Java és C++ gyakorlat fizikus–informatikusoknak

1. zárthelyi dolgozat

2004. november 11., csütörtök

1. A mellékelt programban megtalálja a kvaternio osztályt, amellyel az $\boldsymbol{\alpha} = \alpha_0 + \alpha_1\mathbf{i} + \alpha_2\mathbf{j} + \alpha_3\mathbf{k}$ alakú kvaterniók definiálhatók, értelmezett rájuk az összeadás (+ és += műveletek), a szorzás ill. a valós számmal való szorzás (* ill. *= alakban egyaránt), valamint a konjugálás ($\mathbf{i}$, $\mathbf{j}$, $\mathbf{k}$ együtthatójának egyidejű előjelváltása) a conjg függvénnyel. Helyvektorokat kvaterniókkal $\mathbf{r} = x\mathbf{i}+y\mathbf{j}+z\mathbf{k}$ alakban adhatunk meg. Mutassa meg numerikus példákon (véletlenszerű kvaterniókat kiválasztva), hogy az $\boldsymbol{r}' = \boldsymbol{\alpha}\boldsymbol{r}\boldsymbol{\alpha}^*$ transzformáció tetszőleges $\boldsymbol{\alpha}$ kvaternió esetében a vektor elforgatásának felel meg, amennyiben $\alpha_0^2+\alpha_1^2+\alpha_2^2+\alpha_3^2 = 1$! Mutassa meg továbbá, hogy a z tengely körüli $\varphi = 45°$ szögű elforgatás a $\cos(\varphi/2) + \sin(\varphi/2)\mathbf{k}$ kvaterniónak felel meg, az x tengely körüli $\beta = 60°$ szögű elforgatás pedig a $\cos(\beta/2) + \sin(\beta/2)\mathbf{i}$ kvaterniónak! Határozza meg ennek alapján numerikusan, a kvaterniók használatával a $\varphi = 45°$, $\beta = 60°$, $\psi = 30°$ Euler-szögekhez (φ szögű forgatás a z tengely körül, majd β szögű forgatás az x tengely körül, végül ψ szögű forgatás a z tengely körül) tartozó forgatás mátrixát!

2. Virtuális függvények alkalmazásával írjon olyan C++ programot, amely minden kétismeretlenes, két egyenletből álló lineáris egyenletrendszer esetében automatikusan megtalálja a lehetséges megoldásokat!

3. Minden négyzetes mátrix felbontható egy alsó háromszögmátrix (L) és egy felső háromszögmátrix (U) szorzatára. (Hatékonyan alkalmazható lineáris egyenletrendszerek megoldásakor.) A felbontás neve LU dekompozíció, melynek egy C nyelvű, float típusra megírt változata a Numerical Recipes programkönyvtárban (a gépen a *numrec* directory-ban) `ludcmp.c` néven található meg, a hozzátartozó dokumentáció a `doc/c2-3.ps` file.

   C++ programjában sablonok alkalmazásával általánosítsa az LU dekompozíciós algoritmust úgy, hogy float, double és std::complex<double> típusú adatok esetén egyaránt működjön!