Kezdés ideje	2020. május 14., csütörtök, 13:17
Állapot	Befejezte
Befejezés dátuma	2020. május 14., csütörtök, 14:11
Felhasznált idő	54 perc 22 mp
Pont	**5** a maximum 5 közül (**100**%)

1 kérdés

Helyes

1 közül 1 leosztályozva

Egy hangszerbolt gitárjait egyirányú láncolt listában szeretnénk tárolni. Minden gitárról ismert a húrjainak száma, ami 6, 7, 8 vagy 12 lehet. Tudjuk az árát, ami legalább 200.000 forint, de 5.000.000 forintnál alacsonyabb. Illetve tároljuk a hangszedője típusát, ami vagy "humbucker" vagy "single coil".

Készítsen general nevű függvényt, ami létrehoz egy gitárokból álló láncolt listát. A listán található gitárok száma legalább nulla és maximum 10. A maximális darabszámot, a minimális árat és a maximális árat a következő szimbolikus állandókkal adja meg: MAX_DB, MIN_AR és MAX_AR. Véletlenszerűen döntse el, hogy hány darab gitár kerül a listára és a gitárok minden paramétere (húrok száma, ár, hangszedő típusa) szintén véletlenszerűen kerüljön meghatározásra, a fenti szabályok figyelembe vételével. A véletlenszám generálás inicializálásáról nem kell gondoskodnia. A függvény visszatérési értéke a generált listában az első gitár címe. A függvénynek bemeneti paraméterei nincsenek.

A lista egy elemének a szerkezetét a következő struktúra adja meg (ennek létrehozásáról nem kell gondoskodnia):

```
struct gitar {
  int hurok_szama;
  int ar;
  string hangszedo;
  gitar* kov;
};
```

Answer: (penalty regime: 0 %)

Reset answer

```
#define MAX_DB 10
#define MIN_AR 200000
#define MAX_AR 5000000

gitar* general() {
    gitar* horgony = NULL;
    int db = rand()%(MAX_DB+1);
    for(int i=0; i<db; i++){
        gitar* uj = new gitar;
        int a = rand()%4;
        switch(a){
            case 0 : uj->hurok_szama = 6; break;
            case 1 : uj->hurok_szama = 7; break;
            case 2 : uj->hurok_szama = 8; break;
            case 3 : uj->hurok_szama = 12; break;
        }
        uj->ar = rand()%(MAX_AR - MIN_AR) + MIN_AR;
        int h=rand()%2;
        if(h==0){
            uj->hangszedo = "humbucker";
        }else{
            uj->hangszedo = "single coil";
        }
        horgony = uj;
```

	Test	Expected	Got	
✔	general_teszt();	OK	OK	✔

Passed all tests! ✔

Question author's solution:

```
#define MAX_DB 10
#define MIN_AR 200000
#define MAX_AR 5000000

gitar* general() {
  gitar* lista = NULL;
  int hur[] = {6, 7, 8, 12};
  string hangszedok[] = {"humbucker", "single coil"};
  int db = rand()%(MAX_DB + 1);
  for(int i = 0; i < db; i++) {
    gitar* uj = new gitar;
    uj->hurok_szama = hur[rand()%4];
    uj->ar = rand()%(MAX_AR-MIN_AR) + MIN_AR;
    uj->hangszedo = hangszedok[rand()%2];
    uj->kov = lista;
    lista = uj;
  }
  return lista;
}
```

Helyes

Leadásra bejelöl: 1/1.

2 kérdés

Helyes

1 közül 1 leosztályozva

Készítsen hozzafuz nevű függvényt, ami a paraméterül kapott gitárokból álló egyirányú láncoltlista végére egy új gitárt fűz hozzá. A függvény bemenetként 4 paramétert vár: a gitárok listáját (a láncolt listában az első gitár címe) és az új gitár paramétereit (húrok száma, ár, hangszedő típusa). A függvény visszatérési értéke a láncolt lista legyen (a lista első gitárjának a címe).

