8. Praktika. Bezero/zerbitzari eredua

1.- B baliabide bati dagozkion eskaerak tratatzen dituen **zerbitzari1_funtz** izeneko programa bat eraiki behar da. Eskaerak MBX_ZERB izeneko buzoi batean kokatzen dira, eta bezeroaren kodeak ondoko eskema jarraituko du:

```
struct eskaera {
   int bezeroa;
   char bez_buzoia[20];
   char zerbitzatzeko[40];
 }
```

```
struct erantzuna {
   int bezeroa;
   char emaitza[50];
 }
```

```
main(int argc, char *argv[])
{
  struct eskaera esk;
  struct erantzuna eran;
  int fd_zerb, fd_bez;
  ...
  fd_zerb = open("MBX_ZERB", O_WRONLY);
  sortu_izena(esk.bez_buzoia); //bezeroaren buzoiari izen bat eman
  esk.bezeroa = 76154221; //bezeroaren identifikadorea, kodea
  strcpy(esk.zerbitzatzeko, "B baliabidean 2010 errenta kalkulatu");
  mkfifo(esk.bez_buzoia, 0666);
  fd_bez = open(esk.bez_buzoia, O_RDWR);
  write(fd_zerb, &esk, sizeof(struct eskaera));
  read(fd_bez, &eran, sizeof(struct erantzuna));
  ...
}
```

Bi bezero ezin dute aldiberean erabili **B** baliabidea. Bezero batek eskaera sinkrono bat egiten duenean blokeatuta geratuko da bere ***eskaera zerbitzatzen hasten den arte***. Bidaltzen den mezuan bezeroak sinkronizatzeko erabiltzen duen buzoiaren izena ere barneratzen da.
Eskaera bat zerbitzatzeko ondoko **funtzio** hau erabil daiteke:

```
zerbitzatu_B_baliabidea(char *zerbitzua);
```

2.- Aurreko zerbitzariaren **zerbitzari2_funtz** izeneko bertsio berri bat eraiki behar da, bezeroa desblokea dadin bere ***eskaera zerbitzatzen bukatzen denean***.

3.- zerbitzari1_prog programa, **zerbitzari1_funtz** zerbitzariaren bertsio berri bat eraiki behar da, zerbitzatu_B_baliabidea, **programa** bat bada, eta programa honek, emaitza itzuliko du itzulera-kodea bezala.

```
zerbitzatu_B_baliabidea  zerbitzua
```

4.- zerbitzari2_prog programa, **zerbitzari2_funtz** zerbitzariaren bertsio berri bat eraiki behar da, zerbitzatu_B_baliabidea, **programa** bat bada, eta programa honek, emaitza itzuliko du itzulera-kodea bezala.

```
zerbitzatu_B_baliabidea  zerbitzua
```

5.- zerbitzari3A izeneko zerbitzari berri bat idatzi behar da. Kasu honetan **N** baliabide ditugu erabiltzaileek modu konkurrentean erabiltzeko prest. Bezeroek baliabide bat baino gehiago eska dezakete eskaera bakoitzean, baina eskaera zerbitzatuko da soilik eskatu bezain beste baliabide libre daudenean. Lehen bertsio honetan, bezeroa desblokeatu beharko da bere ***eskaera buzoitik jasotzen den unean***.

Bezeroaren kodean aldaketa bakarra hau izango da:

```
struct eskaera {
   int bezeroa;
   char bez_buzoia[20];
   char zerbitzatzeko[40];
   int kopurua;
 }
...
esk.kopurua = 4; //Zenbat baliabide eskatu behar dituen
...
```

Baliabidea erabiltzeko, zerbitzatu_B_programa izena duen **programa** erabiliko dugu:

`zerbitzatu_B_programa  zerbitzua  kopurua`

Programa honek 2 argumentu ditu, zerbitzatzeko eskatu nahi dena eta eskatutako B baliabide kopurua; programa honek emaitza gisa —itzulera-kode moduan— erabilitako baliabide-kopurua itzuliko du.

6.- zerbitzari3B izeneko zerbitzari berri bat idatzi behar da. Aurreko bertsioarekin alderatuz, zerbitzatzeko ***baliabide nahikoak daudela ziurtatzen denean desblokeatuko da bezeroa***, hau da, eskaera zerbitzatzen hasten denean.

7.- zerbitzari3C izeneko zerbitzari berri bat idatzi behar da. Aurreko bertsioekin alderatuz, bezeroa desblokeatu behar da bere eskaera zerbitzatzen bukatzen denean.