**BM102 - ALGORİTMA VE PROGRAMLAMA – 2**

**ÖDEV - 4**

**AÇIKLAMALAR;**

- Ödevler Bireysel olarak yapılacaktır.
- **https://turnitin.com/login_page.asp?lang=tr** sitemine yüklenecektir.
- **Turnitin** kullanımı **http://akademik.duzce.edu.tr/arafatsenturk/Profil/Dokumanlar** adresinde mevcuttur.
- Sınıf Numarası: **25888105** Sınıf Şifresi:**2468**'dir.
- Ödevler her hangi bir derleyici (DevC gibi) kullanılarak yapılacaktır
- PDF dosyasının ismi şu şekilde olacaktır: ÖğrenciNO.pdf

  Bu standartta gönderilmeyen ödevler değerlendirilmeye alınmayacaktır.
- **Turnitin'a** yükleme yaparken BAŞLIK bölümü de öğrenci No giriniz.
- **Turnitin** sistemindeki Ödev 4'ün son yüklenme tarihi **11 Eylül 2020 saat: 14:00'dır**. Bu tarih ve saatten sonra sistem otomatik olarak kapılacaktır.

- **Turnitin** sisteminin dışında e-mail veya elden teslim edilen ödevler kesinlikle kabul **EDİLMEYECEKTİR.**

# ÖDEV - 4

**1** Aşağıda verilen sınıf tanımlamalarını dikkate alarak UML sınıf diyagramını çiziniz ve aşağıda verilen komutların geçerli olup olmadıklarını belirtiniz.

```
//---------- A.h ----------
#ifndef A_H
#define A_H
class A
{
    int x;
    public:
        A(int _x=0):x(_x){}
        int xAl(){return x;}
        virtual void f() = 0;
};
#endif
//---------- B.h ----------
#include "A.h"
class B: public A
{
    public:
        virtual void f() = 0;
};
//---------- C.h ----------
#include "A.h"
class C: public A
{
    public:
        void f();
};
//---------- C.pp ----------
#include "C.h"
#include <iostream>
using namespace std;
void C::f()
{
    cout<<xAl()<<endl;
}
//---------- D.h ----------
#include "B.h"
class D: public B
{
    public:
        void f();
};
```

```
//----------------D.pp ----------------
#include "D.h"
#include <iostream>
using namespace std;
void D::f()
{
    cout<<xAl()<<endl;
}
```

**a.**
```
B b;
```

**b.**
```
A a;
```

**c.**
```
C c;
A* a = &c;
a->f();
```

**d.**
```
D d;
B* b = &d;
b->f();
```

**e.**
```
C c;
D d;
B* b = &c;
b = &d;
```

**f.**
```
C c;
D d;
A* a = &d;
a = &c;
a->f();
```

**g.**
```
D d;
C* c = &d;
```

**h.**
```
D d;
A& a = d;
a.f();
```

**2** Aşağıda verilen programın çıktısını yazınız.

*//---------- A.h ----------*

```
#ifndef A_H
#define A_H
#include <iostream>
using namespace std;
class A
{  public:
       A(){
         cout << "1. A" << endl;
       }
       ~A()
       {
         cout << "2. ~A" << endl;
       }
       virtual void f1();
       void f2();
};
#endif
```

*//---------- A.cpp ----------*

```
#include "A.h"
void A::f1()
{
    cout << "3. A.f1()" << endl;
}
void A::f2()
{
    cout << "4. A.f2()" << endl;
}
```

*//---------- B.h ----------*

```
#include "A.h"
class B: public A
{
   public:
      B()
      {
        cout << "5. B" << endl;
      }
      void f1();
      void f2();
};
```

```
//---------- B.cpp ----------
#include "B.h"
void B::f1()
{
    cout << "6. B.f1()" << endl;
}
void B::f2()
{
    cout << "7. B.f2()"<< endl;
}
//---------- Uygulama.h ----------
#include "A.h"
#include "B.h"
int main()
{
    A a;
    B b;
    A *aGosterge = new B;
    a.f1();
    a.f2();
    b.f1();
    b.f2();
    aGosterge->f1();
    aGosterge->f2();
    delete aGosterge;
    return 0;
}
```