A lista egy elemének a szerkezetét a következő struktúra adja meg (ennek létrehozásáról nem kell gondoskodnia):

```
struct gitar {
  int hurok_szama;
  int ar;
  string hangszedo;
  gitar* kov;
};
```

Answer: (penalty regime: 0 %)

Reset answer

```
gitar* hozzafuz(gitar* lista, int hurok_szama, int ar, string hangszedo) {
    gitar*horgony=lista;
    gitar* uj = new gitar;
  uj->hurok_szama = hurok_szama;
  uj->ar=ar;
  uj->hangszedo=hangszedo;
  uj->kov = NULL;
  if(horgony == NULL){
    return uj;
  }else{
    gitar* akt = horgony;
    while(akt->kov != NULL){
    akt = akt->kov;
    }
    akt->kov = uj;
    return horgony;
  }
}
```

	Test	Expected	Got	
✔	`hozzafuz_teszt();`	OK	OK	✔

Passed all tests! ✔

Question author's solution:

```
gitar* hozzafuz(gitar* lista, int hurok_szama, int ar, string hangszedo) {
  gitar* uj = new gitar;
  uj->hurok_szama = hurok_szama;
  uj->ar = ar;
  uj->hangszedo = hangszedo;
  uj->kov = NULL;
  if(lista == NULL) {
    lista = uj;
  } else {
    gitar* akt = lista;
    while(akt->kov != NULL) {
      akt = akt->kov;
    }
    akt->kov = uj;
  }
  return lista;
}
```

Helyes

Leadásra bejelöl: 1/1.

3 kérdés

Helyes

1 közül 1 leosztályozva

Készítsen szures nevű függvényt, ami gitárokat tartalmazó egyirányú láncolt listát vár paraméterül. A függvény készítsen egy új listát, amelyre a 6 húros, "single coil" típusú hangszedővel, valamint a 8 húros, "humbucker" hangszedővel rendelkező gitárok kerülnek fel. A szűrt listára a hangszerek olyan sorrendben kerüljenek, amilyen sorrendben egymás követik az eredeti listában. A függvény visszatérési értéke a szűrt lista legyen (a lista első gitárjának a címe). Ha nincs ilyen gitár, akkor NULL értékkel térjen vissza.

A lista egy elemének a szerkezetét a következő struktúra adja meg (ennek létrehozásáról nem kell gondoskodnia):

```
struct gitar {
  int hurok_szama;
  int ar;
  string hangszedo;
  gitar* kov;
};
```

Answer: (penalty regime: 0 %)

Reset answer

```
gitar* szures(gitar* lista) {
    gitar* szurtHorgony = NULL;
    gitar* vege=NULL;
    gitar* akt = lista;
    while(akt!=NULL){
        if((akt->hurok_szama==6 and akt->hangszedo=="single coil") or (akt->hurok_szama==8
            gitar* uj = new gitar;
            uj->hurok_szama = akt->hurok_szama;
            uj->ar = akt->ar;
            uj->hangszedo= akt->hangszedo;
            uj->kov = NULL;
            if(szurtHorgony == NULL){
                szurtHorgony=uj;
                vege=uj;
            }else{
                vege->kov=uj;
                vege=vege->kov;
            }
        }
        akt=akt->kov;
    }
    return szurtHorgony;
}
```

	Test	Expected	Got	
✔	szures_teszt();	OK	OK	✔

Passed all tests! ✔

Question author's solution:

```
gitar* szures(gitar* lista) {
  gitar* szurt = NULL;
  gitar* szurtUtolso = NULL;
  while(lista != NULL) {
    if((lista->hurok_szama == 6 and lista->hangszedo == "single coil") or (lista->hurok_szama == 8 and lista-
>hangszedo == "humbucker")) {
      gitar* uj = new gitar;
      uj->ar = lista->ar;
      uj->hurok_szama = lista->hurok_szama;
      uj->hangszedo = lista->hangszedo;
      uj->kov = NULL;
      if(szurt == NULL) {
        szurt = uj;
      } else {
        szurtUtolso->kov = uj;
      }
      szurtUtolso = uj;
    }
    lista = lista->kov;
  }
  return szurt;
}
```

Helyes

Leadásra bejelöl: 1/1.

4 kérdés

Helyes

1 közül 1 leosztályozva

Készítsen szures nevű függvényt, ami paraméterül vár egy gitárokat tartalmazó egyirányú láncolt listát. A függvény készítsen egy új listát a drága gitárokról. Egy gitárt akkor nevezünk drágának, ha az ára magasabb, mint a listán szereplő gitárok árának az átlaga. A szűrt lista legyen a függvény visszatérési értéke (a lista első gitárjának a címe). Ha nincs ilyen gitár, akkor NULL értékkel térjen vissza.

