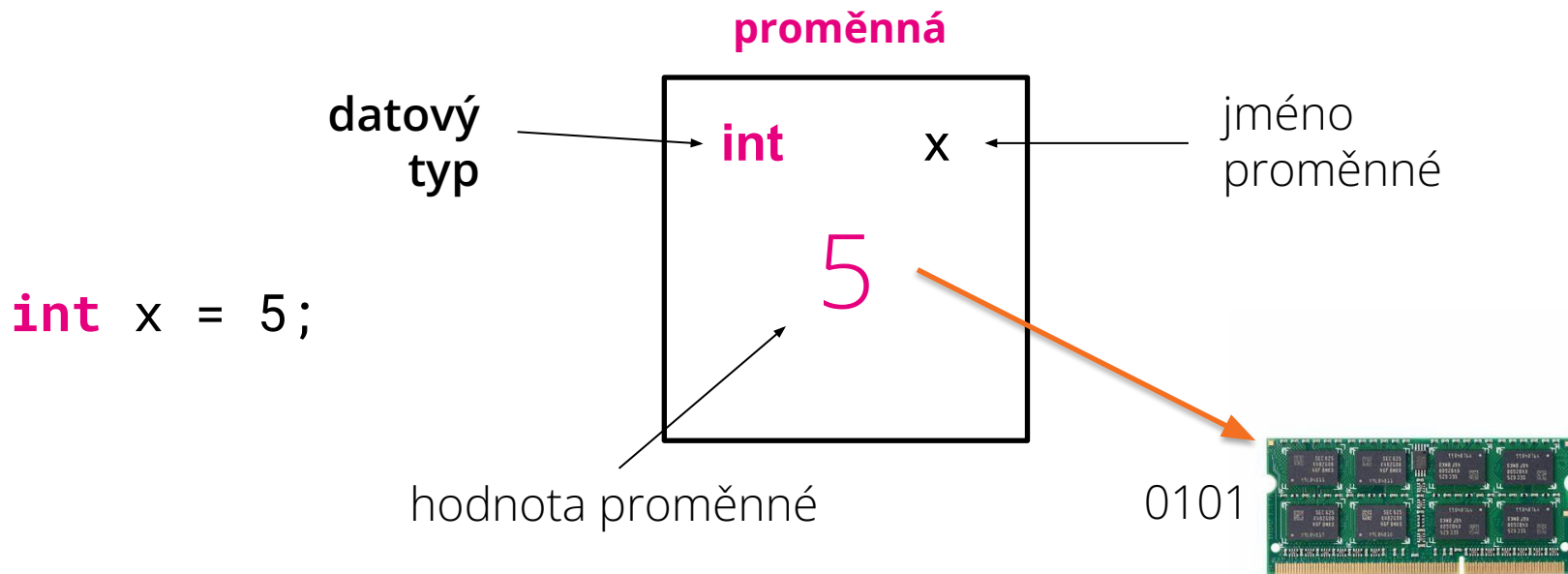
PROMĚNNÉ A DATOVÉ TYPY



PROČ PROMĚNNÁ?

- Způsob, jak krátkodobě uchovat data a jak s nimi pracovat v programu
- Hodnota se zadává/definuje na jednom místě, není nutné přepisovat na více místech v kódu -> eliminuje chyby, urychluje práce, kód má univerzální použití

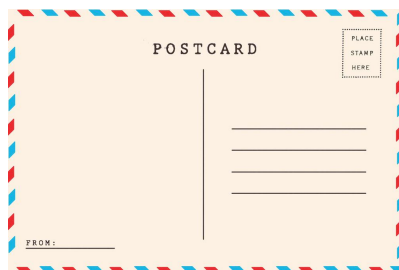
PROMĚNNÁ A DATOVÝ TYP



PROČ DATOVÉ TYPY?

