DÚ 3 – Skupiny 06 a 23

Termín odevzdání DÚ (prvního pokusu) je **PÁTEK 5. 12. 23:59**.

Třetí domácí úkol se bude věnovat výpočtu odmocniny. Přečtěte si zadání až do konce, i na poslední stránce se mohou skrývat užitečné informace (třeba ukázky běhu programu). Abych vás donutil procvičit si funkce a rekurzi, vaše řešení úlohy **NESMÍ OBSAHOVAT JAKÝKOLIV CYKLUS (while, for, do while) ANI PŘÍKAZ GOTO A NESMÍ VYUŽÍVAT VESTAVĚNÉ MATEMATICKÉ FUNKCE JAZYKA C ANI KNIHOVNU MATH.H**. Protože je to opravdu důležité, radši to ještě jednou zopakuji:

vaše řešení úlohy **NESMÍ OBSAHOVAT JAKÝKOLIV CYKLUS (while, for, do while) ANI PŘÍKAZ GOTO A NESMÍ VYUŽÍVAT VESTAVĚNÉ MATEMATICKÉ FUNKCE JAZYKA C ANI KNIHOVNU MATH.H**

Nyní již k samotnému problému. Program přečte od uživatele dvě čísla: jaké číslo chceme odmocňovat (**odmocněnec**) a kolikátou odmocninu chceme vypočítat (pro jednoduchost ho nazveme **odmocnitel**). Pro hodnoty platí tyto podmínky:

- Mocněnec musí být kladné reálné číslo (pro záporná čísla není odmocnina definována),
- odmocnitel kladné celé číslo (záporná odmocnina není definována a desetinné odmocniny ignorujeme),
- výsledkem je opět kladné reálné číslo.

Pro nalezení odmocniny použijeme **algoritmus binárního půlení**. Bližší informace naleznete zde:

https://is.muni.cz/auth/el/1433/podzim2012/IB111/um/cisla.pdf

Podívejte se na **slide 33** a dále, je tam i algoritmus, kterým se můžete inspirovat (nerekurzivní a pouze pro druhou odmocninu). Jeho funkci zde shrneme znova (podívejte se na obrázek ve výše uvedených materiálech):

Algoritmus pracuje tak, že „hádá“ čísla a ověřuje, jestli se jedná o hledanou odmocninu. Nehádá je ale náhodně. V každém kroku pracuje s nějakým číselným intervalem (tzn. čísly mezi určitou horní a dolní hranicí). V prvním kroku nastaví spodní hranici na **0** a horní na **odmocněnec**. Dále:

1. Určí **kandidáta** na výsledek jako průměr těchto dvou čísel – nalezne tedy střed intervalu;

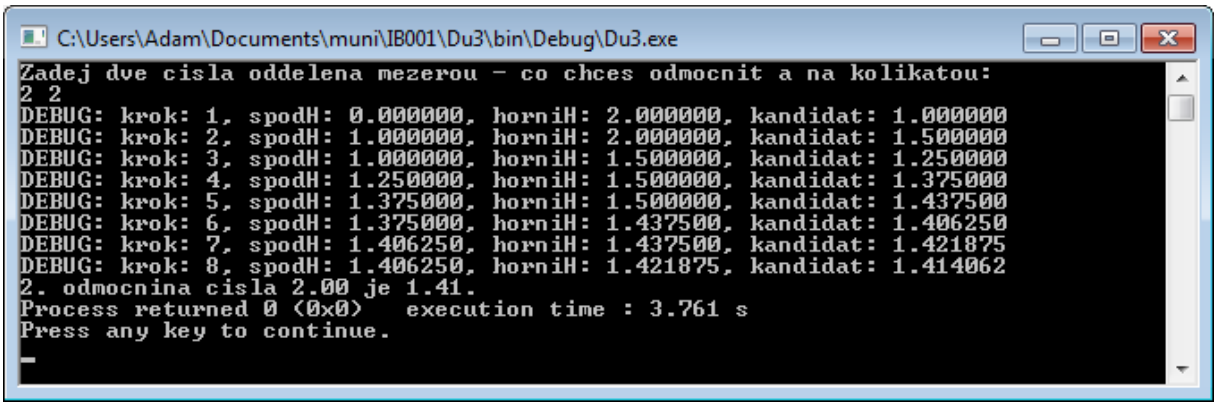
2. Provede **umocnění kandidáta** na **odmocnitele** - Hledáme např. třetí odmocninu ze dvou, takže umocníme kandidáta na třetí, abychom zjistili, jestli je kandidát vyhovující a jeho třetí mocnina se rovná dvěma – viz dále;
3. Porovná ho s původním **odmocněncem** - Vzhledem k tomu, že výpočty s desetinnými čísly v počítači nejsou zcela přesné, není možné kontrolovat přesnou shodu umocněného kandidáta a mocněnce, musíme proto postupovat následovně:
    a. Odečte od sebe **umocněného kandidáta** a **odmocněnce**
    b. Zjistí **absolutní hodnotu** tohoto rozdílu
    c. Pokud je **absolutní hodnota** menší, než nějaká programátorem stanovená hodnota (v našem případě ji nastavíme na **0.001**), prohlásí kandidáta za výsledek, jinak pokračuje dále.
4. Dále mohou nastat dvě situace, na základě kterých omezíme interval vyhledávání a spustíme celý proces od bodu 1 znovu (toto je místo pro rekurzi):
    a. **Umocněný kandidát** je větší než **odmocněnec** – V takovém případě spodní hranici necháme stejnou a horní hranici snížíme na hodnotu kandidáta (všechna vyšší čísla také budou nepoužitelná) a znovu spustíme celý vyhledávací proces;
    b. **Umocněný kandidát** je menší než **odmocněnec** – V takovém případě budou naopak všechna čísla menší než kandidát nepoužitelná, nastavíme proto spodní hranici hledání na hodnotu kandidáta a horní necháme stejnou a znova spustíme celý vyhledávací proces.

