Červená = návrh na změnu

Modrá = návrh čím nahradit

Zelená = poznámka

1. .NET \*\*neobsahuje\*\* platformu:

--

.NET Code

--

Xamarin

--

.NET Standard

--

.NET Core

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-uvod-do-jazyka-a-dot-net-framework}

2. Mezi vlastnosti jazyk~~u~~a C# \*\*nepatří\*\*:

--

Závislost na platformě, tedy na typu procesoru a operačním systému

--

Přenositelnost programů mezi různými platformami

//přenositelnost programů mezi platformami je **vlastnost** C#?

--

Stabilita

--

Málo zranitelný kód

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-uvod-do-jazyka-a-dot-net-framework}

3. C# je:

--

Jazyk s virtuálním strojem

--

Jazyk symbolických adres

--

Interpretovaný jazyk

--

Kompilovaný jazyk

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-uvod-do-jazyka-a-dot-net-framework}

4. Metoda, která se vykoná \*\*ihned po spuštění aplikace\*\*, se nazývá:

--

Main

--

Program

--

Console

--

App

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-instalace-a-ovladani-visual-studia-konzolova-aplikace}

5. Který z daných zápisů by do konzole vypsal text `Hello world`?

//Který kód vypíše do konzole `Hello world`?

--

/--code csharp

Console.WriteLine("Hello world");

\--

--

/--code csharp

Console.WriteLine(Hello world);

\--

--

/--code csharp

Console.WriteText("Hello world");

\--

--

/--code csharp

Console.Print("Hello world");

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-instalace-a-ovladani-visual-studia-konzolova-aplikace}

6. Metodu na třídě voláme pomocí operátoru:

//Proč je vše v závorkách? Zavádí to...

--

Tečka (\*\*.\*\*)

--

Středník (\*\*;\*\*)

--

Závorky (\*\*( )\*\*)

--

Ani jedna odpověď není správně

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-instalace-a-ovladani-visual-studia-konzolova-aplikace}

7. \*\*Statický\*\* typový systém:

--

Vyžaduje definovat typ proměnné a tento typ je dále neměnný

--

Nás plně odstiňuje od toho, že proměnná má vůbec nějaký datový typ

--

Umožňuje něco uložit do proměnné, která ještě nebyla deklarována a jazyk ji sám založí

--

Umožňuje ukládat např. text, potom objekt uživatele a potom desetinné číslo do té samé proměnné

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-promenne-typovy-system-a-parsovani}

// Je lepší hledat jednu chybnou odpověď mezi správnými tvrzeními než opačně

7. Které tvrzení je \*\*chybné\*\*:

--

Statický typový systém automaticky doplní datový typ proměnné

--

Statický typový systém vyžaduje definovat typ proměnné

--

Statický typový systém neumožňuje datový typ měnit

--

Díky statickému typovému systému může kompiler datové typy zkontrolovat

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-promenne-typovy-system-a-parsovani}

8. Který z daných zápisů je správně, jestliže chceme od uživatele získat \*\*celé číslo\*\*?

--

/--code csharp

string vstup = Console.ReadLine();

int a = int.Parse(vstup);

// int a = int.Parse(Console.ReadLine());

\--

--

/--code csharp

string vstup = Console.ReadKey();

float b = float.Parse(vstup);

\--

--

/--code csharp

int c = Console.ReadLine();

\--

--

/--code csharp

string vstup = Console.ReadLine();

int d = int.ConvertToInt(vstup);

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-promenne-typovy-system-a-parsovani}

9. Který z daných zápisů vypíše do konzole \*\*číslo 8\*\*?

--

/--code csharp

int a = 5;

int b = 3;

Console.WriteLine(a + b);

\--

--

/--code csharp

Console.WriteLine(3 + “5“);

\--

--

/--code csharp

Console.WriteLine(“3“ + “5“);

\--

--

/--code csharp

int a = 5;

int b = “3“;

Console.WriteLine(a + b);

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-promenne-typovy-system-a-parsovani}

//nevytvářejme zbytečně chybné kódy...

