2-3 Algoritmus, analýza, značky vývojového diagramu

15.9.2017

Algoritmizace

Algoritmus je schematický postup pro řešení určitého druhu problémů,
který je prováděn pomocí konečného množství přesně definovaných kroků.
Algoritmus musí splňovat:

- **Konečnost** - Každý algoritmus musí skončit v konečném počtu kroků.

Tento počet kroků může být libovolně velký (podle rozsahu a hodnot vstupních údajů),
ale pro každý jednotlivý vstup musí být konečný.

- **Determinovanost** - Každý krok algoritmu musí být jednoznačně a přesně definován;

v každé situaci musí být naprosto zřejmé, co a jak se má provést,
jak má provádění algoritmu pokračovat.
Protože běžný jazyk obvykle neposkytuje naprostou přesnost a jednoznačnost vyjadřování,
byly pro zápis algoritmů navrženy programovací jazyky,
ve kterých má každý příkaz jasně definovaný význam.
Vyjádření výpočetní metody v programovacím jazyce se nazývá program.

- **Efektivnost** - Obecně požadujeme, aby algoritmus byl efektivní, v tom smyslu, že požadujeme,

aby každá operace požadovaná algoritmem, byla dostatečně jednoduchá na to,
aby mohla být alespoň v principu provedena v konečném čase pouze s použitím tužky a papíru.

- **Obecnost** - Algoritmus neřeší jeden konkrétní problém (např. „jak spočítat 3×7“),

ale obecnou třídu obdobných problémů (např. „jak spočítat součin dvou celých čísel“).

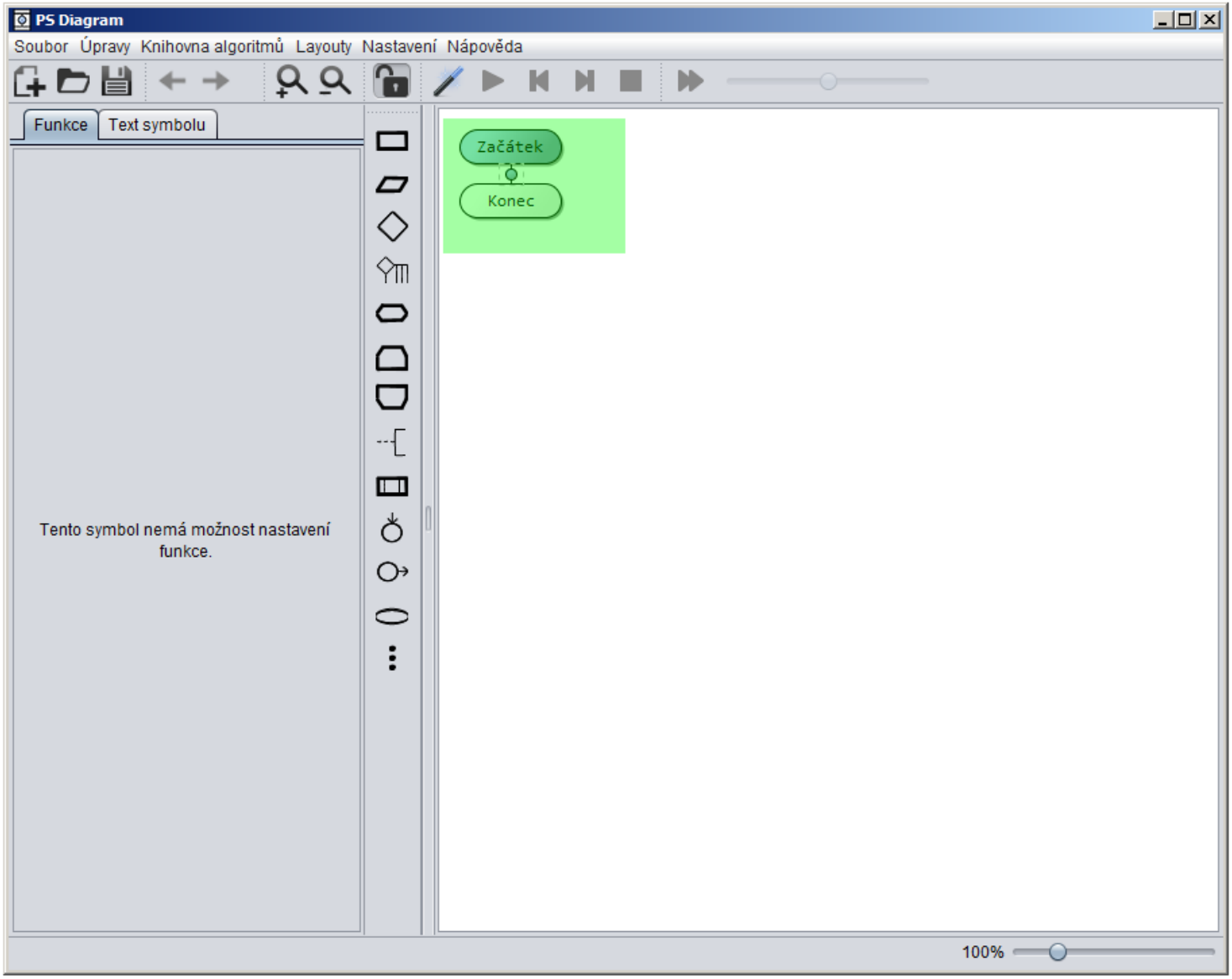
- **Rezultativnost** - Algoritmus při zadání vstupních dat vždy vrátí nějaký výsledek