**3** Aşağıdaki programın çıktısını bulunuz.

```
//---------- Bir.h ----------
#include <iostream>
using namespace std;
class Bir
{
    protected:
        int a;
    public:
        Bir(int _a=0):a(_a){}
        virtual ~Bir(){cout<<"One Destructor"<<endl;}
        void f1();
        virtual void f2();
};
```

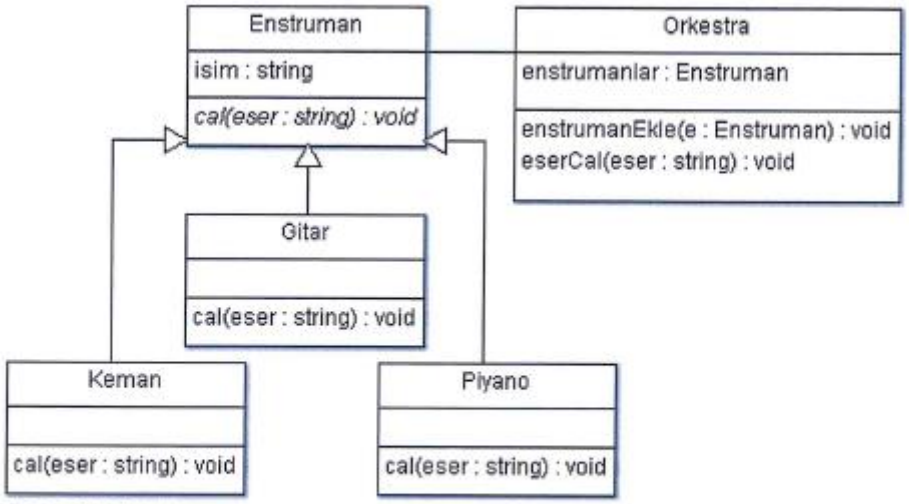
```
//---------- Bir.cpp ----------
#include "Bir.h"
void Bir::f1()
{
    a+=2;
}
void Bir::f2()
{
    cout<<a<<endl;
}
//---------- Iki.h ----------
#include "Bir.h"
class Iki:public Bir
{
    int b;
    int *c;
    public:
        Iki(int _b, int* _c):b(_b),c(_c){}
        void f1();
        virtual void f2();
        virtual ~Iki()
        {
            delete c;
            cout<<"Two Destructor"<<endl;
        }
};
//---------- Iki.cpp ----------
#include "Iki.h"
void Iki::f1()
{
    b+=2;
    *c +=1;
}
void Iki::f2()
{
    cout<<a<<" "<<b<<" "<<*c<<endl;
}
```

```
//---------- Uygulama.h ----------
#include "Bir.h"
#include "Iki.h"
int main()
{
    int a,b;
    int* c;
    cin>>a;
    cin>>b;
    c=new int(b);
    Bir *p[2];
    Iki obj(b,c);
    p[0]= new Bir(a);
    p[1]= new Iki(b,c);
    obj.f1();
    obj.f2();
    for (int i=0;i<2;i++){
        p[i]->f1();
        p[i]->f2();
    }
    delete p[0];
    delete p[1];
    return 0;
}
```

***Örnek Girdi:***

```
3 2 5 4 7
```

**4** Bu soruda bir orkestra nesnesini modellemeniz beklenmektedir. Bir orkestraya herhangi bir enstrüman eklenebilmektedir. Orkestradan bir eseri çalması istendiğinde, orkestrada yer alan tüm enstrümanlar kendilerine özgü eseri çalmaya başlayacaklardır. Problem tanımına ek olarak UML sınıf diyagramı, `Enstruman` sınıfı, `main()` fonksiyonu ve örnek çıktı verilmiştir. Buna göre gerekli sınıf ve fonksiyon tanımlamalarını yapınız.



