Příklady na rekurzi

Všechny programy naprogramujte pomocí rekurze! Na konci souboru je ke každému programu nápověda.

Program A1: faktoriál (byl na hodině)
Naprogramujte funkci fact(n), která spočte n!

Program A2: dělení buněk
Varianta 1: Buňka se každý den rozdělí na dvě. Na začátku máme jedinou buňku. Naprogramuj funkci, která zjistí počet buněk po *d* dnech.
Varianta 2: Na začátku máme *n* buněk a každá buňka se za jeden den rozdělí na *r* buněk. Kolik budeme mít buněk po *d* dnech?

Program A3: radioaktivní rozpad
Každá radioaktivní látka má tzv. poločas rozpadu, tj. dobu, za kterou se rozpadne právě polovina hmotnosti jejích atomů. Na začátku máme *m* gramů (double) radioaktivní látky a poločas rozpadu je *t* dní (integer). Zjisti, kolik látky nám zbyde po *n* dnech (integer). Nejdřív program testuj pro *n*, která jsou násobky *t*. Poté program vhodně ošetři tak, aby fungoval i pro ostatní čísla (výsledek pak stačí udělat jen přibližně).

Program A4: střádání
Střádat znamená mít účet v bance a na něj pravidelně (typicky jednou měsíčně) vkládat pořád stejnou částku. Napiš funkci, které se zadá, kolik měsíčně vkládáme, jaký máme úrok a kolik měsíců celkem střádáme a ona vypočte, kolik peněz bude na účtu při ukončení.

Program A5: aritmetická posloupnost
Aritmetická posloupnost je posloupnost čísel, kde každé následující číslo je větší o pevně zvolené číslo nazývané diference.
Např: 5, 7, 9, 11, 13, 15, … je aritmetická posloupnost, kde první člen je 5 a diference je 2.
Naprogramuj funkci clenAP(prvni, d, k), která spočte *k*-tý člen aritmetické posloupnosti s prvním členem *první* a diferencí *d*. Např. clenAP(5, 3, 4) = 11.
Dále naprogramuj funkci soucetAP(první, q, k), která sečte prvních *k* členů této posloupnosti. Při testování zadávej malá čísla, zpracování může být velmi časově náročné!

Program A6: geometrická posloupnost
Geometrická posloupnost je posloupnost čísel, kde každé následující číslo je q-krát větší než předchozí. Číslo *q* se nazývá kvocient.
Např. 2, 6, 18, 54, 162, … je geometrická posloupnost, kde první člen je 2 a kvocient je 3.
Naprogramuj funkci clenGP(prvni, q, k), která spočte *k*-tý člen geometrické posloupnosti s prvním členem *prvni* a kvocientem *q*.
Dále naprogramuj funkci soucetGP(první, q, k), která sečte prvních *k* členů této posloupnosti. Při testování zadávej malá čísla, zpracování může být velmi časově náročné!

Program A7: Fibonacciho posloupnost
Budeme zkoumat chov králíků. Na začátku máme jeden pár. Každý pár měsíčně vrhne další pár, kromě prvního měsíce života. Králíci nám neumírají a mají schopnost vrhat páry neustále. První pár na začátku vrhá další pár až během druhého měsíce, jako všechny ostatní páry. Napiš funkci, která zjistí, kolik párů králíků máme po *n* měsících.

Příklady na pole

Všechny funkce ať pracují s globálně zadaným polem cisla[].

Program B1: Součet/součin
Implementuj dvě funkce, které najdou součet/součin všech prvků pole.

Program B2: Maximum/minimum
Implementuj dvě funkce, které najdou minimum/maximum pole.

Program B3: Nalezení prvku
Implementuj funkci, která vyhledá zadaný prvek (číslo) v poli. Pokud prvek najde, vrací jeho index, jinak vrátí −1.

Program B4: Řazení
Implementuj funkci, která pomocí rekurze seřadí vzestupně/sestupně pole.