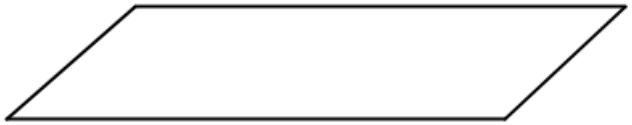
(může se jednat i jen o chybové hlášení).

Reprezentace a zápis algoritmů:
Slovním popisem - slovní popis v přirozeném jazyce
Vývojovým diagramem (blokové schéma) -
grafické znázornění algoritmu řešení úlohy jednotnými značkami a zkratkami
(tento způsob kombinovaný se slovním popisem zde budu používat, tudíž přibude i článek na toto téma).
Strukturogramy - používá obdobné symboly ale přesnější,
tento systém přesně splňuje podmínky důležité pro strukturované programování
Datově orientované diagramy HIPO - (z angl. Hierarchy plus Input- Process- Output),
je grafickým vyjádřením funkčního členění problému, struktury dat a postupu řešení problému při různém stupni podrobnosti.
Rozhodovací tabulky - používané při velmi složitých větveních.

PS diagram
- odevzdávárna
- uu
- http://www.psdiagram.cz/

Z:
PVA
- písemky
- dú
- cvičení
- PS diagram


PS Diagram
Soubor Úpravy Knihovna algoritmů Layouty Nastavení Nápověda
Funkce
Text symbolu
Tento symbol nemá možnost nastavení funkce.
Začátek
Konec
100%



Vstup/výstup

jméno:
a-zA-Z1-9_
a!=A
unikátní - jednoznačný
nesmím použít klíčová slova
!!ne jména tříd, vlastností, metod...
poznám o co jde
Velbloudí konvence:
ProměnnáProPrvníČíslo
proměnnáproprvníčíslo

česky
česky bez háčků a čárek
anglicky

vnitřní operace

skládání řetězce

zadávám text >> ERR
cislo1 <- 5
cislo2 <- 10
cislo1 + cislo2 >> 15
"zadávám text" + cislo1 >>
"zadávám text5"

c1
c2
c1 > 0 a zároveň c2 > 0
c1 > 0 && c2 > 0

& >> pravý alt + c
pravý Alt <> levý Alt + Ctrl

cv:
zeptat se, jestli chce Obdélník >> zadá o nebo Čtverec >> zadá c
cokoli jiného ukončí program

