NYP Uygulama 5

Factory

```
public class DatabaseFactory {
    public static Database getDatabase(String type) {
        if (type.equalsIgnoreCase("MySQL")) {
            return new MySQLDatabase();
        } else if (type.equalsIgnoreCase("PostgreSQL")) {
            return new PostgreSQLDatabase();
        } else {
            return null;
        }
    }
}
```

Factory

```
public class KarakterFabrikasi {
    public static Karakter karakterOlustur(String tur) {
        if (tur.equalsIgnoreCase("Savasci")) {
            return new Savasci();
        } else if (tur.equalsIgnoreCase("Buyucu")) {
            return new Buyucu();
        }
        return null;
    }
}
```

Factory

- Java dilinde, kullanıcı nesnelerinin doğrudan oluşturulmasını engellemek amacıyla private constructor kullanan bir 'Kullanici' sınıfı tasarlayınız. Bu sınıf, kullanıcı ismi ve rol bilgisini saklamalı ve bilgileri ekrana yazdıran bir metoda sahip olmalıdır.
- Nesne üretimini kontrol altına almak için 'KullaniciFabrikasi' adında iç statik bir sınıf tanımlayarak, 'Yönetici' ve 'Standart' kullanıcılar için ayrı üretim metotları oluşturunuz. Ayrıca, bu yapının kullanımını örnekleyen bir ana sınıf (Main) yazınız.

```
public class Main {
    public static void main(String[] args) {
        // Kullanıcıları fabrika metodları ile oluşturuyoruz
        Kullanici yonetici = Kullanici.KullaniciFabrikasi.yoneticiOlustur("Ahmet");
        Kullanici standartKullanici = Kullanici.KullaniciFabrikasi.standartKullaniciOlustur("Mehmet");

        // Kullanıcı bilgilerini gösterme
        yonetici.kullaniciBilgileriniGoster();
        standartKullanici.kullaniciBilgileriniGoster();
    }
}
```

```
class Kullanici {
    private String isim;
    private String rol;

    // Constructor private: Nesne doğrudan oluşturulamaz!
    private Kullanici(String isim, String rol) {
        this.isim = isim;
        this.rol = rol;
    }

    public void kullaniciBilgileriniGoster() {
        System.out.println(isim + " ==> Kullanıcı Türü: " + rol );
    }

    // Üretici metodları içeren sınıf
    public static class KullaniciFabrikasi {
        public static Kullanici yoneticiOlustur(String isim) {
            return new Kullanici(isim, "Yönetici");
        }

        public static Kullanici standartKullaniciOlustur(String isim) {
            return new Kullanici(isim, "Standart");
        }
    }
}
```

C++

- Final yerine constexpr ya da const ifadesi kullanılabilir.
- std:: → standard kütüphane
- cout << → ekrana çıtkı
- <<endl → satır sonu

```cpp
#include <iostream>
using namespace std;

class Araba {
private:
    string marka;
    int hiz;

public:
    // Yapıcı (Constructor)
    Araba(string m, int h) {
        this->marka = m;
        this->hiz = h;
    }
    void BilgiGoster() {
        cout << "Marka: " << marka << ", Hiz: " << hiz << " km/h" << endl;
    }

};
```

```cpp
int main() {
    Araba araba1("Toyota", 120);
    araba1.BilgiGoster();
    return 0;
}
```

```cpp
int main() {
    std::cout << "30 C = " << Donusturucu::santigrattanFahrenheita(30) << "F" << std::endl;
    std::cout << "20 km = " << Donusturucu::kmdenMile(20) << " mil" << std::endl;
    std::cout << "r=2 => çevre= " << Donusturucu::yariCaptanCevre(2) << std::endl;

    std::string text = "Merhaba C++";
    std::cout << Donusturucu::kucukHarftenBuyukHarfe(text) << std::endl;

    return 0;
}
```

Donusturucu sınıfını yazın.

F = (C * 9 / 5) + 32;
mil = km * 0.621371

30 C = 86.0 F

20 km = 12.42742 mil

r=2 => çevre= 12.56

Merhaba C++

```cpp
#include <iostream>
#include <string>
class Donusturucu {
private:
    static constexpr double PI = 3.14;
public:
    static double santigrattanFahrenheita(double celsius) {
        return (celsius * 9.0 / 5.0) + 32.0;    }
    static double kmdenMile(double km) {
        return km * 0.621371;    }
    static double yariCaptanCevre(double r) {
        return 2 * PI * r;    }
    static std::string kucukHarftenBuyukHarfe(const std::string& text) {
        std::string result;
        for (char ch : text) {
            result += std::toupper(ch);
        }
        return result;
    }
};
```

C++ statik değişkenler

- Tüm nesnelerin eriştiği statik değişkenler sınıf dışında tanımlanır.

```
int main() {
    BankaHesabi hesap1("Ali", 1000);
    BankaHesabi hesap2("Ayse", 500);

    std::cout << "Hesap Sayisi: " << BankaHesabi::getHesapSayisi() << std::endl;
    std::cout << "Toplam Para: " << BankaHesabi::getToplamPara() << std::endl;

    return 0;
}
```

SORU:

Üstte verilen kodun yandaki çıktığı sağlayacağı şekilde BankaHesabi sınıfını kodlayın.

Çıktı:

2

1500.0

İPUCU:

```
// Statik değişkenlerin sınıf dışında tanımlanması gerekir
int BankaHesabi::hesapSayisi = 0;
double BankaHesabi::toplamPara = 0;
```

```cpp
#include <iostream>
#include <string>

class BankaHesabi {
private:
    std::string isim;
    double bakiye;
    static int hesapSayisi;
    static double toplamPara;

public:
    BankaHesabi(std::string isim, double bakiye) {
        this->isim = isim;
        this->bakiye = bakiye;
        hesapSayisi++;
        toplamPara += bakiye;
    }

    static int getHesapSayisi() {
        return hesapSayisi;
    }

    static double getToplamPara() {
        return toplamPara;
    }
};

// Statik değişkenlerin sınıf dışında tanımlanması gerekir
int BankaHesabi::hesapSayisi = 0;
double BankaHesabi::toplamPara = 0;
```