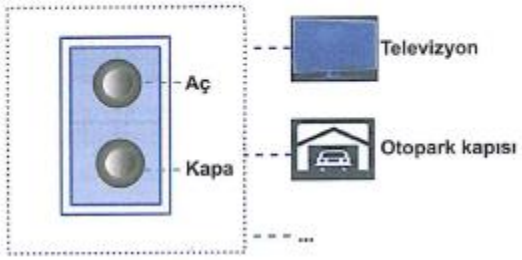

```
class Enstruman
{
    protected:
        string isim;
    public:
        Enstruman(string _isim):isim(_isim){}
        virtual void cal() = 0;
};
     //...
int main()
{
    Orkestra orkestra;
    Keman keman;
    Piyano piyano;
    Gitar gitar;
    orkestra.enstrumanEkle(&keman);
    orkestra.enstrumanEkle(&piyano);
    orkestra.enstrumanEkle(&gitar);
    orkestra.eserCal("Pavane");
   return 0;
}
```

***Örnek Çıktı***

```
Keman caliyor…
Piyano caliyor…
Gitar caliyor…
```

**5** Aşağıdaki şekilde de gösterildiği gibi, çok amaçlı bir uzaktan kumanda yapılmak istenmektedir. Kumanda üzerinde açma ve kapama işlemlerini gerçekleştiren iki adet düğme yer alacaktır. Bu düğmelerin kumanda edeceği cihazlar farklılaşabilecektir. Aşağıda `main()` fonksiyonu ve örnek çıktıda da görüldüğü gibi, bir `UzaktanKumanda` nesnesi yaratılmış ve bu nesnenin açma ve kapama düğmelerine ilk olarak televizyonu açan ve kapayan komutlar atanmıştır. Daha sonra aynı kumanda nesnesine bu defa otopark kapısını açan ve kapayan komutlar atanmıştır. Verilen `main()` fonksiyonu ve örnek çıktıyı dikkate alarak,

- UML sınıf diyagramını çiziniz
- Programın çalışması için gerekli olan sınıf ve fonksiyon tanımlamalarını yapınız




```
int main()
{   UzaktanKumanda kumanda;
    TelevizyonAcKomutu tvAcKomutu;
    TelevizyonKapaKomutu tvKapaKomutu;
    kumanda.acmaKomutuAta(&tvAcKomutu);
    kumanda.kapamaKomutuAta(&tvKapaKomutu);
    kumanda.ac();
    kumanda.kapa();
    OtoparkKapiAcKomutu kapiAcKomutu;
    OtoparkKapiKapaKomutu kapiKapaKomutu;
    kumanda.acmaKomutuAta(&kapiAcKomutu);
    kumanda.kapamaKomutuAta(&kapiKapaKomutu);
    kumanda.ac();
    kumanda.kapa();
   return 0;
}
```

***Örnek Çıktı***

```
Televizyon aciliyor…
Televizyon kapaniyor…
Otopark kapisi aciliyor…
Otopark kapisi kapatiliyor…
```

**6** Bir hastanede iki tip hasta tedavi görmektedir: normal hastalar, ve sigortalı hastalar. Eğer hasta sigortalı ise sigorta şirketi hastane ücretinin tümünü ve ilaç masrafının %80'ini ödemektedir. Normal hastalar hem hastane hem de ilaç ücretini ödemektedir. Hastanede yatan 20 hastanın faturalarını hesaplamak için aşağıdaki `main()` fonksiyonunu kullanarak programı tamamlayınız.

