Ödev-04

1. Ödev-04

1.1. Object Pool

Aşağıdaki örnek bir oyun içersindeki bir dükkan ile ilgilidir.

Burada dükkan Singleton sınıf olarak oluşturulnuştur ve Object Pool görevi görmektedir. Esya için de ayrı bir sınıf oluşturulmuştur. Dukkan sınıfı esya sınıfını uretir ve barındırır. Dukkan sınıfı en fazla 3 esya barındırabilir.

Örnekte threadler ile dükkandan eşya alınıp dükkana eşya satılmaktadır.

```
std::mutex
```

komutu ile herhangi bir sıkıntı çıkmadan bu işlem gerçekleştirilebilmektedir.

```
#include <iostream>
#include <string>
#include <list>
#include <future>
#include <mutex>
#ifdef _WIN32
#include <Windows.h>
#else
#include <unistd.h>
#endif

static std::mutex _mtx;

class Esya                                                    ❶
{
    std::string isim;

public:
    Esya() : isim("")
    {

    }
```

```
    Esya(const std::string &isim) : isim(isim)
    {

    }

    std::string getIsim()
    {
        return isim;
    }

    void setIsim(std::string name)
    {
        isim = name;
    }

    void reset()
    {
        isim = "";
    }
};

class Dukkan
{
private:
    std::list<Esya*> esyalar;

    static Dukkan* instance;
    static int eleman_sayisi;
    static int mevcut_eleman_sayisi;
    int max_size = 3;
    Dukkan() {}

public:

    static Dukkan* getInstance()
    {
        _mtx.lock();
        if (instance == 0)
        {
            instance = new Dukkan;
            eleman_sayisi = 0;
        }
        _mtx.unlock();
        return instance;
```

❷ ❸ ❹

```
    }

    Esya* esyaSat(std::string isim)                                    ❺
    {
        _mtx.lock();
        if (esyalar.empty())
        {

            if(eleman_sayisi < 3) {
                std::cout << "Yeni esya yaratildi." << std::endl;
                eleman_sayisi++;
                std::cout << "Kalan esya sayisi: " <<
mevcut_eleman_sayisi << std::endl;
                _mtx.unlock();
                return new Esya(isim);
            }
            else{
                std::cout << "Dukkanda esya bulunmamaktadir." <<
std::endl;
                _mtx.unlock();
                return nullptr;
            }
        }
        else
        {
            std::cout << "Dukkandan esya alindi." << std::endl;
            Esya* esya = esyalar.front();
            esya->setIsim(isim);
            esyalar.pop_front();
            mevcut_eleman_sayisi--;
            std::cout << "Kalan esya sayisi: " << mevcut_eleman_sayisi
<< std::endl;
            _mtx.unlock();
            return esya;
        }
    }

    void esyaSatinAl(Esya* esya)                                       ❻
    {
        _mtx.lock();
        if(esya == nullptr)
        {
            std::cout << "Dukkana nullptr eklenmedi." << std::endl;
        }
        else{
```

```
            if(mevcut_eleman_sayisi < max_size)
            {
                std::cout << "Esya dukkana eklendi" << std::endl;
                esya->reset();
                esyalar.push_back(esya);
                mevcut_eleman_sayisi++;
                std::cout << "Kalan esya sayisi: " <<
 mevcut_eleman_sayisi << std::endl;
            }
            else
            {
                std::cout << "Dukkan dolu." << std::endl;
            }
        }
        _mtx.unlock();
    }
};

Dukkan* Dukkan::instance = 0;
int Dukkan::eleman_sayisi = 0;
int Dukkan::mevcut_eleman_sayisi = 0;

void esyaYazdir(std::string name){
    Dukkan* dukkan = Dukkan::getInstance();
    Esya* esy;
    esy = dukkan->esyaSat(name);
    usleep(500000);
    while(esy == nullptr)
    {
        esy = dukkan->esyaSat(name);
    }
    _mtx.lock();
    std::cout << "Esya ismi: " << esy->getIsim() << std::endl;
    _mtx.unlock();
    dukkan->esyaSatinAl(esy);
}

int main()
{
    Dukkan* dukkan = Dukkan::getInstance();
    Esya* bir;
    Esya* iki;
    Esya* uc;

    bir = dukkan->esyaSat("");
```

❼

```
    iki = dukkan->esyaSat("");
    uc = dukkan->esyaSat("");

    dukkan->esyaSatinAl(bir);                                    ❽
    dukkan->esyaSatinAl(iki);
    dukkan->esyaSatinAl(uc);
    std::cout << std::endl;

    auto thread1 = std::async(std::launch::async,esyaYazdir,"1.
 thread");
    auto thread2 = std::async(std::launch::async,esyaYazdir,"2.
 thread");
    auto thread3 = std::async(std::launch::async,esyaYazdir,"3.
 thread");
    auto thread4 = std::async(std::launch::async,esyaYazdir,"4.
 thread");
    auto thread5 = std::async(std::launch::async,esyaYazdir,"5.
 thread");
    auto thread6 = std::async(std::launch::async,esyaYazdir,"6.
 thread");

    return 0;
}
```

❶ Esya sinifi tanimlanir.

❷ Esya sınıfının constructer'ı overload edilir.

❸ Dukkan sınıfı tanımlanır.

❹ Dukkan sınıfının Singleton sınıf olması için gerekli fonksiyon tanımlanır (koşullu constructer denebilir).

❺ Object Pool olarak görev gören Dukkan sınıfı ile Esya örneği(instance) oluşturmak veya oluşturulmuş örneği dışarı vermek için kullanılır. Aynı zaman Object Pool içerisinde örnek olup olmadığını kontrol eder.

❻ Object Pool olarak görev gören Dukkan sınıfına Esya örneği eklemek için kullanılır.

❼ esyaSat() fonksiyonunun kullanımı.

❽ esyaSatınAl() fonksiyonunun kullanımı.

UML Diagramı