9. Který kód \*\*nevypíše\*\* do konzole \*\*8\*\*?

--

/--code csharp

string cislo = "5";

Console.WriteLine(cislo + "3");

\--

--

/--code csharp

int a = 5;

int b = 3;

Console.WriteLine(a + b);

\--

--

/--code csharp

int a = 6;

Console.WriteLine(2 + a);

\--

--

/--code csharp

Console.WriteLine(4 + 4);

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-promenne-typovy-system-a-parsovani}

10. Kód:

/--code csharp

string text = " C# je nejlepší! ";

Console.WriteLine(text.ToUpper().TrimEnd());

\--

Vypíše:

--

`C# JE NEJLEPŠÍ!` (s mezerou na začátku)

--

`C# JE NEJLEPŠÍ!` (bez mezer na začátku a na konci)

--

`C# je nejlepší!` (bez mezer na začátku a na konci)

--

`C# JE NEJLEPŠÍ!` (s mezerou na začátku i na konci)

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-typovy-system-podruhe-datove-typy-string}

11. Kód:

/--code csharp

string text = "C# je NEJLEPŠÍ!";

Console.WriteLine(text.Replace("Java", "C#").ToLower());

\--

Vypíše:

--

`c# je nejlepší!`

--

`java je nejlepší!`

--

`C# je nejlepší!`

--

`java je NEJLEPŠÍ!`

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-typovy-system-podruhe-datove-typy-string}

12. Který z daných zápisů \*\*vypíše\*\* do konzole `False`? //Super

--

/--code csharp

string s = "False";

Console.WriteLine(s.StartsWith("f"));

\--

--

/--code csharp

bool b = 16 < 128;

Console.WriteLine(b);

\--

--

/--code csharp

string s = "Nepravda";

Console.WriteLine(s.EndsWith("pravda"));

\--

--

/--code csharp

string s = "Pravda";

Console.WriteLine(s.Contains("a"));

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-typovy-system-podruhe-datove-typy-string}

13. Který z daných zápisů konstrukce `switch` je \*\*správný\*\*?

//Který `switch` kód je \*\*správný\*\*?

--

/--code csharp

int cislo = 1;

switch (cislo)

{

case 1:

Console.WriteLine("Číslo je rovno 1");

break;

case 2:

Console.WriteLine("Číslo je rovno 2");

break;

}

\--

--

/--code csharp

int cislo = 1;

switch

{

case (cislo == 1):

Console.WriteLine("Číslo je rovno 1");

break;

case (cislo == 2):

Console.WriteLine("Číslo je rovno 2");

break;

}

\--

--

/--code csharp

int cislo = 1;

switch

{

cislo == 1:

Console.WriteLine("Číslo je rovno 1");

break;

cislo == 2:

Console.WriteLine("Číslo je rovno 2");

break;

}

\--

--

/--code csharp

int cislo = 1;

switch (cislo)

{

case == 1:

Console.WriteLine("Číslo je rovno 1");

break;

case == 2:

Console.WriteLine("Číslo je rovno 2");

break;

}

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-podminky-vetveni-if-switch}

14. Kód:

/--code csharp

int a = 0;

if (a != 0)

Console.WriteLine(5 + a);

else

if (a < 0)

Console.WriteLine(5 / a);

else

Console.WriteLine("Nulou nelze dělit!");

\--

Vypíše:

--

`Nulou nelze dělit!`

--

Způsobí chybu programu

--

`5`

--

Nic

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-podminky-vetveni-if-switch}

15. Podmínku rozhodující, jestli číslo (`a`) \*\*patří\*\* do intervalu `<0;5)`, bychom zapsali:

//proč je **a** v závorkách?; ne všichni znají otevřený a uzavřený interval... v kurzu to myslím nebylo zmiňováno

--

/--code csharp

if ((a >= 0) && (a < 5))

\--

--

/--code csharp

if ((a >= 0) || (a < 5))

\--

--

/--code csharp

if (a < 5)

\--

--

/--code csharp

if ((a > 0) || (a < 5))

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-podminky-vetveni-if-switch}

16. Syntaxe (zápis) cyklu \*\*for\*\* je:

--

/--code csharp

for (promenna; podminka; prikaz)

\--

--

/--code csharp

for (prikaz; podminka; promenna)

\--

--

/--code csharp

for (podminka; prikaz)

\--

--

/--code csharp

for (podminka)

