Općeniti skup (Set)

a.t.p. Set

```
elementtype             . . .
Set                     . . .
SeMakeNull(&A)          . . .
SeInsert(x,&A)          . . .
SeDelete(x,&A)          . . .
SeMember(x,A)           . . .
SeMin(A)                . . .
SeMax(A)                . . .
SeSubset(A,B)           . . .
SeUnion(A,B,&C)         . . .
SeIntersection(A,B,&C)  . . .
SeDifference(A,B,&C)    . . .
```

Na predavanjima su obrađene razne implementacije a.t.p. Set: naprimjer pomoću polja bitova i pomoću sortirane vezane liste. To su prilično jednostavne (straightforward) implementacije koje su brze kod nekih, a spore kod drugih operacija. Naprimjer sortirana lista za `Member`, polje bitova za `Min` i `Max` i sortirano polje za `Insert` trebaju $O(n)$. Dobre implementacije (s $O(\log n)$ za sve operacije) su dosta komplicirane (primjer su red-black stabla koja dolaze gotova u STL-u).

Zadatak 1.

Napišite funkciju kojom se implementira `SeUnion` pod pretpostavkom da je skup dan sortiranom listom svojih elemenata. Neka bude neovisna o a.t.p. List.

Rješenje. Slično kao `Merge`, samo trebamo paziti da isti element ne ubacimo dvaput.

```
typedef List Set;

void SeUnion(Set A, Set B, Set* C) {
    position pA, pB, pC;
    elementtype elA, elB;
    pA = LiFirst(A); pB = LiFirst(B);
    pC = LiMakeNull(C);
    while (pA != LiEnd(A) && pB != LiEnd(B)) {
        elA = LiRetrieve(pA, A); elB = LiRetrieve(pB, B);
        if (elA == elB) {
            LiInsert(elA, pC, C);
            pA = LiNext(pA, A); pB = LiNext(pB, B);
        }
        else if (elA < elB) {
            LiInsert(elA, pC, C);
            pA = LiNext(pA, A);
        }
        else {
            LiInsert(elB, pC, C);
            pB = LiNext(pB, B);
        }
```

```
        pC = LiNext(pC, *C);
    }
    while (pA != LiEnd(A)) {
        LiInsert(LiRetrieve(pA, A), pC, C);
        pA = LiNext(pA, A);
        pC = LiNext(pC, *C);
    }
    while (pB != LiEnd(B)) {
        LiInsert(LiRetrieve(pB, B), pC, C);
        pB = LiNext(pB, B);
        pC = LiNext(pC, *C);
    }
}
```

Zadatak 2.

Napišite funkciju kojom se implementira operacija `SeSubset` pod pretpostavkom da je skup prikazan sortiranom vezanom listom (pomoću pointera).

Rješenje. Sjetimo se implementacije liste pomoću pointera:

```
typedef celltype* Set;

int SeSubset(Set A, Set B) {
    celltype* trenA, trenB;
    trenA = A->next; trenB = B->next;    // preskocimo header
    while (trenA != NULL && trenB != NULL) {
        if (trenA->element == trenB->element) {
            trenA = trenA->next;
            trenB = trenB->next;
        }
        else if (trenA->element > trenB->element)
            trenB = trenB->next;
        else
            return 0;
    }
    if (trenA != NULL)
        return 0;
    else
        return 1;      // return (trenA == NULL);
}
```