Příklad se složitějším zadáním, ale snadným řešením

Program C1: Kombinační číslo

Kombinační číslo $\binom{n}{k}$ (čteme „en nad ká“) vyjadřuje počet *k*-prvkových podmnožin *n*-prvkové množiny.

Např.

a) $\binom{4}{2} = 6$, protože když mám 4 prvkovou množinu {A, B, C, D}, tak jejích dvouprvkových podmnožiny je 6: {A, B}, {A, C}, {A, D}, {B, C}, {B, D}, {C, D}.

b) $\binom{4}{4} = 1$, protože množina {A, B, C, D} má jedinou čtyřprvkovou podmnožinu, sebe samu {A,B,C,D}.

c) $\binom{4}{0} = 1$, protože nula-prvková podmnožina je prázdná podmnožina a ta je jen jedna.

Chceme-li vypočítat kombinační číslo $\binom{6}{4}$, tj. počet 4-prvkových podmnožin 6-prvkové množiny, tak můžeme postupovat následovně. Představme si nějakou 6-prvkovou množinu, např. {A, B, C, D, E, F }. Její 4-prvkové podmnožiny můžeme rozdělit do dvou skupin – do těch, které obsahují prvek A – tedy množiny {A, *, *, *} − a do těch, které prvek A neobsahují. Počet 4-prvkových podmnožin obsahující prvek A je roven $\binom{5}{3}$, protože {A, *, *, *} doplňujeme třemi prvky a k doplnění můžeme použít jen zbylých pět písmen. Počet 4-prvkových podmnožin bez písmene A je zase roven $\binom{5}{4}$, protože vybíráme 4 prvky z 5, neboť z původních 6 máme zakázáno vybrat prvek A.
Dostáváme tedy, že

$$\binom{6}{4} = \binom{5}{3} + \binom{5}{4}.$$

Napiš funkci, která bude mít za úkol vypočítat kombinační číslo n nad k tímto způsobem.

Nápověda

Program A1: Faktoriál počítej jako fact(n) = n * fact(n – 1).

Program A2: Varianta 1: Každý den máme tolik buněk, co jsme měli den před tím krát 2.

Program A3: Po *n* dnech máme tolik látky, kolik jsme měli před *(n − t)* dny děleno dvěma.

Program A4: Např. po 5 měsících máme částku, kterou jsme měli na konci 4. měsíce, kterou jsme navíc zvýšili o vkládanou částku a zúročili.

Program A5: k-tý člen počítej jako (k−1). člen a přičtěte diferenci:

clenAP(prvni, d, k) = clenAP(první,d,k-1)

k-tý součet se spočte jako (k−1) součet plus k-tý člen

Program A6: k-tý člen počítej jako (k−1). člen a vynásob ho kvocientem
k-tý součet počítej jako (k−1). součet plus k-tý člen

Program A7: Najdi si na internetu *Fibonacciho posloupnost.*

Program B1: Naprogramuj funkci soucet(levy, pravy), která spočte součet pole od levého prvku k pravému tak, že si ho rozdělí na jeden prvek vlevo a zbytek, ze kterého rekurzí opět spočte součet. NEBO (složitější, ale efektivnější) rozděl si pole na půl a sečti ho tak, že sečteš levou půlku zvlášť, pak pravou půlku zvlášť a výsledky opět sečteš. Pozor! Ne vždy půjde pole rozdělit přesně na půl a pro optimalizaci můžeš např. udělat ručně i součet 2 prvků, nejen jednoho.

Program B2: Naprogramuj funkci maximum(levy, pravy), která najde maximum v poli od levého prvku k pravému tak, že si ho rozdělí na jeden prvek a zbytek a poté porovná maximum levé a pravé části NEBO podobně jako u programu B1 můžeš pole rozdělit na dvě stejné části.

Program B3: Funkce si rozdělí pole jeden prvek a zbytek NEBO půl na půl a v každé části hledá prvek zvlášť.

Program B4: Naprogramuj funkci sort(levy, pravy), která si pole rozdělí na jeden prvek a zbytek. Poté seřadí pomocí stejné funkce pravou část a levý prvek do ní zařadí tak, aby bylo pole srovnané.