Functia creeazaNod(e) este { $\theta(1)$ }

{creeaza un nod avand informatia utila 'e' si cei doi descendenti NIL}

```
        aloca(p)        {p:↑Nod}
        [p].e←e
        [p].st←NIL
        [p].dr←NIL
        [p].h←0
        creeazaNod←p
```

sf_creeazaNod

Functia adauga_rec(p, e) este { $O(\log_2 n)$ }

{se adauga informatia utila 'e' in subarborele de radacina 'p' si se returneaza noua radacina a subarborelui }

```
        Daca p=NIL atunci
                p← creeazaNod(e)
          altfel
                daca e.c>[p].e.c atunci
                        [p].dr←adauga_rec([p].dr,e)
                        daca h([p].dr)-h([p].st)=2 atunci
                                daca e.c>[[p].dr].e.c atunci     {caz Ia}
                                        p←SRS (p)
                                   altfel                        {caz Ib, Ic}
                                        p←DRS (p)
                                sfDaca
                           altfel
                                [p].h←inaltime(p)
                        sfDaca
                    altfel
                        daca  e.c<[p].e.c atunci
                                @ simetric pe partea stanga – rotatii spre dreapta
                        altfel
                                @ cheie duplicat – nu e permisa in AVL
                        sfdaca
                sfDaca
        sfDaca
        adauga_rec←p
```

sf_adauga_rec

Subalgoritm adauga(ab, e) este { $O(\log_2 n)$ }

{se adauga informatia utila 'e' in arborele 'ab' si se returneaza arborele rezultat}

```
        ab.rad←adauga_rec(ab.rad, e)
```

sf_adauga