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-cykly-for-while}

17. Cyklus `while`:

--

Jen opakuje příkazy v bloku dokud platí podmínka

//Opakuje příkazy v bloku dokud platí podmínka

--

Má stanovený pevný počet opakování

--

Obsahuje tzv. řídící proměnnou (celočíselnou), ve které se postupně během běhu cyklu mění hodnoty

--

Není cyklus

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-cykly-for-while}

18. Která z ukázek \*\*nevypíše\*\* desetkrát slovo `Cykl`? //Super

--

Všechny ukázky vypíší desetkrát `Cykl`

--

/--code csharp

int i = 0;

while(i < 10)

{

Console.WriteLine("Cykl");

i++;

}

\--

--

/--code csharp

for (int i = 0; i < 10; i++)

Console.WriteLine("Cykl");

\--

--

/--code csharp

for (int i = 10; i > 0; i--)

Console.WriteLine("Cykl");

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-cykly-for-while}

19. Kód: //Super

/--code csharp

float f = 1.5f;

while(float.TryParse(Console.ReadLine(), out f))

{

Console.WriteLine(f);

}

\--

Vypíše:

--

Uživatelem zadaná čísla dokud budou správně zadávána

--

`1,5` dokud uživatel správně nezadá nějaké číslo

--

`Neplatné číslo, zadejte prosím znovu:`

--

Vždy `1,5`

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-osetreni-uzivatelskych-vstupu}

20. O poli v ~~#~~C# \*\*neplatí\*\*:

--

Nemůžeme založit pole s velikostí, kterou dynamicky zadáme až za běhu programu

--

Za běhu programu nemůžeme modifikovat velikost existujícího pole

--

Všechny prvky pole jsou očíslované tzv. indexy

--

Uchovává větší množství proměnných stejného typu

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-pole}

21. Která z ukázek \*\*neobsahuje chybu\*\*?

--

/--code csharp

int[] pole = new int[5];

\--

--

/--code csharp

int pole[] = new pole(5);

\--

--

/--code csharp

int[] pole = int[5];

\--

--

/--code csharp

int[] pole = new int[](5);

\--

{ARTICLE <URL:csharp/zaklady/c-sharp-tutorial-pole>}

//neučíme chyby

21. Který kód \*\*obsahuje\*\* chybu?

--

/--code csharp

int[] pole = new int[](5);

\--

--

/--code csharp

int[] pole = new int[5];

\--

--

/--code csharp

int[] pole = { 2, 4, 6, 8, 10 };

\--

--

/--code csharp

int[] pole = new int[5] { 2, 4, 6, 8, 10 };

\--

{ARTICLE <URL:csharp/zaklady/c-sharp-tutorial-pole>}

22. Která z ukázek \*\*nevypíše\*\* všechny prvky celočíselného pole `array` do konzole?

//Která z ukázek nevypíše pole, ale \*\*způsobí chybu\*\*?

--

/--code csharp

foreach(int i in array)

Console.WriteLine(array[i]);

\--

--

/--code csharp

for (int i = 0; i <= array.Length - 1; i++)

Console.WriteLine(array[i]);

\--

--

/--code csharp

foreach(int prvek in array)

Console.WriteLine(prvek);

\--

--

/--code csharp

for (int i = 0; i < array.Length; i++)

Console.WriteLine(array[i]);

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-pole}

23. Kód:

/--code csharp

int[] cisla = { 6, 3, 5, 3, 3, 4, 9, 1 };

Array.Reverse(cisla);

Array.Sort(cisla);

foreach(int cislo in cisla)

Console.Write("{0} ", cislo);

\--

Vypíše:

--

Seřazené prvky pole od nejmenšího po největší

--

Seřazené prvky pole od největšího po nejmenší

--

Jen unikátní prvky pole seřazené od nejmenšího po největší

--

Prvky pole pozpátku

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-pole}

24. Kód:

/--code csharp

string s = "ahoj";

Console.WriteLine((int)s[0]);

\--

Vypíše:

--

ASCII kód písmene `a`

--

Písmeno `a`

--

Způsobí chybu programu

--

Nic

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-textove-retezce-podruhe-prace-se-znaky}

25. Která z ukázek \*\*nevypíše\*\* do konzole text zadaný uživatelem?

--

/--code csharp

string text = Console.ReadLine();

foreach(string s in text)

Console.Write(s);

\--

--

/--code csharp

string text = Console.ReadLine();

for (int i = 0; i < text.Length; i++)

Console.Write(text[i]);

\--

--

/--code csharp

string text = Console.ReadLine();

foreach(char c in text)

Console.Write(c);

\--

--

/--code csharp

string text = Console.ReadLine();

Console.Write(text);

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-textove-retezce-podruhe-prace-se-znaky}

26.