>> Obdélník >> vstup c1, c2, výpočet obvod a plocha
>> Čtverec >> vstup c1, výpočet obvod a plocha

pokud do řetězce (do "") zadáte **\n >> zalomení řádku**

"ahoj babi.\nJak se máš?\nJá dobře.\nOlina"

"ahoj babi.
Jak se máš?
Já dobře.
Olina"

dů vytvořte vývojový diagram v PD Diagramu

1: načtěte 2 "čísla"
- >>>> 5 = 5
- >>>> 5 < 10
- >>>> 10 > 5

2 jako 1 s ohlídáním intervalu <-10,10>
- mimo interval >> chyba

OK >> jako 1

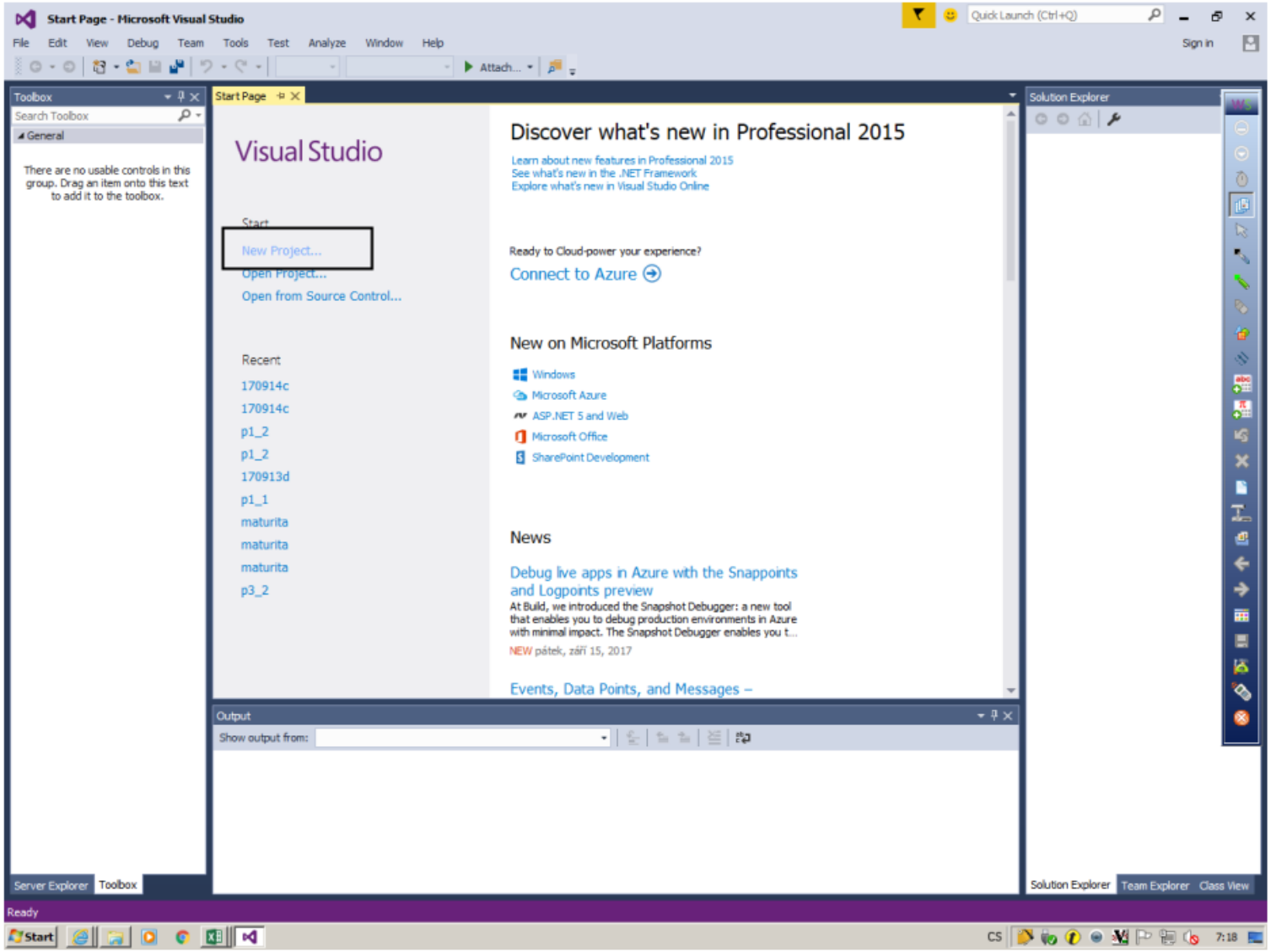
3 zadat 3 čísla
- od kolika
- do kolika
- kolik přičítat
- >> vypis čísel v cyklu