- Důležité pro zápis v paměti – kvůli kapacitě i rychlosti
- Každý datový typ umožňuje jiné operace -> nutno dodržovat při programování – patří k základním principům
- C# je **silně typovaný jazyk** (nelze míchat “jablka” s “hruškami”), typy musí “sedět”

PŘIROVNÁNÍ – POHLEDY, OBÁLKY, BALÍKY



JAK TO VYPADÁ V KÓDU

Deklarace (vytvoření)
proměnné

```
int cislo;  
string text;
```

Použije se, pokud chceme proměnnou
vytvořit bez hodnoty

Deklarace (vytvoření)
proměnné **s přiřazením** hodnoty

```
int cislo = 10;  
string text = "Ahoj!";
```

Přiřazení hodnoty (přepsání)

```
cislo = 25;  
text = "Jak je?";
```


PŘEHLED DATOVÝCH TYPŮ

- **bool** = logická hodnota (true/false)
- **int** = celé číslo (-10, 0, 12, ...)
- **double** = desetinné číslo (-10.01, 0.0, 12.5, ...)
- **string** = řetězec znaků ("Czechitas", "x", "123" , "😊" , ...)
- **char** = znak
('C', 'z', 'e', 'c', 'h', 'i', 't', 'a', 's', 'x', '1', '2', '3')



POJMENOVÁNÍ PROMĚNNÝCH

- Lepší víceslovný název, než nicneříkající a, b, c, ...
- Jak na to:
 - Název proměnné začíná v C# vždy malým písmenem, je bez mezer a nesmí začínat číslem
 - Pozor – **nepoužívat diakritiku!**
 - Pokud má název více slov, má každé slovo kromě prvního velké počáteční písmeno:

nazevPromenne1
totoJeDalsiPromenna



VÝPIS HODNOTY

- Tato funkce je pro usnadnění již naprogramována
- Vypsát hodnotu proto můžeme pomocí jediného příkazu:

```
Console.WriteLine("Hello World!");  
Console.WriteLine(123);
```

- Výpis proměnné do konzole:

```
int cislo = 789;  
Console.WriteLine(cislo);
```

DATOVÝ TYP int

- **int** = celé číslo
- hodnoty: -10, 0, 12, ...
- základní (matematické) operace:
 - sčítání $1+2$
 - odčítání $3-1$
 - násobení $2*2$
 - dělení $4/2$ (celočíselné dělení)
v případě, že vyjde desetinné číslo, desetinná místa se useknou bez zaokrouhlení! $4/3 = 1$

DATOVÝ TYP **double**

- **double** = desetinné číslo
- hodnoty: -10.01, 0.0, 12.5, ...
 - POZOR na desetinnou tečku a čárku, záleží na nastaveném jazyku
 - Proč 0.0? Samotná 0 je automaticky považována za celé číslo
- základní (matematické) operace jsou stejné jako u **intu**

DATOVÝ TYP string

- **string** = řetězec znaků, text
- hodnoty: "Czechitas", "x", "123" , "😊" , ...
- základní operace - skládání textu

"a" + "hoj" = "ahoj"

"12" + "34" = "1234"

DATOVÝ TYP bool

- **bool** = logická hodnota
- hodnoty: **true**, **false**
- základní logické operace
 - **AND** (součin) **true && false**
 - **OR** (součet) **true || false**
 - výsledky logických operací viz tabulka
- **&&** a **||** jsou tzv. logické operátory a používají se stejně jako např. + a -

vstup1	vstup2	výsledek operace	
		AND	OR
FALSE	FALSE	FALSE	FALSE
FALSE	TRUE	FALSE	TRUE
TRUE	FALSE	FALSE	TRUE
TRUE	TRUE	TRUE	TRUE

PŘEHLED DATOVÝCH TYPŮ

- **bool** = logická hodnota (true/false)
- **int** = celé číslo (-10, 0, 12, ...)
- **double** = desetinné číslo (-10.01, 0.0, 12.5, ...)
- **string** = řetězec znaků ("Czechitas", "x", "123" , "😊" , ...)
- **char** = znak
('C', 'z', 'e', 'c', 'h', 'i', 't', 'a', 's', 'x', '1', '2', '3')