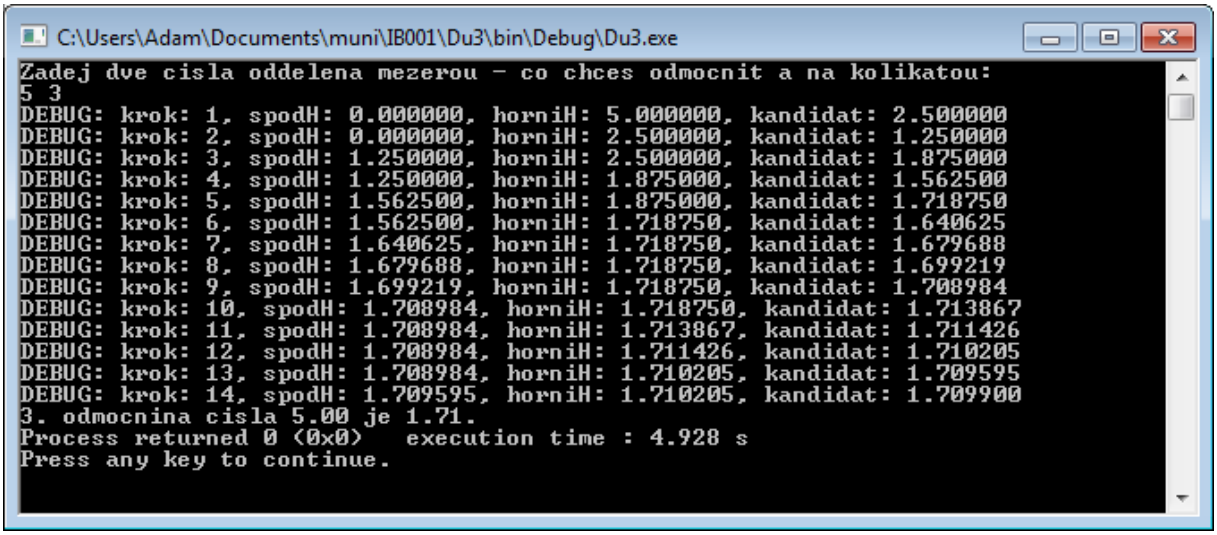
V každém kroku algoritmu se bude interval vždy o nějaký kus zmenšovat, až bude tak malý, že se „trefí“ do hledaného čísla. Po určitém počtu kroků (zhruba 10 – 20) se tedy dobere kýženého výsledku.

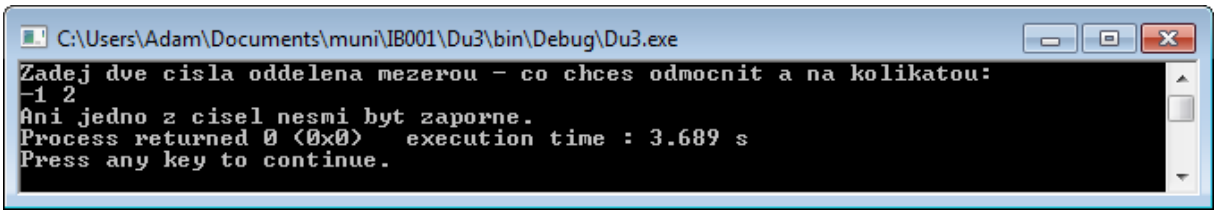
Pro vás to v domácím úkolu znamená zejména:

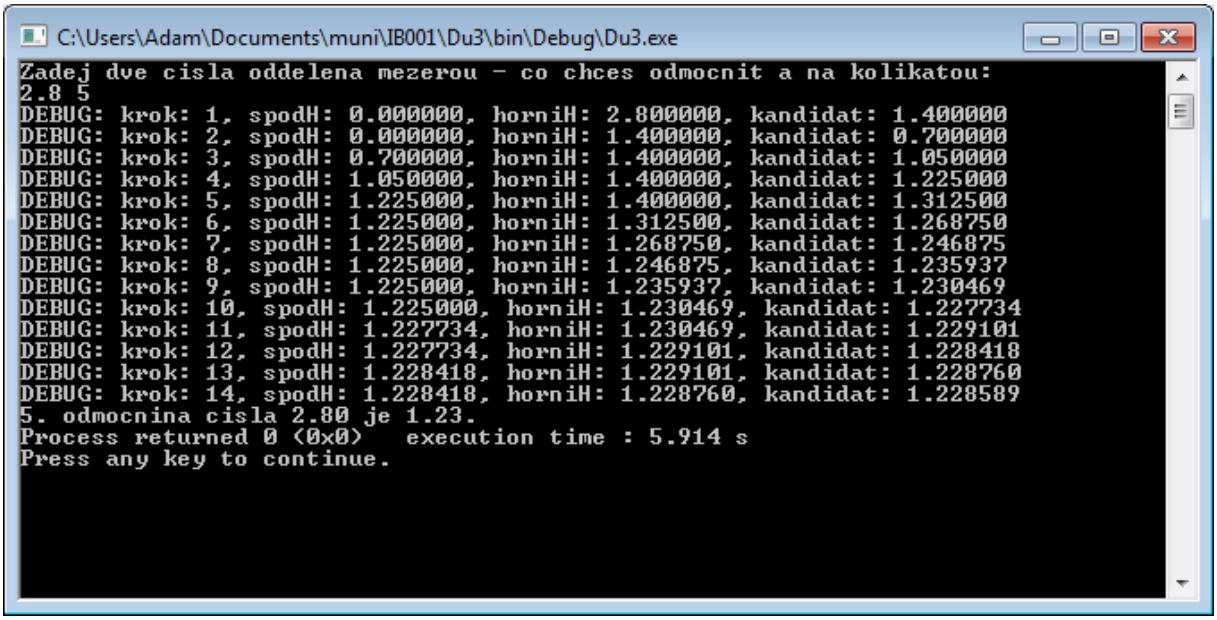
- Načíst od uživatele **odmocněnce** (kladný reálný) a **odmocnitele** (kladný celý);
- Ověřit platnost vstupu;
- Vytvořit **rekurzivní** funkci pro binární půlení podle výše popsaného postupu a zavolat ji s výchozím rozsahem intervalu (0, **odmocněnec**);
- Ošetřit správně porovnávání reálných čísel pomocí porovnání **absolutní hodnoty jejich rozdílu** a prahové hodnoty (**0.001**) – stačí nám, že si čísla jsou „dostatečně blízko“
- Vytvořit jednoduchou **rekurzivní** funkci pro umocnění čísla (Nápověda: $x^n = x * x^{n-1}$ a $x^0 = 1$) – nezapomeňte, že nesmíte používat cykly ani vestavěné funkce;
- Možná si pro přehlednost budete chtít vytvořit i funkci pro výpočet absolutní hodnoty (nezapomínejte, že existující funkce nesmíte použít). V tom případě ji ale nepojmenovávejte **abs**, protože vám název může kolidovat s tou, která je součástí jazyka C.

Program se dá pohodlně napsat na zhruba 60 řádků. Na další stránce vidíte ukázkové běhy programu včetně „ladících“ informací pro každý krok výpočtu – aktuální krok, spodní a horní hranici intervalu a aktuálního kandidáta na výsledek. Tyto výpisy váš program nemusí obsahovat. Ověřte si však, že váš program pro všechny vzorové hodnoty funguje správně. Ve **studijních materiálech našich cvičení** se podívejte na příklady s funkcemi a rekurzí, kde se počítá faktoriál a n-tý prvek Fibbonaciho posloupnosti.


