C++ 3.

İşleçlerin Aşırı Yüklenmesi

Şevket Umut ÇAKIR

PAÜ

28 Nisan 2020

Anahat

1. İşleçlerin Aşırı Yüklenmesi(Operator Overloading)
2. Deneyin Uygulanması

İşleçlerin Aşırı Yüklenmesi

- C++ fonksiyonlar gibi işleçlerin aşırı yüklemesine imkan tanımaktadır.
- Sadece dil içindeki işleçler aşırı yüklenebilir.
- İşlecin aşırı yüklenmesi ile bu işleçler kullanıcı tanımlı tipler için kullanılabilir.
- Örneğin bir `Elma` nesnesi ile `Armut` nesnesini toplayabilirsiniz.
- İşlecin aşırı yüklenmesi için fonksiyon tanımında `operator` anahtar kelimesi ve ilgili işleç kullanılır.

Dizi Sınıfı

Sınıf Tanımı

```
class Dizi {
    public:
        Dizi(unsigned int boyut); //Constructor
        ~Dizi(); //Destructor
        void yazdir(); //Yazdırma metodu
        void operator +=(int); //+= aşırı yüklenmesi
        Dizi operator*(int ) const; //* aşırı yüklenmesi
        int operator [](int i) const {return pointer[i];} //köşeli
↪   parantezlerle erişim
        int & operator [](int i) {return pointer[i];} //köşeli
↪   parantezlerle erişim
        friend ostream & operator << (ostream &out, const Dizi &d);
↪   //cout ile kullanmak için
        unsigned int boyut() const; //_boyut değerini döndür
    private:
        int * pointer; //bellek
        unsigned int _boyut; //boyut
};
```

Dizi Sınıfı

Yapıcı ve Yıkıcı Metotlar

```
Dizi::Dizi(unsigned int boyut) {
    this->_boyut=boyut;
    pointer = new int[boyut];
    for (int i = 0; i < boyut; ++i) {
        pointer[i]=0;
    }
}

Dizi::~Dizi() {
    delete pointer;
}
```

Dizi Sınıfı

boyut ve yazdir Metotları

```
unsigned int Dizi::boyut() const{
    return _boyut;
}

void Dizi::yazdir(){
    for (unsigned int i = 0; i < _boyut; ++i) {
        cout << pointer[i] << " ";
    }
    cout << endl;
}
```

Dizi Sınıfı

+= Operatörü

```
void Dizi::operator +=(int artis) {
    for (unsigned int i = 0; i < _boyut; ++i) {
        pointer[i] += artis;//artis ne arar kodda
    }
}
```

Dizi Sınıfı

* Operatörü

```
Dizi Dizi::operator*(int carpim) const {
    Dizi dondur(_boyut);
    for (unsigned int i = 0; i < _boyut; ++i) {
        dondur[i] = pointer[i]*carpim;
    }
    return dondur;
}
```

Dizi Sınıfı

« Operatörü

```
ostream & operator << (ostream &out, const Dizi &d) {
    out << "Dizi[";
    for (unsigned int i = 0; i < d.boyut(); ++i) {
        if(i!=d.boyut()-1)
            out << d[i] <<", ";
        else
            out << d[i]<<"]"<<endl;
    }
    return out;
}
```

Dizi Sınıfı

Kullanımı

```
#include "Dizi.cpp"

int main(int argc, char* argv[])
{
    for (int i = 5; i < 10; ++i) {
        Dizi a(i);
        a += i;
        Dizi b = a*(i-1);//a*i-1 hata verir
        cout << b;
    }
    return 0;
}
```

Deneyin Uygulanması

```
int main(int argc, char* argv[])
{
        Complex c1(4,5);
        Complex c2(1,2);
        Complex c3=c1-c2;
        cout << c3 << endl;
        cout << c1+c2*c3 << endl;
        cout << -c2 << endl;
        cout << c1[0] << "," << c1[1] << "i" << endl;
        return 0;
}
```

Çıktı:

```
(3,3i)
(1,14i)
(-1,-2i)
4,5i
```

Kaynaklar I

C++ overloading.
https://www.tutorialspoint.com/cplusplus/cpp_overloading.htm.

Operator overloading.
https://isocpp.org/wiki/faq/operator-overloading.