NAČTENÍ VSTUPU

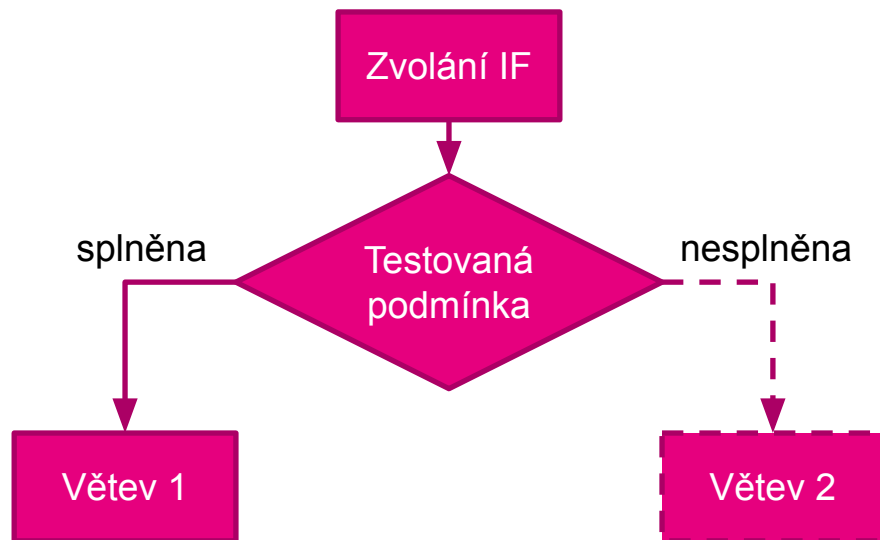
- Opět hotová funkce pro programátora.
- Použijeme v případě, že chceme, aby uživatel zadal hodnotu nějaké proměnné sám.
- Může se jednat např. o jméno, nebo o počet položek, se kterými má program pracovat.
- **Vstup vždy ukládáme do proměnné datového typu **string**.**

```
string vstupUzivatele = Console.ReadLine();
```