/--code csharp

string s = "text";

s[1] = "x";

\--

Tento kód:

--

Vyhodí chybovou hlášku, jelikož znaky na dané pozici řetězce jsou v C# read-only

--

Vyhodí chybovou hlášku, jelikož ke znakům v řetězci nemůžeme přistupovat přes hranaté závorky

--

Dosadí za první písmeno ~~řeťezce~~ řetězce písmeno `x`

--

Dosadí za druhé písmeno ~~řeťezce~~ řetězce písmeno `x`

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-textove-retezce-podruhe-prace-se-znaky}

27. Kód:

/--code csharp

string s = "Jan;Marek;Jana;Radim";

Console.WriteLine(string.Join(" ", s.Split(';')));

\--

Vypíše:

--

`Jan Marek Jana Radim`

--

`Jan;Marek;Jana;Radim`

--

`JanMarekJanaRadim JanMarekJanaRadim`

--

Způsobí chybu programu

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-textove-retezce-do-tretice-split-join}

28. Modifikátor, který escapuje \*\*celý\*\* námi napsaný řetězec v kódu (např. cestu k souboru), je:

--

`@`

--

`\n`

--

`\t`

--

`=>`

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-textove-retezce-do-tretice-split-join}

29. Kód:

/--code csharp

Console.WriteLine(@"C:\Users\Ja\Dropbox".Insert(1, "D").Remove(0,1));

\--

Vypíše:

--

`D:\Users\Ja\Dropbox`

--

`CD:\Users\Ja\Dropbox`

--

`"D:\Users\Ja\Dropbox"`

--

`C:UsersJaDropbox`

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-textove-retezce-do-tretice-split-join}

30. Syntaxe (zápis) \*\*ternárního výrazu\*\* je:

--

`(podminka) ? hodnota1 : hodnota2`

--

`(parametr) => { prikaz }`

--

`(podminka) ? (hodnota1) / (hodnota2)`

--

`(podminka) : hodnota1 ? hodnota2`

{ARTICLE URL:csharp/zaklady/podminky-v-csharp-podruhe-ternarni-vyraz-a-propadavaci-switch}

31. Kód:

/--code csharp

int i = 0;

string s = (i > 0) ? "A" : (i < 0) ? "B" : "C";

Console.WriteLine(s);

\--

Vypíše:

--

`C`

--

`A`

--

`B`

--

Nic. C# neobsahuje ternární výraz.

//Způsobí chybu

{ARTICLE URL:csharp/zaklady/podminky-v-csharp-podruhe-ternarni-vyraz-a-propadavaci-switch}

32. Cyklus `do-while`:

--

Se vždy vykoná nejméně jednou

--

Se v ničem neliší od cyklu `while`

--

Má vždy omezený počet opakování

--

V C# vůbec neexistuje

{ARTICLE URL:csharp/zaklady/cykly-v-csharp-net-podruhe-do-while-break-a-continue}

//méně nesmyslů, více informací

32. Které tvrzení \*\*neplatí\*\*:

--

Cyklus `do-while` se v ničem neliší od cyklu `while`

--

Cyklus `do-while` se vykoná nejméně jednou

--

V cyklu `do-while` je podmínka umístěna až za tělem cyklu

--

Pro automatické vypsání cyklu `do-while` napíšeme `do` a dvakrát stiskneme `Tabulátor`

{ARTICLE URL:csharp/zaklady/cykly-v-csharp-net-podruhe-do-while-break-a-continue}

33. Syntaxe (zápis) cyklu \*\*`do-while`\*\* je:

--

/--code csharp

do

{

// kód...

} while (podminka);

\--

--

/--code csharp

while (podminka) do

{

// kód...

}

\--

--

/--code csharp

do while (podminka)

{

// kód...

}

\--

--

/--code csharp

do (podminka1)

{

// kód...

