MŰEGYETEM 1782

A PROGRAMOZÁS ALAPJAI 2.

HÁZI FELADAT DOKUMENTÁCIÓ

TELEFON

KÉSZÍTETTE: PAPP ISTVÁN
34gles96@gmail.com

2022. 05. 17.

TARTALOMJEGYZÉK

Felhasználói dokumentáció

A programom objetum orientalt perspektivabol ad betekintest a telefonalas, mint kapcsolat modellezesebe.

A program txt kiterjesztesű fájlból olvassa be a meglevo elofizetoket, mely adott szolgaltatonal levo keszulek vasarlasara kepesek. Az elvárt kimenet ezen keszulekek(amiket az elofizetok vasaroltak) random hivasgenerálása kovetkezteben generalt állapotjelzok. Peldaul h egy készulek kapcsolhato e masik hivasra, vagy mennyi egyenleg van a keszuleken.

Osztályok statikus leírása

Person osztaly

Felelőssége

Ez egy fifo stiluso adatszerkezet az elofizetok tarolasara.

Ősosztályok

Serializacio absztrakt ososztalyabol oroklodik, mivel annak interfeszet kapja meg, melyet egy harmadik osztaly, a Load hasznal.

Attribútumok

Privát

int* age; dinamikus adattag, elofizeto kora

string* name; dinamikus adattag, elofizeto neve

int counter dinamikus adattag az adatok szamanak fentartasara

int aktualis; ezt egy masik osztaly fogja majd hasznalni..

Publikus

int ageAt(int) const;

string nameAt(int) const;

void deserializacio(ifstream& is) override;

void serializacio_(ofstream& os) override;

Person();

int getcounter() const;

void addPerson(string nameparam, int ageparam); -elofizeto hozzaadasa

friend ostream& operator<<(ostream& os, const Person& other); -diagnosztikai celu kiirato operator

~Person();-destruktor dinamikus adattagoknak

Person(const Person& theOther); // masolo konstruktor

stringstream operator[](int a)const; //adott a indexu adat stringstream objektum visszateressel az indexhez tartozo adatok szerint

void inreaseAct();

int getAct() const;

Keszulek osztaly

Keszulek osztalya , mely a callinterfesz osztalybol szaramzik le, megorokolve annak interfeszet, hivasra valo képességet, melyet a Kapcsolat osztaly 'kasznal ki'

Metódusok

privat:

double egyenleg; //'feltolto kartyas telefonok'

bool avaible;

double beszeltemdij; // operator+ impelementalasahoz, egyenleg ujraertekelesehez

const MyLibSzolg::Szolgaltato* szolgaltatom; //aggragacio

public:

Keszulek();

string getAvaible() const;

double getEgyenleg() const;

friend ostream& operator<<(ostream& os, const Keszulek& ohter) ; //elerhetosegre, hivasra valo állapot diagnosztiai celu kiiratasa

```
void egyenleg_init();
void setAvaible();
bool operator+(Keszulek& other) override;
bool operator-(Keszulek& other) override;
Keszulek(const MyLibSzolg::Szolgaltato& param);
void decreaseEgyenleg(double);
```

Kapcsolat osztaly

callinterfesz absztrakt osztalybol szarmazik le interfesz orokles miatt. Két telefonalasra képes keszuelekt kapcsol ossze operatortulterhelessel.

privat :

```
Keszulek* k1;
Keszulek* k2;
callInterface* p1;
callInterface* p2;
```

public:

```
Kapcsolat(vector<callInterface&> pitem);   //vector adatstrukturabol veletlenszeruen inicializalja a tagvaltozokat, mely a callinterface osztaly modositja-> oda vissza kommunikacio

void call();          //randomizalt hivasgeneralas a vector adattagokbol, hivas dijaval ter vissza,
```

Összegzés

Mit tanultál a megvalósítás során?

objektumorientalt szemleletet, generikus osztalyok és virtualis tagfugvenyek interfesz jellegu hasznalata orokles soran absztrakt ososztalyon keresztul

Továbbfejlesztési lehetőségek

Hogyan lehetne továbbfejleszteni az alkalmazást? Lehetne esetleg bővíteni a célközönséget, alkalmazási területet?

Megirt interfeszek kiterjesztese mas jellegu keszulekekre, keszulek osztalyban nem csak a feltoltokartya tulajdonsag implementalasa

Képernyőképek a futó alkalmazásról