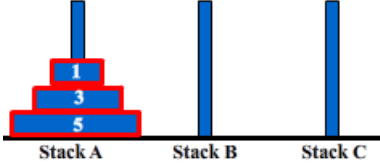


Soru 1)

Aşağıda aynı özelliklere sahip üç kule görünmektedir. Bunlardan A kulesi üzerine sıra ile **5 3 1** değerleri yüklenmiştir. Her bir kule bir yığın gibi düşünülürse, **A yığınınındaki sayıları C yığına verileştirecek kodu cevap kâğıdına yazınız.**

- Kulelere eleman eklenirken küçük değerler büyük değerlerin üstüne gelecek şekilde yerleştirilmelidir. (Örneğin 3 değerinin üstüne 5 değeri gelemez).
- Sayıların A yığınınında olduğu varsayılacak.
- Sayılar sadece başka bir yığından çıkartılıp diğerine eklenebilir. Yeni bir sayı yığınlar eklenemez



```
class Stack {
public:
    bool Push(int item);
    bool Pop(int& item);
private:
    int dizi[10];
    int Top;
};
```

```
int temp;
A.pop(temp);
C.push(temp);
A.pop(temp);
B.push(temp);
C.pop(temp);
B.push(temp);
A.pop(temp);
C.push(temp);
B.pop(temp);
B.pop(temp);
C.push(temp);
A.pop(temp);
C.push(temp);
```

Soru2)

Yandaki C++ kodu derlendiğinde ekran çıktısı ne olur

```
aerhmbalar
aarhmbelar
aaahmbelrr
aaabmhelrr
aaabehmlrr
aaabehmlrr
aaabehlmrr
aaabehlmrr
aaabehlmrr
```

```
#include<iostream>
using namespace std;
char dizi[] = "merhabalar";
void T(char* p){
    if(*(p+1)!='\0') return;
    int k = 0;
    for(int i=1;p[i]!='\0';i++){
        if(p[k]>p[i])
            k = i;
    }
    char temp = p[k];
    p[k] = p[0];
    p[0] = temp;
    cout<<dizi<<endl;
    T(p+1);
}
int main(){
    T(dizi);
}
```

Soru3)

DaireselKuyruk sınıfının prototipi aşağıda verilmiştir. Buna göre sağdaki kodda oluşturulan q[0] ve q[1] kuyruklarının **for** döngüsü sonrasındaki

- Veri dizisi
- ES,KB ve KS

değişkenlerinin değerleri ne olacaktır. (Veri dizisinin elamanları tek satırlık tablo şeklinde yazılmalıdır.)

```
#define MAX 5
class DaireselKuyruk
{
private:
    int Veri[MAX];
    int KB;
    int KS;
    int ES;
public:
    DaireselKuyruk();
    bool Ekle(int newData);
    bool Getir(int & hucre);
};
```

i = 0 için

```
Q[1]
13 5 2
KS=2 KB=1 ES=3
Q[0]
KS=-1 KB=-1 ES=0
```

i = 1 için

```
Q[1]
13 5 2
KS=2 KB=1 ES=3
Q[0]
1 1 2 3 5
KS=4 KB=0 ES=5
```

i = 2 için

```
Q[1]
13 5 2 13 5
KS=4 KB=0 ES=5
Q[0]
1 1 2 3 5
KS=4 KB=0 ES=5
```

i = 3 için

```
Q[1]
13 5 2 13 5
KS=4 KB=0 ES=5
Q[0]
1 1 2 3 5
KS=4 KB=0 ES=5
```

```
int f(int k){
    if(k==2) return 1;
    if(k==1) return 1;
    return f(k-1)+f(k-2);
}
int main(){
    int temp;
    int a[] = {1,4,13,11,22};
    DaireselKuyruk q[2];

    for(int i=0;i<5;i++){
        int k = a[i]%4;
        switch(k)
        {
            case 0:{
                for(int j=1;j<=7;j++)
                    q[k].Ekle(f(j));
                break;
            }
            case 1:{
                for(int j=7;j>1;j-=2)
                    q[k].Ekle(f(j));
                break;
            }
            case 2:{
                for(int j=0;j<=f(8);j++)
                    q[k-2].Getir(temp);
                break;
            }
            case 3:{
                for(int j=f(8);j>=1;j-=2)
                    q[k-3].Ekle(k);
                break;
            }
        }
    }
}
```

