Pretraživanje liste

```
// trazenje elementa liste
// vraca pokazivac na trazeni element ili NULL ako ga ne nadje
atom *trazi (atom *glava, int element) {
  atom *p;
  for (p = glava; p != NULL; p = p->sljed) {
          if (p ->element == element) return p;
  }
  return NULL;
}
```

Dodavanje na početak liste

```
int dodaj (atom **glavap, int element) {
  atom *novi;
  if ((novi = (atom *) malloc(sizeof(atom))) == NULL)
                  return 0;
  novi->element = element;
  if (*glavap == NULL || (*glavap)->element >= element) {
    // Dodavanje na pocetak liste
    novi->sljed = *glavap;
    *glavap = novi;
  }....
```

Dodavanje unutar liste

```
int dodaj (atom **glavap, int element) {
  atom *novi, *p;
  if ((novi = (atom *) malloc(sizeof(atom))) == NULL)
                  return 0;
  novi->element = element;
// ako element dodajemo unutar liste
  for (p = *glavap; p->sljed &&(p->sljed)->element < element; p = p->sljed);
                  novi->sljed = p->sljed;
                  p->sljed = novi;
...
```

Brisanje elementa s početka liste

```
int brisi (atom **glavap, int elem) {
  atom *p;
  for (; *glavap && (*glavap)->elem != elem; glavap = &((*glavap)->sljed));
  if (*glavap) {
          p = *glavap;
          *glavap = (*glavap)->sljed;
          free (p);
       return 1;
  } else return 0;
}
```

Brisanje elementa iz sredine liste

```
int brisi (atom **glavap, int elem) {
  atom *p;
  for (; *glavap && (*glavap)->elem != elem; glavap = &((*glavap)->sljed));
  if (*glavap) {
          p = *glavap;
          *glavap = (*glavap)->sljed;
          free (p);
       return 1;
  } else return 0;
}
```

Prikaz stabla dinamičkom strukturom

```
struct cvor{
  tip podatak;
  struct cvor *lijevo_dijete;
  struct cvor *desno_dijete;
/* ako treba: */
  struct cvor *roditelj;
};
```

Dodavanje elementa u stablo

```
struct cvor* dodaj(struct cvor* cvor, tip elem) {
  if (cvor == NULL) {
    return(NoviCvor(elem));
  }
  else {
    if (elem <= cvor->podatak)
            cvor->lijevo = dodaj(cvor->lijevo, elem);
    else
       cvor->desno = dodaj(cvor->desno, elem);
  return(cvor);

  }
}
```

Funkcija koja stvara novi čvor

```
struct cvor* NoviCvor(int elem) {
  struct cvor* novi =
                    (cvor *) malloc(sizeof(struct cvor));
  cvor->podatak = elem;
  cvor->lijevo = NULL;
  cvor->desno = NULL;
  return(cvor);
}
```

Pretraživanje stabla

```
int trazi (struct cvor* cvor, int trazeno) {
  if (cvor == NULL) {
    return 0;
  }
  else {
    if (trazeno == cvor->podatak) return 1;
    else {
      if (trazeno < cvor->podatak)
                    return(trazi(cvor->lijevo, trazeno));
      else return(trazi(cvor->desno, trazeno));
    }
  }
}
```