



Sınavda Uyulması Gereken Kurallar

1. Cep telefonlarının, hesap makinesi, saate bakmak gibi herhangi bir amaçla kullanılması yasaktır. Telefon **kapalı** ve **cepte** olmalıdır.
2. **Sınavın başında** sorular kısaca açıklanacaktır. Öğrencilerin soruları cevaplandıktan sonra sınav boyunca soru sormak yasaktır.
3. Soru kağıdına numaranızı ve isminizi yazıp imzalayınız. Soru kağıdı **Sizde kalacaktır**.

NUMARA :

AD SOYAD :

İMZA :

```
void headerTrailer()
{
    header->next->next->prev = trailer->prev;
    trailer->prev->prev->next = header->next;

    header->next->prev = trailer->prev->prev;
    trailer->prev->next = header->next->next;

    header->next->next = trailer;
    trailer->prev->prev = header;

    DoublyNode* headerNext = header->next;
    header->next = trailer->prev;
    trailer->prev = headerNext;
}

void addBack(const string& e)
{
    add(trailer, e);
}

void add(DoublyNode* v, const string& e)
{
    DoublyNode* u = new DoublyNode;
    u->elem = e;
    u->next = v;
    u->prev = v->prev;
    v->prev->next = u;
    v->prev = u;
}

void print()
{
    DoublyNode* first = header;

    while (!(first->next == trailer))
    {
        cout << first->next->elem << endl;
        first = first->next;
    }
}

void main()
{
    DoublyLinkedList list;
    list.addBack("Omer");
    list.addBack("Oguzhan");
    list.addBack("Fatih");
    list.addBack("Ali Osman");

    list.headerTrailer();
    list.print();
}
```

```
void insert(const int& e){
    T.addLast(e);
    Position v = T.last();
    while (!T.isRoot(v)) {
        Position u = T.parent(v);
        if (!isLess(*v, *u)) break;
        T.swap(v, u);
        v = u;
    }
}

void removeMin(){
    if (size() == 1) T.removeLast();
    else{
        Position u = T.root();
        T.swap(u, T.last()); T.removeLast();

        while (T.hasLeft(u)){
            Position v = T.left(u);
            if (T.hasRight(u) && isLess(*(T.right(u)),*v))
                v = T.right(u);
            if (isLess(*v, *u)){
                T.swap(u, v); u = v;
            }else break;
        }
    }
}

void main()
{
    HeapPriorityQueue Heap;
    Heap.insert(1);
    Heap.insert(2);
    Heap.insert(4);
    Heap.insert(6);
    Heap.insert(5);
    Heap.insert(7);
    Heap.insert(11);
    Heap.insert(9);
    Heap.insert(10);
    Heap.insert(8);
    Heap.insert(12);
    Heap.insert(13);
    Heap.insert(14);
    Heap.insert(15);
    Heap.insert(3);

    cout << "Heap Insertions :"; Heap.print();

    Heap.removeMin();

    cout << "Heap removeMin :"; Heap.print();
}
```

1. Yukarıdaki programın çıktısı nedir? (15P)

2. Yukarıdaki programın çıktısı nedir? (20P)

```

void LinkedList::deleteNode(Node* p, int e)
{
    Node* temp; Node* parent;

    while(p != NULL)
    {
        if(p->elt == e) break;
        else
        {
            parent = p;
            if (p->elt > e) p = p->left;
            else p = p->right;
        }
    }

    if( p->left != NULL && p->right != NULL)
    {
        if(parent->left == p)
        {
            parent->left = p->right;
            p->right->par = parent;
            temp = p->right;

            while(temp->left != NULL) temp=temp->left;

            temp->left = p->left;
            temp->left->par = temp;
            p->left = NULL;
            p->right = NULL;
        }
        else
        {
            parent->right = p->left;
            p->left->par = parent;
            temp = p->left;

            while(temp->right != NULL) temp=temp->right;

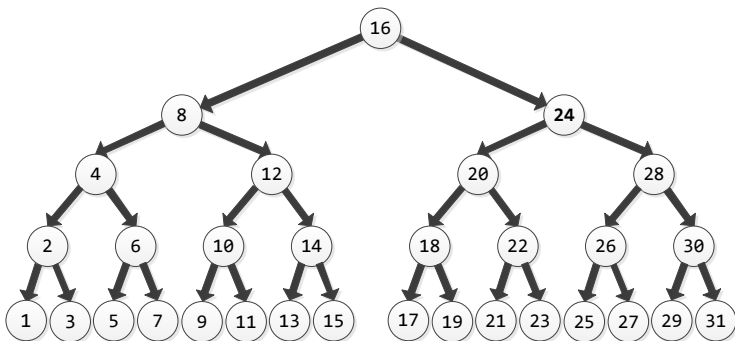
            temp->right = p->right;
            temp->right->par = temp;
            p->left = NULL;
            p->right = NULL;
        }
        delete p;
    }
}

```

```

void main()
{
    // Elemanların daha önce eklendiğini varsayın.
    LinkedList nuTree;
    nuTree.deleteNode(nuTree._root, 8);
    nuTree.deleteNode(nuTree._root, 24);
}

```



Şekil 1

3. Şekil 1 'deki ikili ağaçtan deleteNode() 'a göre 8 ve 24 silindiğinde ağacın son hali ne olur? (30P)

```

int Hash (char* key)
{
    int sum = 0;
    for (int j=0; j<4; j += 2)
        sum = (sum + 10*key[j] + key[j+1]);

    sum = sum % 11 ;
    return sum;
}

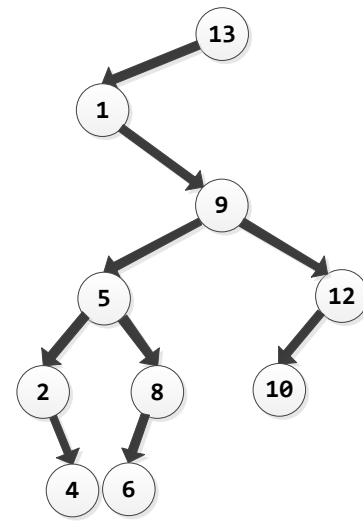
```

dictionary.txt	
ambiguous	belirsiz
array	dizi
artificial	yapay
binary	ikili
child	cocuk
circuit	devre
class	sinif
client	istemci

relative.txt		
0	*	
1	class	sinif
2	circuit	devre
3	child	cocuk
4	ambiguous	belirsiz
5	artificial	yapay
6	*	
7	*	
8	array	dizi
9	binary	ikili
10	client	istemci

4. Yukarıda dictionary.txt'de verilen kelimeleri Hash() fonksiyonunu kullanarak relative.txt'ye yukarıdaki formatta yazınız. Çakışma çözümleme yöntemi olarak linear probing'i kullanınız. (15P)

a	b	c	d	e	f	g
97	98	99	100	101	102	103
h	i	j	k	l	m	n
104	105	106	107	108	109	110
o	p	q	r	s	t	u
111	112	113	114	115	116	117
v	w	x	y	z	ASCII TABLO	
118	119	120	121	122		



Şekil 2

5. Şekil 2 'deki splay ağacına 7 eklendiğinde ağacın son hali ne olur? (20P)

NOT → İkili ağaçtan düğüm silme yöntemi güncellenmiştir. 3. sorudaki yöntem artık derste anlatılmamaktadır.