C:\Users\Adam\Documents\muni\IB001\Du3\bin\Debug\Du3.exe
Zadej dve cisla oddelena mezerou - co chces odmocnit a na kolikatou:
2 2
DEBUG: krok: 1, spodH: 0.000000, horniH: 2.000000, kandidat: 1.000000
DEBUG: krok: 2, spodH: 1.000000, horniH: 2.000000, kandidat: 1.500000
DEBUG: krok: 3, spodH: 1.000000, horniH: 1.500000, kandidat: 1.250000
DEBUG: krok: 4, spodH: 1.250000, horniH: 1.500000, kandidat: 1.375000
DEBUG: krok: 5, spodH: 1.375000, horniH: 1.500000, kandidat: 1.437500
DEBUG: krok: 6, spodH: 1.375000, horniH: 1.437500, kandidat: 1.406250
DEBUG: krok: 7, spodH: 1.406250, horniH: 1.437500, kandidat: 1.421875
DEBUG: krok: 8, spodH: 1.406250, horniH: 1.421875, kandidat: 1.414062
2. odmocnina cisla 2.00 je 1.41.
Process returned 0 (0x0)   execution time : 3.761 s
Press any key to continue.


C:\Users\Adam\Documents\muni\IB001\Du3\bin\Debug\Du3.exe
Zadej dve cisla oddelena mezerou - co chces odmocnit a na kolikatou:
5 3
DEBUG: krok: 1, spodH: 0.000000, horniH: 5.000000, kandidat: 2.500000
DEBUG: krok: 2, spodH: 0.000000, horniH: 2.500000, kandidat: 1.250000
DEBUG: krok: 3, spodH: 1.250000, horniH: 2.500000, kandidat: 1.875000
DEBUG: krok: 4, spodH: 1.250000, horniH: 1.875000, kandidat: 1.562500
DEBUG: krok: 5, spodH: 1.562500, horniH: 1.875000, kandidat: 1.718750
DEBUG: krok: 6, spodH: 1.562500, horniH: 1.718750, kandidat: 1.640625
DEBUG: krok: 7, spodH: 1.640625, horniH: 1.718750, kandidat: 1.679688
DEBUG: krok: 8, spodH: 1.679688, horniH: 1.718750, kandidat: 1.699219
DEBUG: krok: 9, spodH: 1.699219, horniH: 1.718750, kandidat: 1.708984
DEBUG: krok: 10, spodH: 1.708984, horniH: 1.718750, kandidat: 1.713867
DEBUG: krok: 11, spodH: 1.708984, horniH: 1.713867, kandidat: 1.711426
DEBUG: krok: 12, spodH: 1.708984, horniH: 1.711426, kandidat: 1.710205
DEBUG: krok: 13, spodH: 1.708984, horniH: 1.710205, kandidat: 1.709595
DEBUG: krok: 14, spodH: 1.709595, horniH: 1.710205, kandidat: 1.709900
3. odmocnina cisla 5.00 je 1.71.
Process returned 0 (0x0)   execution time : 4.928 s
Press any key to continue.


C:\Users\Adam\Documents\muni\IB001\Du3\bin\Debug\Du3.exe
Zadej dve cisla oddelena mezerou - co chces odmocnit a na kolikatou:
-1 2
Ani jedno z cisel nesmi byt zaporne.
Process returned 0 (0x0)   execution time : 3.689 s
Press any key to continue.


C:\Users\Adam\Documents\muni\IB001\Du3\bin\Debug\Du3.exe
Zadej dve cisla oddelena mezerou - co chces odmocnit a na kolikatou:
2.8 5
DEBUG: krok: 1, spodH: 0.000000, horniH: 2.800000, kandidat: 1.400000
DEBUG: krok: 2, spodH: 0.000000, horniH: 1.400000, kandidat: 0.700000
DEBUG: krok: 3, spodH: 0.700000, horniH: 1.400000, kandidat: 1.050000
DEBUG: krok: 4, spodH: 1.050000, horniH: 1.400000, kandidat: 1.225000
DEBUG: krok: 5, spodH: 1.225000, horniH: 1.400000, kandidat: 1.312500
DEBUG: krok: 6, spodH: 1.225000, horniH: 1.312500, kandidat: 1.268750
DEBUG: krok: 7, spodH: 1.225000, horniH: 1.268750, kandidat: 1.246875
DEBUG: krok: 8, spodH: 1.225000, horniH: 1.246875, kandidat: 1.235937
DEBUG: krok: 9, spodH: 1.225000, horniH: 1.235937, kandidat: 1.230469
DEBUG: krok: 10, spodH: 1.225000, horniH: 1.230469, kandidat: 1.227734
DEBUG: krok: 11, spodH: 1.227734, horniH: 1.230469, kandidat: 1.229101
DEBUG: krok: 12, spodH: 1.227734, horniH: 1.229101, kandidat: 1.228418
DEBUG: krok: 13, spodH: 1.228418, horniH: 1.229101, kandidat: 1.228760
DEBUG: krok: 14, spodH: 1.228418, horniH: 1.228760, kandidat: 1.228589
5. odmocnina cisla 2.80 je 1.23.
Process returned 0 (0x0)   execution time : 5.914 s
Press any key to continue.