Ad/Soyad: 15.11.2014
Numara: Süre: 60 dakika

SAÜ BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ 2014-2015 GÜZ VERİ YAPILARI DERSİ VİZE SINAVI

1- Aşağıdaki a ve b'de istenen algoritma sözde kod şeklinde olacağı için C++ kullanımı zorunlu değildir. (20 p)
   a. İki yönlü bağıl listede araya eleman eklemek için bir algoritma (sözde kod) yazınız.
   b. Dairesel kuyruğa eleman ekleme algoritmasını (sözde kod) yazınız.

a.

```
void insert(eleman, konum){
        ListeGezici *itr = oncekiniKonumuileBul(konum);
        Dugum *onceki = itr.simdiki->ileri;
        itr. simdiki ->ileri = new Dugum(eleman, itr. simdiki ->ileri);

        if(onceki != NULL) önceki->geri = itr. simdiki ->ileri;
}
```

b.

```
void enqueue(eleman) {

        if(isEmpty()){
                on = arka = new Dugum(eleman);
                arka->ileri = on;
        }
        else{
                Dugum *eskiArka = arka;
                arka = new Dugum(eleman, on);
                eskiArka->ileri = arka;
        }
}
```

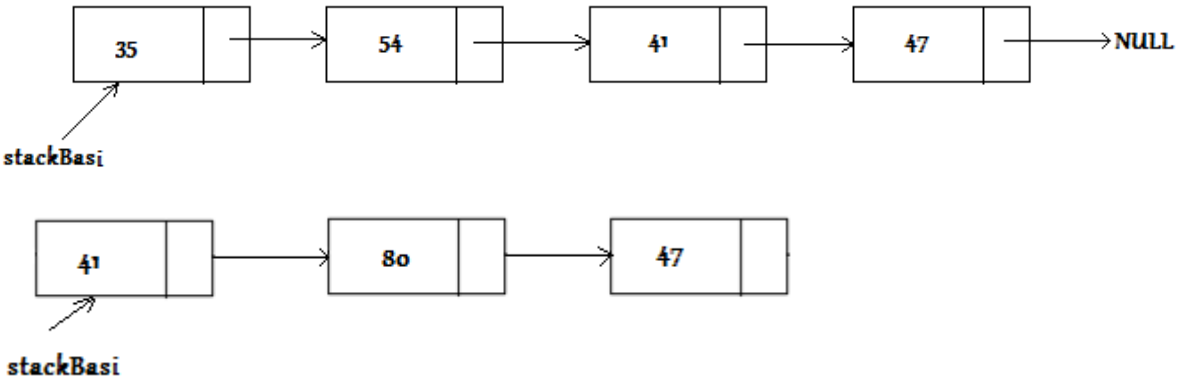
2- Bir labirentte çıkışı bulmak için programcı bir veri yapısından yararlanmak istemektedir. Labirentte gittiği yollara taş koyma suretiyle daha önce geçmiş olduğu yerleri işaretlemektedir. Bu durumda bu programcı hangi en uygun veri yapısından faydalanmıştır? Nedenini bir cümle ile açıklayınız. (10 p)

Stack veri yapısından faydalanmıştır. Taş koyarak ilerlediği için önüne engel çıktığında geri dönecek ve en son attığı taşı alacaktır. Son giren ilk çıkar.

3- Aşağıdaki kod çalıştırıldıktan sonra aşağıda gösterilen yığıtın yeni halini bu sayfanın arkasına çiziniz. (10 p)

```
pop();
pop();
push(80);
Dugum *x = stackBasi->ileri;
stackBasi->ileri = stackBasi->ileri->ileri;
x->ileri = stackBasi;
stackBasi=x;
```