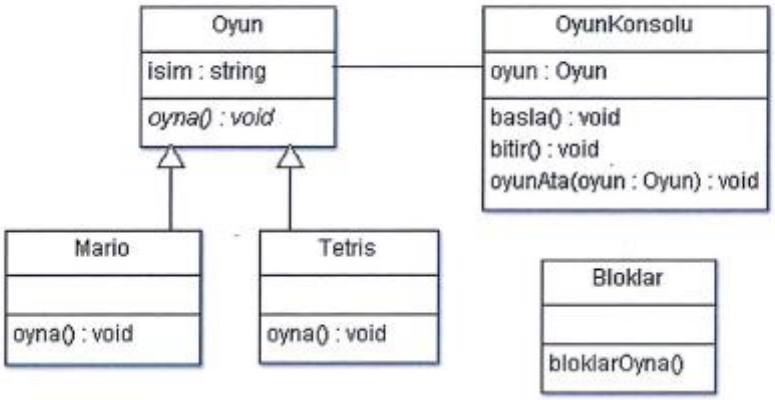
```
int main()
{
    char s;
    string isim,sirket;
    int ilacMasrafi;
    int hastanePayi;
    HastaFatura* faturalar[20];
    for (int i=0;i<20;i++){
        cout<<" Hasta sigortali mi?(E/H):";
        cin>>s;
        cout<<"Hastanin ismi:";
        cin>>isim;
        cout<<"Hastane ucreti:";
        cin>>hastanePayi;
        cout<<"Ilac masrafi:";
        cin>>ilacMasrafi;
        if (s=='E'){
            cout<<"Sigorta sirketi:";
            cin>>sirket;
            faturalar[i]=new SigortaliHastaFatura
                            (isim,hastanePayi,ilacMasrafi,sirket);
        }else
            faturalar[i]=new
HastaFatura(isim,hastanePayi,ilacMasrafi);
        faturalar[i]->odeme();
    }
    return 0;
}
```

***Örnek Çıktı***

```
Hasta sigortali mi?(E/H):H
Hastanin ismi: Ali
Hastane ucreti: 300
Ilac ucreti: 150
Hasta Ismi: Ali --- Toplam Ucret: 450 lira

Hasta sigortali mi?(E/H):E
Hastanin ismi: Veli
Hastane ucreti: 200
Ilac ucreti: 100
Sigorta sirketi: Aksigorta
Hasta Ismi: Veli --- Sigorta: Aksigorta --- Toplam Ucret: 20 lira
…
```

**7** Bilgisayar oyunları için genel amaçlı bir oyun konsolu geliştirilmiştir. Bu oyun konsolu `Oyun` sınıfından türeyen herhangi bir sınıf ile çalışabilmektedir. Konsolun testi için örnek olarak `Tetris` ve `Mario` oyunları geliştirilmiştir. Bu oyunların yanı sıra aşağıda gösterildiği gibi, `Bloklar` adında başka bir oyunun da bu konsol ile çalıştırılması istenmektedir. Ancak aşağıda gösterildiği gibi `Bloklar` ile `Oyun` sınıfı arasında miras ilişkisi yoktur ve bu nedenle konsol ile kullanılamamaktadır. `Bloklar` oyununun da bu konsol ile çalıştırılması sağlanabilir mi? Nasıl? Not: Çözümünüzde `OyunKonsolu`, `Oyun` ve `Bloklar` sınıfında hiçbir değişiklik yapmamanız gerekmektedir.




```
OyunKonsolu konsol;
Mario mario;
Tetris tetris;
Bloklar blok;
konsol.oyunAta(&mario);
konsol.basla();
konsol.oyunAta(&tetris);
konsol.basla();
konsol.oyunAta(&blok);  //geçersiz. Oyun sınıfından türememektedir
konsol.basla();
```

**8** Hayal Film Yapım Şirketi dizi film ve reklam filmi çekmektedir. Çekilen her reklamın ücretini belirlemek için Hayal şirketi, reklamın süresini (dakika olarak) ve gösterileceği zaman dilimini kullanarak aşağıdaki tablodaki fiyatlarla hesap yapar. Şirket, çektiği her dizi filmin ücretini belirlemek için dizinin her bölümünün uzunluğunu (dakika olarak) ve yılda kaç bölüm çekileceğini kullanarak aşağıdaki tablodaki fiyatlarla hesap yapar

| Zaman Dilimi | Fiyat |
|---|---|
| Sabah | 20000 lira/dakika |
| Akşam | 40000 lira/dakika |

| Bölüm uzunluğu | Fiyat |
|---|---|
| 30 dakika | 50000 lira/bölüm |
| 60 dakika | 100000 lira/bölüm |

Soyut bir sınıfın başında yer aldığı bir sınıf hiyerarşisi tanımlayarak Hayal şirketinin çektiği reklam ve dizi film ücretlerini belirleyin. Her sınıfta veri üyelerini okumak için >> operatör yükleme fonksiyonu ve filmin ücretini hesaplayan `hesap()` fonksiyonu çoklu işlev yöntemleri kullanarak tanımlanacaktır. `main()` fonksiyonunda bir reklam bir de dizi film nesnesi için ücret hesaplaması yapın.