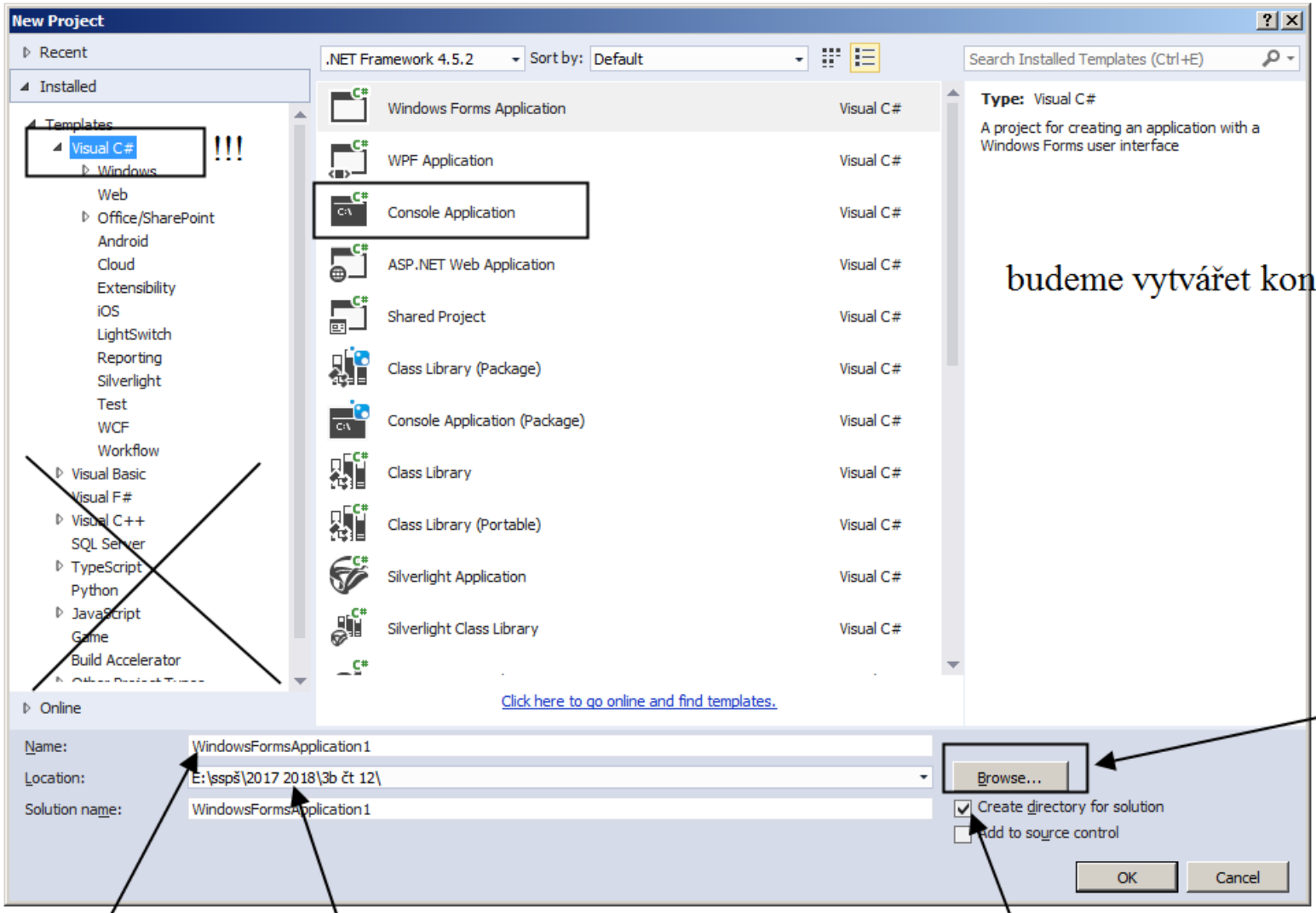
4 u příkladu 3
- ZKUSTE VHODNĚ URČIT INTERVALY A OHLÍDAT

