Data structure HW2 (Assignment 4) 20121324 oh ju young

2.6.7

(a)

J	0	1	2	3	4
Pat	a	a	a	a	b
f	-1	0	1	2	-1

(b)

J	0	1	2	3	4	5
Pat	a	b	a	b	a	a
f	-1	-1	0	1	2	-1

(c)

J	0	1	2	3	4	5	6	7	8
Pat	a	b	a	a	b	a	a	b	b
f	-1	-1	0	0	1	2	3	4	-1

2.8.1

```
void changearray(double *list)
{
        double temp=0; // 1

        for(double i=0; i<n/2; i++)  // even n: n/2+1, odd n:(n-1)/2+1
        {
                temp=list[i];
                list[i]=list[n-i-1];
                list[n-i-1]=temp;     //even n: n/2*3, odd n:(n-1)/2*3
        }
} //total : even n : 2n+2, odd n: 2n
```

3.6.1

(a) A*B*C → AB*C* (b) (A+B)*D+E / (F+A*D)+C → AB+D*EFAD*+/C+

4.3.4

```
int sum(chain &x)
{
        Iterator*a=x.first;
        Iterator*b=a->link->link->link->link;
        int sum=0;

        for(int i=1; i<=n-5; i++)
        {       sum=a.data + b.data;
                a=a->link;
                b=b->link;
        }

        return sum;
}
```

4.10.1

```
void delete(node *x)
{
        x.data=x->right.data;
        x->right=x->right->right;
}
```

5.2.4.

We can make node pointer with node structure which have d components. These components are also pointer that point other node's data value. This node structure will be in the private of Node class.

5.3.1

TREE	INFIX	PREFIX	POSTFIX	LEVEL
(a)	EDCBA	ABCDE	EDCBA	5
(b)	HDIBEAFCG	ABDHIECFG	HIDEBFGCA	4

5.4.1

```
int count(*node)
{
        if(node==NULL) //1
                return 0; //1

        if(node->left==NULL && node->right==NULL) //1
                return 1; //1
        else
                return count(node->left)+count(node->right); //level of tree
}
//total time : 3n+3 (n is level of tree)
```

5.5.1

```
void insertnode(node*l, node*s)
{
        l->left = s->left;
        l->leftThread = s->leftThread;
        l->right = s;
        r->rightThread = true;
        s->left = l;
        s->leftThread = false;

        if(!l->leftThread)
                node *temp = InorderSucc(); //function in the book
                temp->right = l;
}
```