***Örnek Çıktılar***

```
Reklam icin (R), dizi film icin (D) girin:R
Reklamin suresini girin:3
Reklamin zaman dilimi (S/A):A
Reklamin fiyatı: 120000 lira
```

```
Reklam icin (R), dizi film icin (D) girin:D
Dizinin bolum sayisini girin: 13
Bolumun uzunlugunu girin:30
Dizinin fiyatı: 650000 lira
```

**9** Kütüphaneden öğrenciler ve öğretmenler kitap ödünç alabilirler. Öğrenciler bir kitabı 3 günlüğüne ve öğretmenler de 5 günlüğüne alabilir. Gecikilen her gün için öğrencinin 2 lira, öğretmenin ise 1 lira ödemesi beklenmektedir. Sistem aylık olarak çalışmaktadır. Bu sistemi yaratmak için aşağıdaki bileşenlerden oluşan bir program yazınız:

- Uye sınıfı:
  - veri üyeleri: `isim`, `alinanGun`
  - üye fonksiyonlar:
    - `>>` operatör yükleme fonksiyonu: veri üyelerini okur
    - `yaz()`: veri üyelerini ekrana yazar
    - `hesapla()`: saf fonksiyon
- Uye sınıfından turemis `Ogrenci` sınıfı
  - veri üyeleri: `ogrenciNo`
  - üye fonksiyonlar:
    - `>>` operatör yükleme fonksiyonu: veri üyelerini okur
    - `yaz()`: veri üyelerini ekrana yazar
    - `hesapla()`: kitabın günü geçmişse ödenecek miktar, geçmemişse kaç gün kaldığını ekrana yazar.
- Uye sınıfından turemis `Ogretmen` sınıfı
  - veri üyeleri: `personelNo`
  - üye fonksiyonlar:
    - `>>` operatör yükleme fonksiyonu: veri üyelerini okur
    - `yaz()`: veri üyelerini ekrana yazar
    - `hesapla()`: kitabın günü geçmişse ödenecek miktar, geçmemişse kaç gün kaldığını ekrana yazar.
- `main()` fonksiyonu kullanıcıdan bir karakter okur (öğrenci için `'O'`, hoca için `'H'`), karaktere göre dinamik bir nesne yaratır. Nesne bilgilerine göre ödenecek miktarı veya iadeye kaç gün kaldığını ekrana yazar.

**10** Aşağıda farklı şablon parametreleri ile tanımlı fonksiyon şablonları ve fonksiyonlar verilmiştir.

```
template<class T1, class T2>
void f(T1, T2);          //1

template<class T>
void f(T);               //2

template<class T>
void f(T, T);            //3

template<class T>
void f(T*);              //4

template<class T>
void f(T*, T);           //5

template<class T>
void f(int, T*);         //6

template<>
void f<int>(int);        //7

void f(int, double );    //8

void f(int);             //9
```

Bu fonksiyonları dikkate alarak aşağıda verilen komutların yukarıdaki hangi fonksiyonu çağıracağını belirtiniz.

```
int             x;
double          y;
float           z;
f(x);          //a
f<int>(x);     //b
f(x, x);       //c
f(x, z);       //d
f(x, y);       //e
f(x, &y);      //f
f(&y, y);      //g
f(&y);         //h
f(y, &x);      //j
f(&x, &x);     //k
```