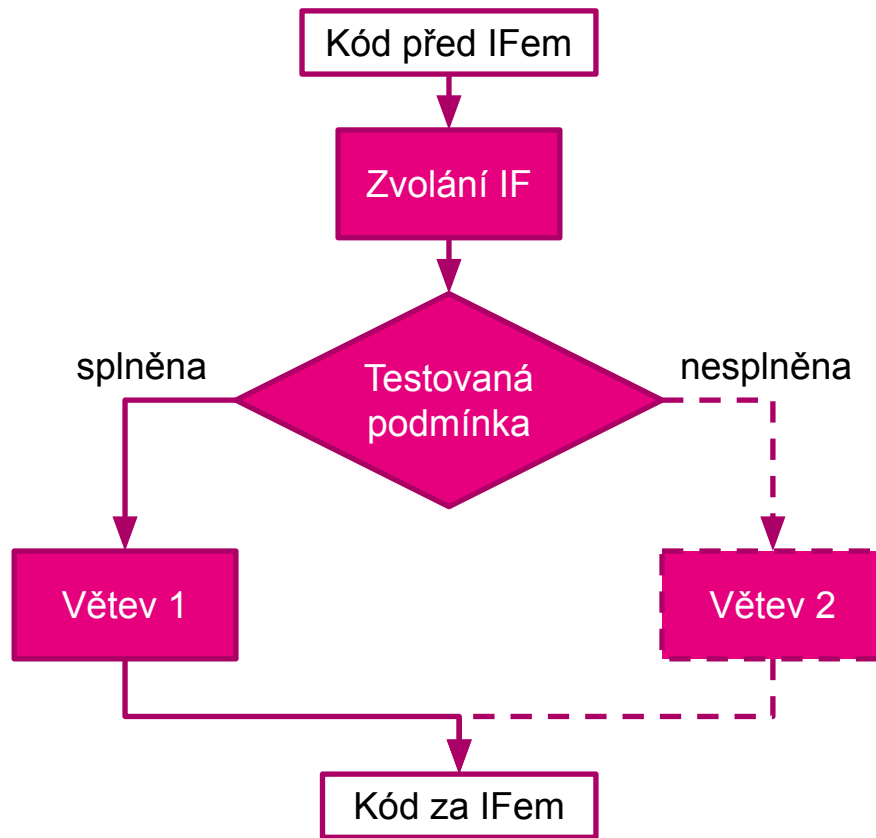
PODMÍNĚNÝ PŘÍKAZ



PODMÍNĚNÝ PŘÍKAZ if/else



PODMÍNĚNÝ PŘÍKAZ if/else



// Kód před IFem

```
if (testovanaPodminka)
{
    //Větev 1
}
else // nepovinné
{
    //Větev 2
}
```

// Kód za IFem

SLOŽENÉ ZÁVORKY

Ohraničují logický celek



PODMÍNĚNÝ PŘÍKAZ if/else

```
int vek = 15;
if (vek >= 18)
{
    // Větev 1 - podmínka splněna
    Console.WriteLine("Na zdraví!");
}
else
{
    // Větev 2 - podmínka nesplněna
    Console.WriteLine("Mladistvým nenaléváme.");
}
Console.WriteLine("Konec programu.");
```

PODMÍNĚNÝ PŘÍKAZ if/else

- Umožňuje větvit program pro různé možnosti podle splněných/nesplněných podmínek
- Podmínka je buď splněna nebo nesplněna -> existují jen dvě možnosti jejího vyhodnocení -> nový datový typ **bool**

DATOVÝ TYP bool

- **bool** = logická hodnota
- hodnoty: **true**, **false**
- základní logické operace
 - **AND** (součin) **true && false**
 - **OR** (součet) **true || false**
 - výsledky logických operací viz tabulka
- **&&** a **||** jsou tzv. logické operátory a používají se stejně jako např. + a -

vstup1	vstup2	výsledek operace	
		AND	OR
FALSE	FALSE	FALSE	FALSE
FALSE	TRUE	FALSE	TRUE
TRUE	FALSE	FALSE	TRUE
TRUE	TRUE	TRUE	TRUE

OPERÁTORY PRO POROVNÁVÁNÍ HODNOT

- Podobně jako v matematice, pozor na správný zápis:
 - větší než $10 > 10$... **false**
 - větší nebo rovno $10 >= 10$... **true**
 - menší než $5 < 10$... **true**
 - menší nebo rovno $9 <= 10$... **true**
 - rovno $"a" == "b"$... **false**
 - nerovno $"a" != "b"$... **true**
- Výsledkem porovnání je pak **bool**. Porovnání se používá v podmíněných příkazech a dá se skládat pomocí logických operátorů (&&, ||)

PODMÍNĚNÝ PŘÍKAZ if/else

- **else** není povinné, použije se pouze v případě, že chceme vykonat nějaké speciální příkazy v případě nesplnění podmínky
- složené závorky mohou obsahovat více příkazů dle potřeby

```
int cislo = 12;  
if (cislo > 10 && cislo < 25)  
{  
    Console.WriteLine("Jsi ve správném intervalu mezi 10 a 25.");  
    cislo = cislo + 5;  
}  
Console.WriteLine(cislo);
```

PODMÍNĚNÝ PŘÍKAZ `else if`

- Jedná se o rozšíření podmíněného příkazu v případě, že chceme vyhodnotit více podmínek zvlášť -> vytvoříme tím více možných větví programu.

PODMÍNĚNÝ PŘÍKAZ else if

```
int cislo = 12;  
if (cislo > 10)  
{  
    Console.WriteLine("Číslo je větší než 10");  
}  
else if (cislo == 10)  
{  
    Console.WriteLine("Číslo je rovno 10");  
}  
else  
{  
    Console.WriteLine("Číslo je menší než 10");  
}
```