Cvičenie č. 11

Zadanie vychádza zo zadania predchádzajúceho cvičenia. Modifikácia spočíva v zmene uloženia záznamov. Miesto poľa sa vyžaduje použitie DYNAMICKÉHO SPÁJANÉHO ZOZNAMU. Zadanie zopakujeme, ale zmeníme niektoré formulácie.

Úvod - Štruktúry: Uvažujte záznamy o študentoch, kde sa uchovávajú položky: priezvisko+meno (jedna položka), aktuálny ročník a priemer známok v aktuálnom ročníku. Nepredpokladajte prácu s reťazcami dlhšími ako 100 znakov. Štruktúra musí byť schopná tvoriť jednosmerný spájaný zoznam.

```
typedef struct student {
    char priezvisko_meno[100];
    int rocnik;
    float priemer;
    struct student *dalsi;
} STUDENT;
```

Úloha prvá. Štruktúra v poli – načítanie.

Napíšte funkciu `nacitajSpajany`, ktorá otvorí súbor daný v parametri (v main() funkciu volajte tak, že súbor sa bude volať „vstup.txt“). Prvý riadok obsahuje počet študentov. Funkcia potom dynamicky alokuje záznamy jednotlivých študentov, číta dáta ZO SUBORU a prepája záznamu do spájaného zoznamu.. Každý údaj v súbore bude na samostatnom riadku. Predpokladajte správne formátovanie súboru. V tejto, ani žiadnej funkcii v zadaní nepoužívajte globálne premenné. Funkcia vracia začiatok spájaného zoznamu. **Počet** Vám netreba!

Povinný prototyp funkcie:

```
STUDENT *nacitajSpajany(char *subor)
```

Príklad súboru:

```
3
Jozko Mrkvicka
1
2.3
Ferko Maly
2
1.3
Anicka Svarna
3
1.2
```

```
Čiže pre každého študenta sú vždy 3 riadky:
Priezvisko a Meno (fgets – nie fscanf, lebo ten Vám nenačíta medzeru)
Rocnik
Priemer
```

Úloha druhá. Štruktúra v spájanom zozname – výpis.

Napíšte funkciu `vypisSpajany`, ktorá vypíše všetky záznamy, každý záznam v jednom riadku, pričom položky sú v riadku oddelené vždy znakom bodkočiarka ";".

Povinný prototyp funkcie:

```
void vypis (STUDENT *studenti)
```

Príklad výpisu:

```
Jozko Mrkvicka;1;2.3
Ferko Maly;2;1.3
Anicka Svarna;3;1.2
```

Úloha tretia. Štruktúra v spájanom zozname – analýza.

Napíšte funkciu `najlepsiSpajany`, ktorá vráti smerník na najlepšieho študenta (najlepší priemer známok) v ročníku (teda nie zo všetkých v zozname, len z ročníka). Ak pre daný ročník nie je študent, funkcia vráti NULL. Pomôcka: Stále platí, že počet nepotrebujete. Viete kedy je koniec zoznamu!

Povinný prototyp funkcie:

```
STUDENT *najlepsiSpajany (STUDENT *studenti, int rocnik)
```

Úloha štvrtá. Štruktúra v spájanom zozname – uvolniť pamäť

Napíšte funkciu ktorá uvoľní pamäť pre celý spájaný zoznam. Uvedomte si, že tak ako ste alokovali osoby po jednej, tak ich po jednej musíte aj uvoľňovať.

Povinný prototyp funkcie:

```
void uvolniSpajany (STUDENT *studenti)
```

Funkcie vierohodne demonštrujte v MAIN:

```
STUDENT *spajany, *naj;
spajany = nacitajSpajany("vstup.txt");
if (spajany == NULL) {
  printf("nepodarilo sa nacitat a naplnit zoznam");
  return -1;
}
vypisSpajany(spajany);
naj = najlepsiSpajany(spajany, 3);
if (naj == NULL) {
  printf("v rocniku nie je student");
  return -2;
}
printf("najlepsi student je %s", naj->priezvisko_meno);
uvolniSpajany(spajany);
return 0;
```