Pentru rezolvarea problemei am declarat o structură (**booked**) care reține ora de început, respectiv ora încheierii unei întâlniri. //float in,sf;

Declararea variabilelor:

**n**- numărul de meeting-uri pentru c1;

**meet**- cu ajutorul ei verific dacă cei 2 clienți au un interval orar comun pt. a se putea întâlni;

**minutes**- numărul de minute alocat pt. întâlnirea dintre cei doi clienți;

**v**- vector de tip structură (booked) care reține orele meeting-urilor pt. clientul1;

**nr**- se memorează numarul de întâlniri din c1 și c2 (dacă există);

**c1**, **c2**- intervalul orar în care cei doi clienți sunt disponibili.

Cu ajutorul unei funcții **void read(...)** am citit din fișier datele de intrare. De asemenea, am verificat dacă cei doi clienți au interval/e comun/e pt. a se putea întâlni, caz în care afișez un mesaj corespunzător și modific (dacă este cazul) variabila meet.

Pentru rezolvarea problemei am folosit metoda greedy, ordonând crescător după ora deî nceput a fiecărei ședințe, în cazul în care 2 ședințe încep la aceeași oră am ordonat crescător după ora de terminare a ședinței. (funcția **void sorting(int n,booked v[])**)

Pentru afișarea soluției am verificat dacă suma dintre **minutes** și ora finală a ședinței i, este mai mică sau egală cu ora de început a ședinței i+1. În caz afirmativ afișez ora finală a ședinței i și ora de început a ședinței i+1.

În main am verificat dacă c1 și c2 se pot întâlni (cu ajutorul variabilelor meet și nr); dacă cei doi nu se pot întâlni se afișează în fișier un mesaj corespunzător și se incheie execuția programului, în caz afirmativ se afișează un mesaj, se apelează funcțiile și se afișează intervalele orare în care c1 și c2 se pot întâlni.

Presupuneri:

* se cunosc numărul de meeting-uri pentru c1;
* variabila **minutes** este strict mai mica decât 60;
* partea fracționară a sumei dintre **minutes** și **ora finală a ședinței i** este strict mai mică decât 6.