od 5, do 10 po 1

5
6
7
8
...
10

Microsoft Visual Studio 2015





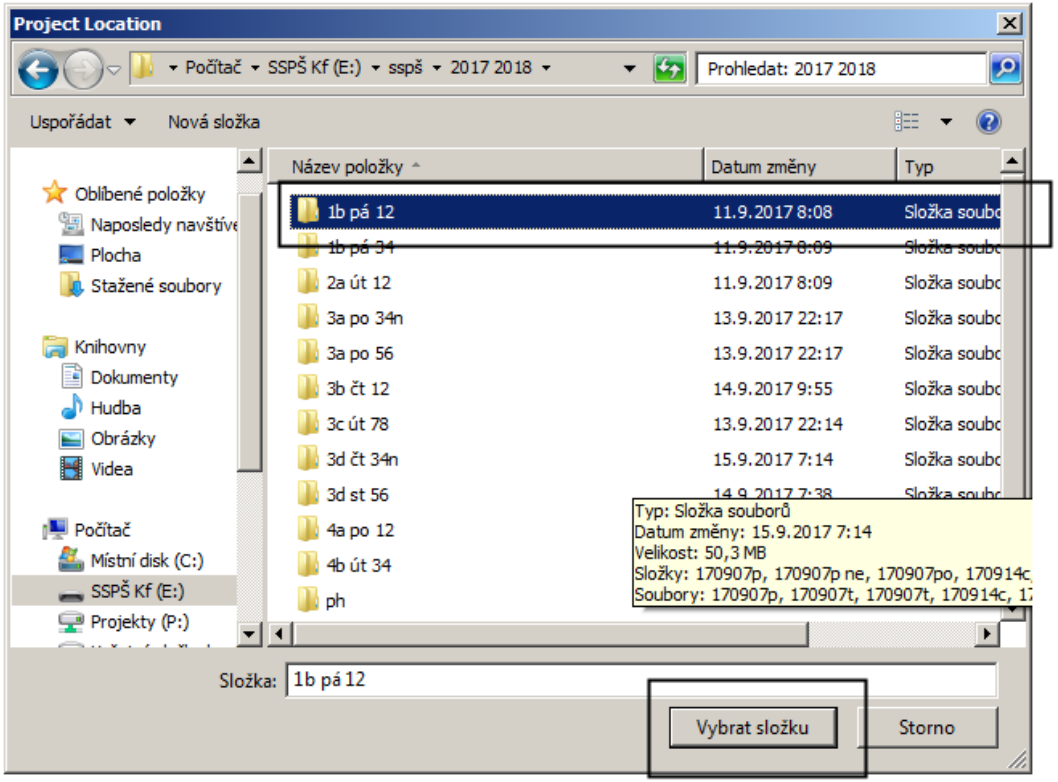
budeme vytvářet konzolovou aplikaci

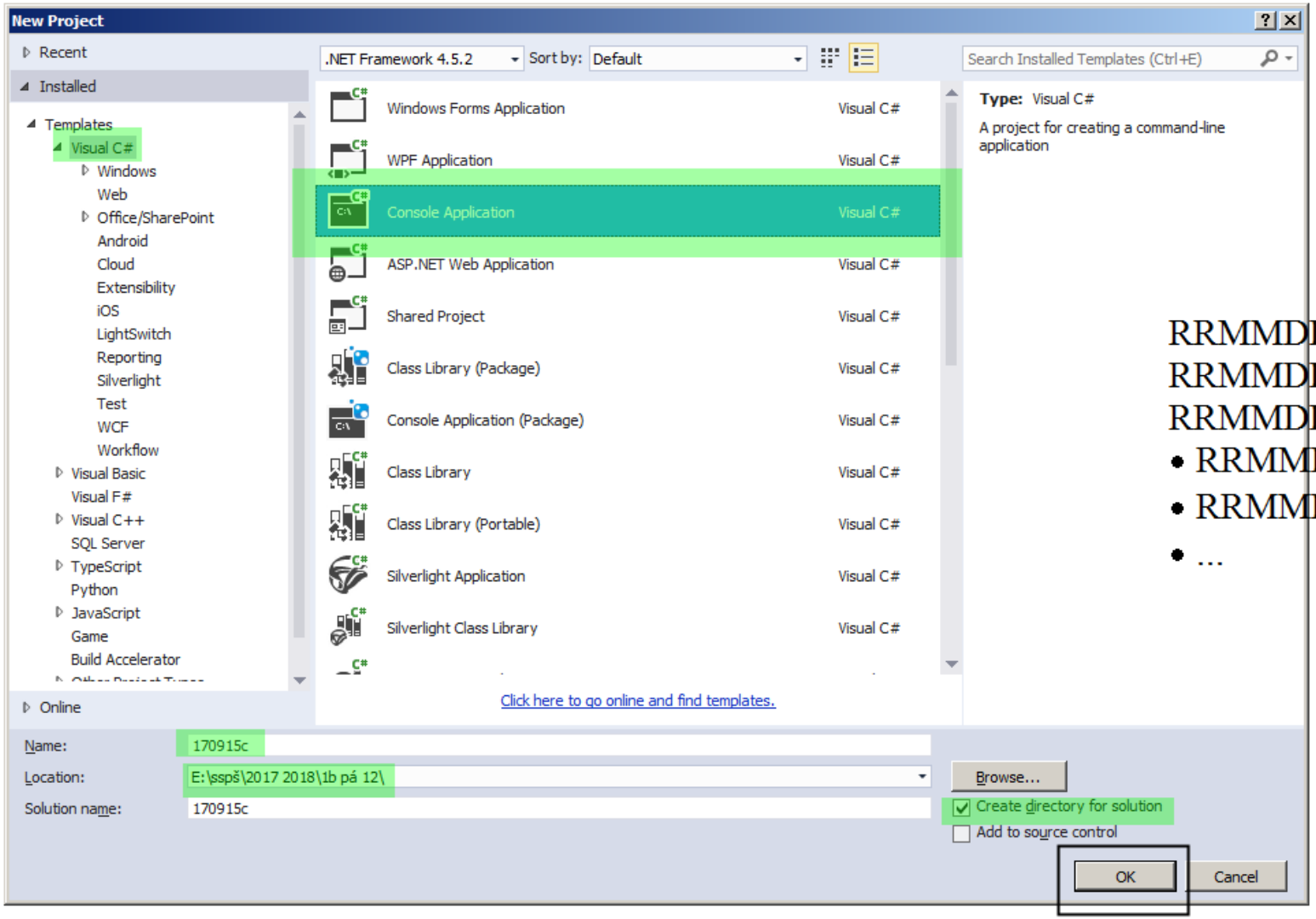
změnit cestu

jméno projektu

cesta k uložení projektu

nechat zaškrtnuté
Vytvoří samostatnou složku projektu

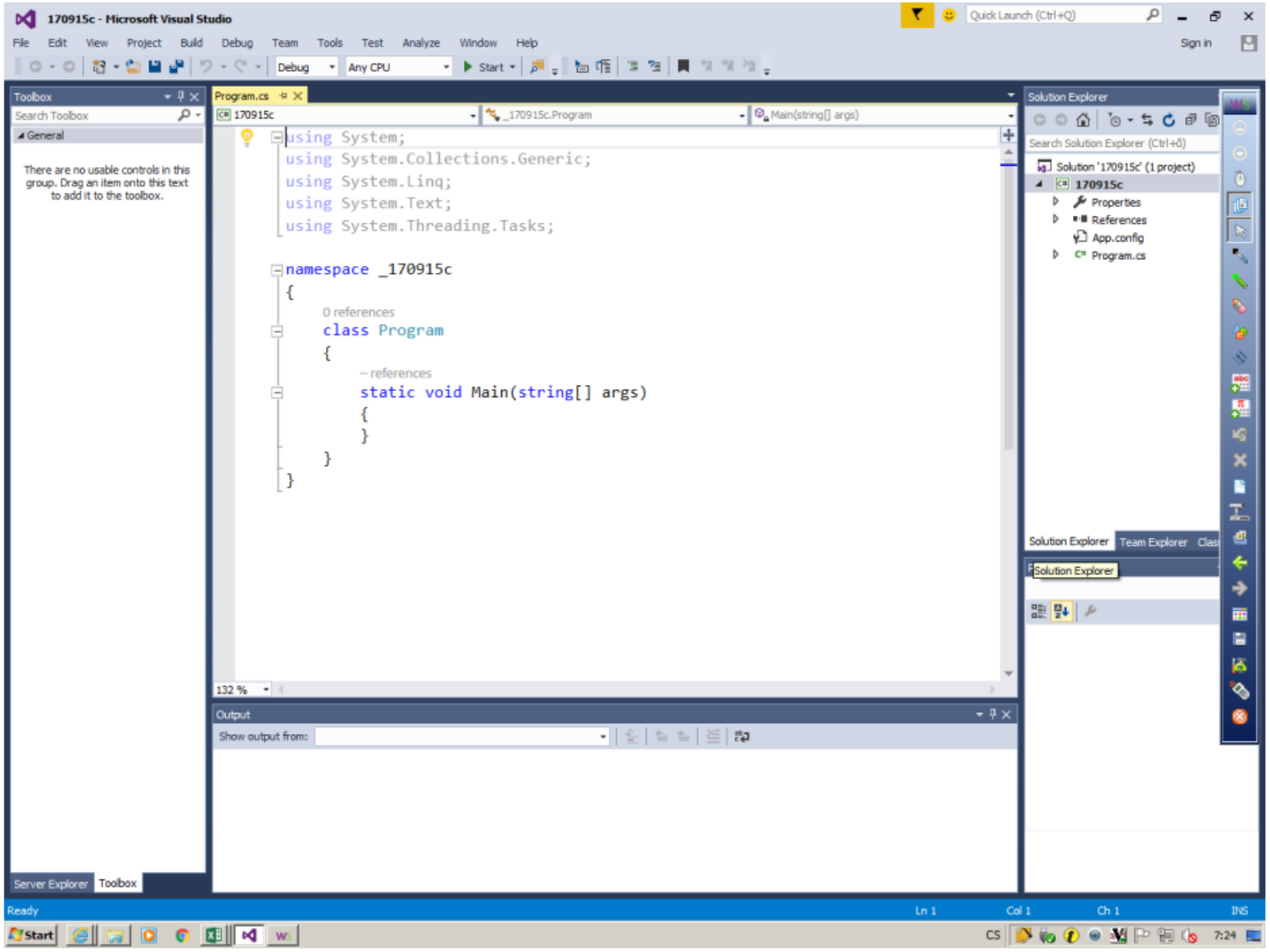

Project Location
Počítač ▸ SSPŠ Kf (E:) ▸ sspš ▸ 2017 2018 ▸
Prohledat: 2017 2018
Uspořádat
Nová složka
Oblíbené položky
Naposledy navštíve
Plocha
Stažené soubory
Knihovny
Dokumenty
Hudba
Obrázky
Videa
Počítač
Místní disk (C:)
SSPŠ Kf (E:)
Projekty (P:)
Název položky
Datum změny
Typ
1b pá 12 11.9.2017 8:08 Složka soubo