} while (podminka2);

\--

{ARTICLE <URL:csharp/zaklady/cykly-v-csharp-net-podruhe-do-while-break-a-continue>}

33. Která syntaxe (zápis) cyklu \*\*obsahuje chybu\*\*?

--

/--code csharp

while (true)

{

// kód...

}

do (false)

\--

--

/--code csharp

for (int i = 0; i < length; i++)

{

// kód...

}

\--

--

/--code csharp

foreach (var item in collection)

{

// kód...

}

\--

--

/--code csharp

do

{

// kód...

} while (true);

\--

{ARTICLE <URL:csharp/zaklady/cykly-v-csharp-net-podruhe-do-while-break-a-continue>}

34. Která z ukázek \*\*vypíše\*\* tento řetězec čísel \*\*`12356`\*\*?

--

/--code csharp

for(int i = 1; i <= 6; i++)

{

if(i == 4)

continue;

Console.Write(i);

}

\--

--

/--code csharp

for(int i = 1; i <= 6; i++)

{

if(i == 4)

break;

Console.Write(i);

}

\--

--

/--code csharp

for(int i = 1; i <= 6; i++)

{

if(i == 4)

return;

Console.Write(i);

}

\--

--

Žádná z uvedených ukázek.

{ARTICLE URL:csharp/zaklady/cykly-v-csharp-net-podruhe-do-while-break-a-continue}

35. \*\*Dvourozměrné\*\* celočíselné pole v C# deklarujeme:

--

/--code csharp

int[,] pole = new int[2, 3];

\--

--

/--code csharp

int[2D] pole = new int[2D](2,3);

\--

--

/--code csharp

int[,] pole = new int[,](2,3);

\--

--

/--code csharp

int[2\*3] pole = new int[2\*3];

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-vicerozmerna-pole}

35. Který zápis 2D pole je \*\*chybný\*\*?

--

/--code csharp

int[,] pole = new int[2D] {4, 2};

\--

--

/--code csharp

int[,] pole = new int[4, 2];

\--

--

/--code csharp

int[,] pole = new int[,] { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };

\--

--

/--code csharp

int[,] pole = new int[4, 2] { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-vicerozmerna-pole}

36. Která z daných ukázek \*\*obsahuje chybu\*\*?

--

Žádná

--

/--code csharp

int[][] pole = new int[9][];

\--

--

/--code csharp

int[,,] pole = new int[7,9,10];

\--

--

/--code csharp

int[,] pole = new int[7,7];

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-vicerozmerna-pole}

37. Jak se deklaruje celočíselné 2D \*\*pole polí\*\* (tzv. zubaté pole)?

--

/--code csharp

int[][] pole = new int[3][];

\--

--

/--code csharp

int[]int[] pole = new int[8]int[];

\--

--

/--code csharp

int[int[]] pole = new int[int[8]];

\--

--

/--code csharp

int[][] pole = new int[int[8]];

\--

{ARTICLE <URL:csharp/zaklady/c-sharp-tutorial-vicerozmerna-pole>}

37. Které tvrzení je \*\*chybné\*\*?

--

2D pole polí plníme vždy dvěma for cykly

--

Pole polí deklarujeme takto:

/--code csharp

int[][] pole = new int[3][];

\--

--

U 2D pole polí platí, že si do každého řádku/sloupce můžeme uložit jak velké pole chceme

--

K hodnotám 2D pole polí přistupujeme pomocí 2 indexerů

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-vicerozmerna-pole}

38. Která z ukázek nám vypíše zbytek po celočíselném dělení (tzv. modulo)?

--

/--code csharp

Console.WriteLine(7 % 3);

\--

--

/--code csharp

Console.WriteLine(7 / 3);

\--

--

/--code csharp

Console.WriteLine(7 /! 3);

\--

--

/--code csharp

Console.WriteLine(7 ! 3);

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-knihovna-math-matematicke-funkce}

39. Kód:

/--code csharp

int i = (int)Math.Round(Math.Min(3.5, 1.3));

Console.WriteLine(i);

\--

Vypíše:

--

`1`

--

`4`

--

Způsobí chybu programu

--

`1,3`

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-knihovna-math-matematicke-funkce}

40. Která z ukázek ~~s~~vypočítá 2^8?

--

/--code csharp

double d = Math.Pow(2, 8);

\--

--

/--code csharp

double d = Math.Pow(8, 2);

\--

--

/--code csharp

double d = Math.Sqrt(2, 8);

\--

--

/--code csharp

double d = 2^8;

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-knihovna-math-matematicke-funkce}

41. Metody třídy Array a metody volané na instanci pole se liší v tom, že:

--

Metody volané na instanci pole vracejí nově vytvořené pole a metody třídy Array upravují dané pole.