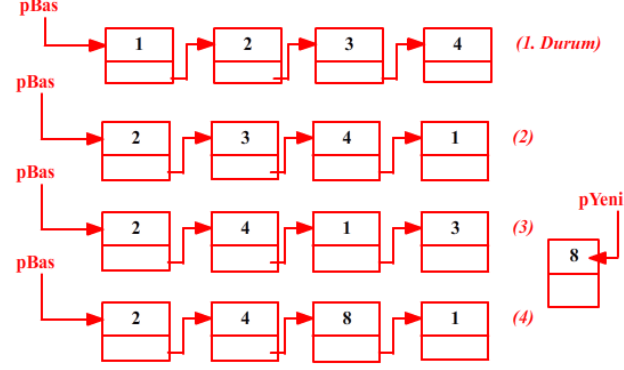
i = 4 için (for döngüsü sonunda)

```
Q[1]
13 5 2 13 5
KS=4 KB=0 ES=5
Q[0]
KS=-1 KB=-1 ES=0
```

Soru 4) Sağda bağılı liste için kullanılacak olan düğüm sınıfı ve aşağıda ise ilk düğümünün adresi **pBas** işaretçisi içerisinde tutulan bir liste verilmiştir. Bu listeyi sırayla birinci durumdan ikinciye, ikinci durumdan üçüncü duruma, üçüncü durumdan da son duruma getirecek c++ kodunu cevap kağıdına yazınız.

- Düğümlere ait olan **Veri** değişkenine dokunulmayacaktır
- pYeni** listeye sonradan eklenecek olan düğümün başlangıç adresini tutmaktadır.

```
class Dugum
{
public:
    Dugum(){
        pSonraki=NULL;
    }
    int Veri;
    Dugum* pSonraki;
};
```



1.Durumdan 2.Duruma

```
Dugum* pYedek = pBas;
Dugum* pTemp = pBas;
while(pTemp->pSonraki!=NULL)
    pTemp=pTemp->pSonraki;
pTemp->pSonraki = pYedek;
pBas = pYedek ->pSonraki;
pYedek->pSonraki = NULL
```

2.Durumdan 3.Duruma

```
pTemp = pBas;
while(pTemp->pSonraki->Veri!=3)
    pTemp=pTemp->pSonraki;

pYedek = pTemp->pSonraki;
pTemp->pSonraki = pYedek->pSonraki;
pTemp = pBas;
while(pTemp->pSonraki!=NULL)
    pTemp=pTemp->pSonraki;
pTemp->pSonraki = pYedek;
pYedek->pSonraki = NULL;
```

3.Durumdan 4.Duruma

```
pTemp = pBas;
while(pTemp->pSonraki->Veri!=1)
    pTemp=pTemp->pSonraki;

pYeni->pSonraki = pTemp->pSonraki;
pTemp->pSonraki = pYeni;

pTemp = pBas;
while(pTemp->pSonraki->pSonraki!=NULL)
    pTemp=pTemp->pSonraki;

pTemp->pSonraki = NULL;
```

Soru 5) Sağda bağılı liste için kullanılacak düğüm sınıfı verilmiştir. Buna göre prototipi aşağıda verilen ve **Veri** değişkeni en büyük değerine sahip olan düğümü listeden çıkartan fonksiyonun gövdesini cevap kağıdına yazınız.

- Fonksiyon parametre olarak listenin ilk düğümünün adresini almaktadır.
- Dönüş değeri de çıkarttığı düğümün adresi olacaktır.

Dugum* EnBuyukElemanCikar(Dugum* pBas);

```
class Dugum
{
public:
    Dugum(){
        pSonraki=NULL;
    }
    int Veri;
    Dugum* pSonraki;
};
```

```
Dugum* pTemp = pBas;
Dugum* pEnBuyuk = pBas;
//En büyük veri elemanına sahip düğüm bulunuyor
while(pTemp!=NULL)
{
    if(pTemp->Veri>pEnBuyuk->Veri)
        pEnBuyuk = pTemp;
    pTemp = pTemp->pSonraki;
}
Dugum* pTemp = pBas;
while(pTemp->pSonraki!=pEnBuyuk)
    pTemp= pTemp->pSonraki;

pTemp->pSonraki = pEnBuyuk->pSonraki;
return pEnBuyuk;
```