**11** Parametre olarak gönderilen herhangi bir veri tipindeki diziyi ekrana yazdıran `diziYazdir()` fonksiyon şablonunu tanımlayınız. Fonksiyonun kullanımına ilişkin örnek komutlar ve çıktı aşağıda verilmiştir.

```
string renkler[3] = {"Siyah","Beyaz","Mavi"};
int sayilar[5] = {1,3,5,7,9};
float agirliklar[6] = {3.4,6.7,8.9,7.6,10.1,2.9};
diziYazdir(renkler, 3);
diziYazdir(sayilar, 5);
diziYazdir(agirliklar, 6);
```

***Örnek Çıktı***

```
dizi[0] = Siyah
dizi[1] = Beyaz
dizi[2] = Mavi

dizi[0] = 1
dizi[1] = 3
dizi[2] = 5
dizi[3] = 7
dizi[4] = 9

dizi[0] = 3.4
dizi[1] = 6.7
dizi[2] = 8.9
dizi[3] = 7.6
dizi[4] = 10.1
dizi[5] = 2.9
```

**12** Aşağıda verilen programın çıktısını yazınız.

```
using namespace std;
template <class T>
void f(T &a, T &b)
{
    T temp;
    temp = a;
    a = b;
    b = temp;
}

int main()
{
    int a = 5, b= 7;
    float x=12.3, y=45.3;
    char ch1='a', ch2='b';
    cout<<"a:"<<a<<" b:"<<b<<endl;
    f(a,b);
    cout<<"a:"<<a<<" b:"<<b<<endl;
    cout<<endl;
    cout<<"x:"<<x<<" y:"<<y<<endl;
    f(x,y);
    cout<<"x:"<<x<<" y:"<<y<<endl;
    cout<<endl;
    cout<<"ch1:"<<ch1<<" ch2:"<<ch2<<endl;
    f(ch1,ch2);
    cout<<"ch1:"<<ch1<<" ch2:"<<ch2<<endl;
    return 0;
}
```

**13** Aşağıda verilen programın çıktısını yazınız.

```
#include <iostream>
#include <math.h>
using namespace std;
class NesneYoneticisi
{
    public:
        template<typename T>
        T* nesneYarat( );
        template<typename T>
        void nesneYarat(T*& n);
};
template<typename T>
T* NesneYoneticisi::nesneYarat( )
{
    return(new T);
}
template<typename T>
void NesneYoneticisi::nesneYarat(T*& n)
{
   n = new T;
   cout<<"Yeni nesne atamasi yapildi..."<<endl;
}
class X
{
    public:
        X(){cout<<"X nesnesi yaratildi"<<endl;}
};
class Y
{
    public:
        Y(){cout<<"Y nesnesi yaratildi"<<endl;}
};
int main()
{
    NesneYoneticisi ny;
    X* p1 = ny.nesneYarat<X>( );
    Y* p2 = ny.nesneYarat<Y>( );
    ny.nesneYarat(p1);
    ny.nesneYarat(p2);
    return 0;
}
```

**14** Aşağıda verilen `Kare` sınıfının `kenar` veri üyesi tamsayı tipinde tanımlanmıştır. `kenar`'ın herhangi bir veri tipinde olabilmesi için `Kare` sınıfını şablon olarak tekrar yazın ve kenarı `2.5` ve `9` olan iki `Kare` nesnesinin alanlarını ekrana yazdıran bir `main()` fonksiyonu yazın.

```
class Kare
{
    int kenar;
    public:
        void kenarAta(int k){kenar=k;}
        int kenarAl(){return kenar;}
        int alan();
};
int Kare::alan()
{
    return kenar*kenar;
}
```

**15** DERSTE ÇÖZÜMÜ YAPILDI.

Matematikte sıralı çift (ordered pair) `(x,y)` formatında yazılmış iki değerden oluşur. Aşağıda sadece tamsayı tipinde x ve y üyelerine sahip `Cift` isimli sınıf verilmiştir. Bu sınıfı x ve y herhangi bir veri tipinde olabilecek şekilde sınıf şablonu olarak tekrar yazınız. `Cift` tipinde içeriği `(2.5,'c')` ve `("Ali",3000)` olan iki nesne yaratıp nesneleri ekrana yazdıran bir `main()` fonksiyonu yazınız.

```
class Cift
{
    int bir,iki;
    public:
        Cift(int,int);
        void yaz();
};
Cift::Cift(int a,int b):bir(a),iki(b){}
void Cift::yaz()
{
    cout<<"("<<bir<<","<<iki<<")"<<endl;
}
```