1b pá 34 11.9.2017 8:09 Složka soubo
2a út 12 11.9.2017 8:09 Složka soubo
3a po 34n 13.9.2017 22:17 Složka soubo
3a po 56 13.9.2017 22:17 Složka soubo
3b čt 12 14.9.2017 9:55 Složka soubo
3c út 78 13.9.2017 22:14 Složka soubo
3d čt 34n 15.9.2017 7:14 Složka soubo
3d st 56
4a po 12
4b út 34
ph
Typ: Složka souborů
Datum změny: 15.9.2017 7:14
Velikost: 50,3 MB
Složky: 170907p, 170907p ne, 170907po, 170914c
Soubory: 170907p, 170907t, 170907t, 170914c,
Složka: 1b pá 12
Vybrat složku
Storno



RRMMDDp
RRMMDDps
RRMMDDpo

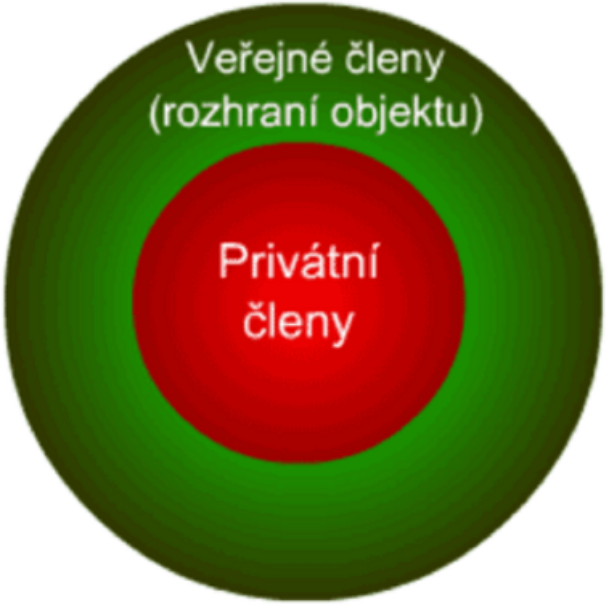
- RRMMDD Amjid
- RRMMDD Bárta
- …

RRMMDDc RRMMDDd RRMMDDt
RRMMDDcs RRMMDDds


