1. Șah nedeterminist

David s-a apucat să joace sah. Surprinzator, in ultimul timp a inceput sa castige mereu, dar el are un secret – are acces la un calculator nedeterminist, care pentru fiecare mutare a adversarului poate alege in mod nedeterminist urmatoarea mutare, alegand pe baza unei functii care calculeaza daca o mutare poate fi castigatoare sau nu. Algoritmul poate analiza in viitor oricate mutari, pana cand jocul se termina, si functioneaza in felul urmator: se alege o mutare de-a lui David in mod nedeterminist, apoi se alege o mutare de-a adversarului in mod nedeterminist si tot asa, pana cand jocul se termina. Daca macar o mutare este castigatoare, se intoarce aceasta mutare.

Imaginati un algoritm posibil in pseudocod pentru a implementa acest joc. Pseudocodul poate fi oricat de simplu, cat timp se ofera functionalitatea descrisa.

f(tabla) - functie de decizie

f(tabla) = 1 ==> David castiga

f(tabla) = 0 ==> remiza

f(tabla) = -1 ==> oponentul castiga

Sah\_nedeterminist()

while(true) //cat timp nu s-a terminat jocul

mutare\_d = choice(piese d) //generare mutare pentru David

mutare\_o = choice(piese o) //generare mutare pentru oponent

if f(tabla) = 1 //David are mutare buna

success //David castiga

break //s-a terminat jocul

else

if f(tabla) = 0 || f(tabla) = -1 //daca e remiza sau oponentul a avut mutare buna

fail //jocul se incheie in fail

break //s-a terminat jocul, se iese din while