**16** DERSTE ÇÖZÜMÜ YAPILDI.

Katsayıları herhangi bir veri tipi olan polinomları yaratabileceğimiz bir `Polinom` sınıf şablonu geliştirelim. Polinomlar aşağıda görüldüğü gibi terimlerden oluşur ve her terimde bir katsayı ve üs yer almaktadır. Örneğin, $2x^4$ teriminin katsayısı 2, üssü ise 4'tür. $a_n x^n + a_{n-1} x^{n-1} + .... + a_1 x + a_0$,

$a_0,..a_n$: katsayılar
$n$: polinomun derecesi ( polinomdaki en büyük üs)

Bir polinomu terimler dizisi olarak saklayabiliriz. Yaratacağınız `Polinom` sınıf şablonu aşağıdaki bileşenlerden oluşacaktır:

- Veri üyeleri: derece, katsayı dizisi (üsleri indeks, katsayıları da elemanlar olarak alabilirsiniz.
  Örneğin,
  $2x^3 + 4x + 5$ polinomunu aşağıdaki dizide saklayabiliriz.

| 0 | 1 | 2 | 3 |
|---|---|---|---|
| 5 | 4 | 0 | 2 |

- Üye fonksiyonlar
  - Yapıcı fonksiyon: Polinomun derecesini parametre olarak alır ve dinamik bir dizi yaratır.
  - >> operatör yükleme fonksiyonu: dizinin içine katsayıları okur.
  - = operatör yükleme fonksiyonu: bir polinom nesnesini başka bir nesneye atar.
  - << operatör yükleme fonksiyonu: polinomu yazdırır.

`main()` aşağıda verilmiştir.

```
int main()
{
    Polinom<int> a,b;
    cin>>a;
    cin>>b;
    cout<<"Polinomlar:"<<endl;
    cout<<a;
    cout<<b;
    b=a;
    cout<<"Atamadan sonra:"<<endl;
    cout<<b;
    return 0;
}
```

***Örnek Çıktı***

```
Polinomun derecesi: 3
x3 katsayı: 2
x2 katsayı: 0
x1 katsayı: 4
x0 katsayı: 5
Polinomun derecesi: 2
x2 katsayı: 3
x1 katsayı: 0
x0 katsayı: 7
Polinomlar:
2x3+4x2+5x0
3x2+7x0
Atamadan sonra:
2x3+4x1+5x0
```

**17** Bir otomat makinesi, ürünlerin self servis satışına olanak veren, bozuk veya kağıt para, akıllı kart veya kredi kartı veya herhangi bir komut ile çalışan cihazdır. Bu cihazlarla, gıda, gazete, kitap gibi türlü ürünlerin satışı gerçekleştirilebilir. Bir otomat makinesinde, farklı sayıda yuvalar ve her yuvada satılacak ürünün ve ürünün fiyatının belirtildiği etiket bulunur. Bu soruda, içerisine ismi ve fiyatı belirli olan herhangi bir ürünü alan bir `Otomat` sınıfı tanımlamanız beklenmektedir. `Otomat` sınıfının örnek kullanımı ve çıktısı aşağıda verilmiştir.

```
int main ()
{
  Otomat<Urun,5> otomat;
  Tarih uTarih(2,3,2012);
  Tarih sTarih(2,3,2013);
  GidaUrunu cikolata("Cikolota",1.5, sTarih, uTarih);
  GidaUrunu su("Su",0.5, sTarih, uTarih);
  KitapUrunu kitap("C++ Kitabı",20.0,"Cigdem Turhan&Cemile Serce");
  otomat.ekle(cikolata,1);
  otomat.ekle(su,2);
  otomat.ekle(kitap,3);
  otomat.goster();
  return 0;
}
```

***Örnek Çıktı***

```
1- Çikolata   1.5TL
2- Su 0.5 TL
3- C++ Kitabı 20 TL
```