--

Metody volané na instanci pole upravují dané pole a metody třídy Array vracejí nově vytvořené pole.

--

V ničem se neliší.

--

Metody třídy Array jsou bezparametrické.

{ARTICLE <URL:csharp/zaklady/c-sharp-tutorial-pole>}

42. Která z ukázek \*\*nevypíše\*\* poslední prvek pole `array`? //Super

--

/--code csharp

Console.WriteLine(array[array.Length]);

\--

--

/--code csharp

Console.WriteLine(array[array.Length - 1]);

\--

--

/--code csharp

Console.WriteLine(array.Reverse().First());

\--

--

/--code csharp

Console.WriteLine(array.Last());

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-pole}

43. Jakým klíčovým slovem se ve `switch` označuje \*\*výchozí větev\*\*?

--

`default`

--

`case`

--

`break`

--

`else`

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-podminky-vetveni-if-switch}

44. Kód: //Super

/--code csharp

for (int i = 1; i <= 5; i++)

{

for (int j = 1; j <= 5; j++)

{

if (j < i)

Console.Write(" ");

else if (i == j)

Console.Write(i);

else

break;

}

Console.WriteLine();

}

\--

Vypíše:

--

/--code csharp

1

2

3

4

5

\--

--

/--code csharp

1

22

333

4444

55555

\--

--

/--code csharp

1

12

123

1234

12345

\--

--

/--code csharp

1

2

3

4

5

\--

{ARTICLE URL:csharp/zaklady/cykly-v-csharp-net-podruhe-do-while-break-a-continue}

45. Čím se od sebe liší \*\*unsigned\*\* a \*\*signed\*\* datové typy?

--

Unsigned typy \*\*neumožňují záporné hodnoty\*\* a na \*\*kladnou\*\* část tak mohou uložit \*\*2x vyšší\*\* hodnotu než signed typy.

--

Signed typy \*\*neumožňují záporné hodnoty\*\* a na \*\*kladnou\*\* část tak mohou uložit \*\*2x vyšší\*\* hodnotu než unsigned typy.

--

Unsigned typy \*\*neumožňují kladné hodnoty\*\* a na \*\*zápornou\*\* část tak mohou uložit \*\*2x vyšší\*\* hodnotu než signed typy.

--

Signed typy \*\*neumožňují kladné hodnoty\*\* a na \*\*zápornou\*\* část tak mohou uložit \*\*2x vyšší\*\* hodnotu než unsigned typy.

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-typovy-system-podruhe-datove-typy-string}

45. Které tvrzení je \*\*chybné\*\*?

--

Ubyte má dvojnásobnou hodnotu připadající na kladné číslo: rozsah 0 až 510.

--

Unsigned typy neumožňují záporné hodnoty a na kladnou část tak mohou uložit 2x vyšší hodnotu než signed typy.

--

Signed jsou celočíselné datové typy umožňující jak kladné, tak záporné hodnoty.

--

Díky implicitní konverzi, mezi celočíselnými datovými typy, můžeme přiřadit int do proměnné typu long bez explicitní konverze.

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-typovy-system-podruhe-datove-typy-string}

46. Kód:

/--code csharp

Console.WriteLine(5 / 2);

\--

Vypíše:

--

`2`

--

`2,5`

--

`5 / 2`

--

Způsobí chybu programu

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-knihovna-math-matematicke-funkce}

47. Jak získáme od uživatele \*\*1 znak\*\* (`char`)?

--

/--code csharp

char c = Console.ReadKey().KeyChar;

\--

--

/--code csharp

char c = Console.ReadLine().ToChar();

\--

--

/--code csharp

char c = Console.ReadLine().KeyChar;

\--

--

/--code csharp

char c = Console.ReadKey();

\--

{ARTICLE URL:csharp/zaklady/c-sharp-tutorial-osetreni-uzivatelskych-vstupu}