170915c - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Sign in
Debug
Any CPU
Start
Toolbox
Search Toolbox
General
There are no usable controls in this group. Drag an item onto this text to add it to the toolbox.
Server Explorer
Toolbox
Program.cs
170915c
_170915c.Program
Main(string[] args)
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
namespace _170915c
{
0 references
class Program
{
0 references
static void Main(string[] args)
{
}
}
}
132 %
Output
Show output from:
Solution Explorer
Search Solution Explorer (Ctrl+ü)
Solution '170915c' (1 project)
170915c
Properties
References
App.config
Program.cs
Solution Explorer
Team Explorer
Solution Explorer
Ready
Ln 1
Col 1
Ch 1
INS
Start
CS
7:24

Zapouzdření objektů znamená, že objekt má některé své členy (metody/atributy) před okolím skryty.

Členy přístupné okolí se nazývají **rozhraní** objektu.



Hodnotové typy (value types) -
do této skupiny patří všechny číselné datové typy, typ char a ostatní struktury.
U těchto jednoduchých typů se jejich hodnota ukládá přímo do proměnné
- místa v paměti určené pro uložení hodnoty.
Referenční typy (reference types) -
do této skupiny patří typ String a všechny třídy.
Na rozdíl od hodnotových typů se jejich hodnota uloží do oblasti paměti nazývané halda.

Do proměnné se uloží pouze adresa paměti, kde je hodnota uložena - reference.