A lista egy elemének a szerkezetét a következő struktúra adja meg (ennek létrehozásáról nem kell gondoskodnia):

```
struct gitar {
  int hurok_szama;
  int ar;
  string hangszedo;
  gitar* kov;
};
```

Answer: (penalty regime: 0 %)

Reset answer

```
1  gitar* szures(gitar* lista) {
2      gitar* akt = lista;
3      int db=0;
4      double atlagAr=0;
5      while(akt!=NULL){
6          db++;
7          atlagAr += akt->ar;
8          akt=akt->kov;
9      }
10     atlagAr /= db;
11
12     akt=lista;
13     gitar* szurtHorgony = NULL;
14     gitar* vege=NULL;
15     while(akt!=NULL){
16         if(akt->ar > atlagAr){
17             gitar* uj = new gitar;
18             uj->hurok_szama = akt->hurok_szama;
19             uj->ar = akt->ar;
20             uj->hangszedo= akt->hangszedo;
21             uj->kov = NULL;
22             if(szurtHorgony == NULL){
23                 szurtHorgony=uj;
24                 vege=uj;
```

	Test	Expected	Got	
✔	szures_teszt();	OK	OK	✔

Passed all tests! ✔

Question author's solution:

```
gitar* szures(gitar* lista) {
  gitar* szurt = NULL;
  gitar* szurtUtolso = NULL;
  double atlag = 0.0;
  int db = 0;
  gitar* akt = lista;
  while(akt != NULL) {
    db++;
    atlag += akt->ar;
    akt = akt->kov;
  }
  atlag /= db;
  while(lista != NULL) {
    if(lista->ar > atlag) {
      gitar* uj = new gitar;
      uj->ar = lista->ar;
      uj->hurok_szama = lista->hurok_szama;
      uj->hangszedo = lista->hangszedo;
      uj->kov = NULL;
      if(szurt == NULL) {
        szurt = uj;
      } else {
        szurtUtolso->kov = uj;
      }
      szurtUtolso = uj;
    }
    lista = lista->kov;
  }
  return szurt;
}
```

Helyes

Leadásra bejelöl: 1/1.

5 kérdés

Helyes

1 közül 1 leosztályozva

Készítsen szures nevű függvényt, ami paraméterül vár egy gitárokat tartalmazó egyirányú láncolt listát. A függvény visszatérési értéke egy 1 elemű lista, ami a legolcsóbb 12 húros gitárt tartalmazza. Ha több gitár is van a listán, ami a legolcsóbbnak minősül, akkor a bemeneti lista sorrendjében az első találatot kell figyelembe venni. Ha nincs ilyen gitár, akkor NULL értékkel térjen vissza.

A lista egy elemének a szerkezetét a következő struktúra adja meg (ennek létrehozásáról nem kell gondoskodnia):

```
struct gitar {
  int hurok_szama;
  int ar;
  string hangszedo;
  gitar* kov;
};
```

Answer: (penalty regime: 0 %)

Reset answer

```
    int olcso12=5000001; // ez a max ar
    while(akt!=NULL){
        if(akt->hurok_szama==12 and akt->ar < olcso12){
            olcso12 = akt->ar;
        }
        akt=akt->kov;
    }

    akt = lista;
    gitar* olcsoHorgony = NULL;
    while(akt!=NULL){
        if(akt->hurok_szama==12 and akt->ar==olcso12){
            gitar* uj = new gitar;
            uj->hurok_szama = akt->hurok_szama;
            uj->ar = akt->ar;
            uj->hangszedo = akt->hangszedo;
            uj->kov = NULL;
            olcsoHorgony=uj;
            break;
        }
        akt=akt->kov;
    }
    return olcsoHorgony;
}
```

	Test	Expected	Got	
✔	szures_teszt();	OK	OK	✔

Passed all tests! ✔

Question author's solution:

```
gitar* szures(gitar* lista) {
  int min_ar = MAX_AR + 1;
  gitar* akt = lista;
  gitar* legolcsobb = NULL;
  while(akt != NULL) {
    if(akt->hurok_szama == 12 and akt->ar < min_ar) {
      min_ar = akt->ar;
      legolcsobb = new gitar;
      legolcsobb->ar = akt->ar;
      legolcsobb->hurok_szama = akt->hurok_szama;
      legolcsobb->hangszedo = akt->hangszedo;
      legolcsobb->kov = NULL;
    }
    akt = akt->kov;
  }
  return legolcsobb;
}
```

Helyes

Leadásra bejelöl: 1/1.

◄ Mintavizsga

Ugrás...

Előadás fóliák ►