Objected Oriented Programming, Autumn 2017
Problema 5 (Backtracking)

Vasilescu Rodica - Mihaela

November 3, 2017

Second Year
Group: C.E.N 2.2B

1 Introducere

Problema 5. Sa se proiecteze si sa se implementeze un modul pentru numerele intregi pozitive. Modulul trebuie sa permita rezolvarea urmatoarelor probleme (toate se refera la numarul n):

- Generarea partitiilor numarului n (trebuie determinate toate modalitatile de scriere a lui n sub forma: n = $p_1 + p_2 + ... + p_k$, unde $p_1, p_2, ..., p_k$sunt numere naturale ce verifica relatia: $p_1 < p_2 < ... < p_k$);
- Determinarea celui de-al n-lea numar palindrom (primul numar palindrom este 1);
- Determinarea submultimilor cu suma n (generarea tuturor submultimilor de numere intregi pozitive pentru care suma elementelor este n);
- Afisarea numarului.

2 Aspecte generale

2.1 Backtracking

Backtracking este numele unui algoritm general de descoperire a tuturor solutiilor unei probleme de calcul, algoritm ce se bazeaza pe construirea incrementala de solutii-candidat, abandonand fiecare candidat partial imediat ce devine clar ca acesta nu are sanse sa devina o solutie valida.

Tehnica backtracking se poate aplica doar pentru probleme ce admit conceptul de candidat partial de solutie si ofera un test relativ rapid asupra posibilitatii ca un astfel de candidat sa fie completat catre o solutie valida. Cand se poate aplica, insa, backtrackingul este adesea mult mai rapid decat cautarea prin metoda fortei brute prin toti candidatii, intrucat este capabila sa elimine dintr-un singur test un mare numar de candidati.

Backtrackingul este util la rezolvarea unor probleme de satisfacere a constrangerilor, cum ar fi cuvintele incrucisate, jocuri de sudoku si alte probleme similare. Ea sta la baza unei serii de limbaje de programare logica, cum ar fi Icon, Planner si Prolog.

Termenul "backtrack" a fost inventat de matematicianul american D. H. Lehmer in anii 1950.

3 Pseudocod

3.1 Generarea partitiilor numarului n

Definitie:

- Partitia numarului natural nenul n este un sistem de numere naturale nenule $(x_1, x_2, ..., x_k)$ astfel incat $x_1 + x_2 + ... + x_k = n$

Reprezentarea soluiei:

- fiecare componenta x_k are valori in mulimea 1,2,.,a_k
- soluia are numar variabil de componente k

Condiia de validare:

- suma pariala $x_1 + x_2 + ... + x_k \leq n$

Initial:

- s = 0
- $x_1 = x_2 = ... = x_n = 0$

PARTITIE (k, s, x, n)
if $s = n$ **then**
    print x_1 + x_2+ ... +x_{k-1} = n
else for $i \leftarrow x_{k-1} + 1$ **to** $s - n + 1$
    $x_k = 1$
    $s = s + x_k$
    $\triangleright$ PARTITIE ($k + 1$, s, x, n)
    $s = s - x_k$

Figure 1: Partitia unui numar n

3.2 Determinarea celui de-al n-lea numar palindrom

Definitie: Un palindrom este un sir de caractere (de obicei cuvinte, fraze sau numere) care citit de la stanga la dreapta sau de la dreapta la stanga ramane neschimbat .

PALINDROM (n)
1. $i = 1$
2. $nr = 0$
3. **while** $i \leq n$ **do**
4. $\quad nr \leftarrow nr + 1$
5. $\quad$ **if** $invers(nr) = nr$ **then**
6. $\qquad i \leftarrow i + 1$
7. **return** nr

Figure 2: Al n-lea palindrom

3.3 Suma Submultimi

Definitie: Se spune ca multimea B este submultimea multimii A daca B "este continuta" de A. Echivalent, putem scrie $B \supseteq A$, citit B include A, sau B contine A. Relatia dintre multimi stabilita de $\supseteq$ se numeste incluziune sau continere.

SUBMULTIME (k, s, x, n)
1. **for** $i \leftarrow x_{k-1} + 1$ **to** $s + 1$
2. $\quad x_k = 1$
3. $\quad$ **if** $x_k = s$ **then**
5. $\qquad$ **print** $x_1, x_2, ..., x_k$
6. $\quad$ **else** $\triangleright$ SUBMULTIME ($k + 1$, s, x, n)

Figure 3: Suma de submultimi