Ad/Soyad: 15.11.2014
Numara: Süre: 60 dakika



4- Verilen ara yüzlere göre yığıt (stack) veri yapısı için, dikdörtgen içerisinde bulunan boşluklara, push (yığıta eleman ekler), pop (yığıtı daraltır) ve top (yığıtta sıradaki verinin değerini döndürür) üye fonksiyonlarını tanımlayınız. Fonksiyonlar içerisinde hata kontrolünü (istisnaları) göz önüne almayınız. Yığıt verilerini saklamak üzere iki yönlü bağlı liste kullanılmıştır. Bu bağlı listenin başını (ilkDüğüm) yığıtın üstü olarak kabul etmelisiniz. (20 p)

```
typedef double DugumVeriTipi;
class Dugum
{
  private:
    DugumVeriTipi dugumVerisi;
    Dugum * sonrakiDugum;
    Dugum * oncekiDugum;
  public:
    Dugum();
    void setDugumVerisi(DugumVeriTipi veri);
    void setSonrakiDugum(Dugum* sonraki);
    void setOncekiDugum(Dugum* onceki);
    DugumVeriTipi getDugumVerisi();
    Dugum* getSonrakiDugum();
    Dugum* getOncekiDugum();
    virtual ~Dugum();
};
class Liste
{
  private:
    Dugum *ilkDugumAdresi;
    Dugum *sonDugumAdresi;
  public:
    Liste();
    virtual ~Liste();
    Dugum * getIlkDugumAdresi();
    Dugum * getSonDugumAdresi();
    void setIlkDugumAdresi(Dugum *);
    void setSonDugumAdresi(Dugum *);
    void push_back(DugumVeriTipi);
    void insert(Iterator konum, DugumVeriTipi eklenecekVeri);
//insert:Belirtilen konuma eklenecek veriyi içeren düğümü ekler.
    Iterator erase(Iterator konum);
//erase: Belirtilen konumdaki düğümü siler
    Iterator begin(); //ilk düğümün konumunu döndürür
    Iterator end(); // son düğümün konumunu döndürür
    void print(Iterator konum);
};
class Stack
{
  private:
    int elemanSayisi;
    Liste * depolamaBirimi; //verileri depolar
  public:
    Stack();
    virtual ~Stack() {}
    void setElemanSayisi(int);
    int getElemanSayisi();
    bool empty() const;
    void pop();
    DugumVeriTipi top();// yığıttaki elemanı döndürür
    void push(const DugumVeriTipi &);
};
```

```
void Stack::push(const DugumVeriTipi &deger)
{

      depolamaBirimi->insert(depolamaBirimi->begin(),deger);
      elemanSayisi++; // this->setElemanSayisi(getElemanSayisi()+1);

}
void Stack::pop()
{
      depolamaBirimi->erase(depolamaBirimi->begin());
      --elemanSayisi; // this->setElemanSayisi(getElemanSayisi()-1);
}

DugumVeriTipi Stack::top()
{
   return ((depolamaBirimi->getIlkDugumAdresi())-> getDugumVerisi());
}
```

5- Dizi ile gerçekleştirilen Kuyruk veri yapısında **on** değişkeni kuyruğun önündeki **arka** değişkeni de kuyruğun arkasındaki elemanın indeksini tutuyorsa Kuyruk'taki eleman sayısını döndüren fonksiyonu C++'ta yazınız. (10 p)

```
int ElemanSayisi() const{
      return arka-on+1;
}
```

Ad/Soyad: 15.11.2014
Numara: Süre: 60 dakika

6- Bir bankada müşterilerin yatırdıkları paralarına faiz uygulanmaktadır. Yatırılan paranın her iki ayda bir faiz artışı gerçekleştirilmektedir. Parasını yatırdığı ay tek sayı ise ilk ay faizi 0,02, eğer çift sayı ise 0,01 dir. Yatırılan para her iki ayda bir 0,01 artırılmaktadır. Paranın miktarını ve yatırım ay sayısı öğrenildikten sonra toplam birikimi hesaplayan programı rekürsif (özyinelemeli) fonksiyon kullanarak gerçekleyiniz. (15 p)

Örnek:
100 TL:
1 ay için:100+100*0.02=102
3 ay için:
100+100*0.02=102
102+102*0.01=103.02
5 ay için:
100+100*0.02=102
102+102*0.01=103.02
103.02+103.02*0.01=104.05
2 ay için :100+100*0.01=101
4 ay için: 100+100*0.01=101
101+101*0.01=102.01
6 ay için: 100+100*0.01=101
101+101*0.01=102.01
102.01+102.01*0.01=103.03

```
double faiz(int miktar,int ay)
{
  if(ay==1) return miktar*1.02;
  if(ay==2) return miktar*1.01;
  else return (faiz(miktar,ay-2)+faiz(miktar,ay-2)*0.01);
}
```

7- Aşağıdaki soruları ilgili yerlere cevaplayınız. (15 p)

a- f(n)=n²+n+16 ifadesinde Big-O notasyonuna göre zaman karmaşıklığı **O(n²)** iken f(n)=3n²+n+16 ifadesinin zaman karmaşıklığı **O(n²)** olarak ifade edilir.

b- Postfix olarak yazılmış olan **5 6 7 + - 3 4 5 * - + 4 ^ 3 +** ifadesinin sonucunu **390628** yazınız. Stack çizerek göstermenize **gerek yoktur**.
Açıklama: b şıkkı büyük bir sayı çıktığı ve işlem yoğunluğu olduğu için herkese 5 puan verilecektir. Buna rağmen doğru yapan kişilere 10 puan verilecektir.

c- Bir kullanıcı web tarayıcısında aynı sekmede önce google.com adresine oradan sakarya.edu.tr adresine oradan bf.sakarya.edu.tr adresine ve son olarak ta cs.sakarya.edu.tr adresine girmiştir. Bilindiği üzere web tarayıcısında geri butonunda Stack (yığıt) mantığı kullanılmaktadır. Eğer Kuyruk mantığı kullanılsaydı ve kullanıcı geri butonuna bir kere basmış olsaydı kendini hangi sitede bulmuş olacaktı. **google.com**