C PROGRAMAZIO-LENGOAIA (VII)

Datu-egiturak (I)

Iñaki Alegria & Montse Maritxalar

BIGARREN kapituluan aztertutakoaren arabera, karaktereak, zenbaki osoak eta zenbaki errealak dira C-ren oinarrizko datuak. Horien gain, C programak idaztean, ondoko datu-motak erabil daitezke: taulak, erakusleak, egiturak, bildurak eta enumeratuak. Bestetik, goi-mailako PASCAL bezalako lengoaietan datu-mota berriak defini daitezkeen bitartean, C-k ez du horrelakorik onartzen, **typedef** espresioaren bidez datu-moten sinonimoak defini badaitezke ere. Beraz, kapitulu honetan eta hurrengoan aipatutako datu-mota zein sinonimoen definizioa eta erabilpena azalduko dugu.

TAULAK ETA ERAKUSLEAK

Taula da edozein lengoaiatan gehien erabiltzen den datu-egitura. Bektore, matrize, array, katea eta string hitzak erabili ohi dira taularen sinonimo gisa, batzuetan taula-motaren azpimotak ez badira ere.

Elkarrekin gorde eta izen berarekin ezagutzen den mota bereko aldagai-multzo batek osatzen du taula. Taulak dimentsio bakarra edo anitza eduki dezake, lehen kasuan bektore ere deitzen zaiolarik. Bi dimentsiokoei matrize deitzen zaie, eta osagaiak bektorearen barruan identifikatzeko indize bat erabiltzen den bitartean, matrizetan bi indize erabiliko dira.

Taularen osagaiak karaktereak direnean, taulari karaktere-katea, katea edo string deitzen zaio, mota honek ezaugarri desberdinak dituelarik.

Taulen definizioa eta erabilpena

Taulak definitzeko osagaien datu-mota, taularen izena eta osagai-kopurua mako artean zehaztu egin behar dira mako artean. Erabilpenean, taula osoaz aritzeko taularen izena soilik aipatzen da. Osagai bat adierazteko aldiz, izena eta indizea mako artean aipatu egin behar dira. Indizeak osagaiaren taula barruko posizioa adierazten du, baina lengoaia gehienetan indizearen balio-tartea batetik osagai-kopururaino (N-raino) den bitartean, C-n zerotik (N-1)eraino erabiltzen da. Kontutan hartzekoa da mako artean zehazten den indizea edozein espresio oso izan daitekeela; konstantea, aldagaia edo beren arteko konbinazio aritmetikoa.

1. programan adibide bat ikus daiteke, non 12 hilabeteko salmentak taula batera irakurri ondoren urteko salmenta eta hileko portzentaiak kalkulatu eta inprimatzen diren. Taula irakurtzeko 3. programan definituko dugun funtzio bat erabiliko da.

```
main ()
{
long salmen [12]; /*taularen definizioa*/
int i;
float ehuneko;
long osora;
void iraur-taula (); /*funtzioaren erreferentzia*/
irakur-taula (salmen, 12); /*funtzioaren ideia*/
for (i=Ø; i<12; i++)
   osora = osora + salmen [i];
for (i=Ø; i<12; i++)
   printf ("%d\t %d\t %f4.2\n",
        i+1, salmen [i], (salmen [i] * 100 /osora);
printf ("guztira\t %d\n, osora);
```

1. programa. Taularen definizioa eta erabilpena.

Definizioan taulen hasieraketa egin daiteke, horretarako osagaien balioak komen bidez banandu eta {

hasieran eta } bukaeran zehaztu behar delarik. Adibidez, hilabete bakoitzak zenbat egun duen adierazten duen taularen definizioa honelakoa litzateke C-z:

int hil-egun [12] = {31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31};

Ezer ez jartzekotan zeroen bidez hasieratzen dira taulak.

Erakusleak

Esan dugunez, taula baten izena indizerik gabe erabiltzen denean taula osoa aipatzen da, baina horrek ez du osagai guztien balioa esan nahi; baizik eta taula memorian kokatzen deneko tokiaren erreferentzia. Kontzeptu hau erakuslearen berbera da eta horrexegatik esaten da taularen izena erabiltzen denean bere erakuslea zehazten dugula. Arrazoi horrexegatik bi taulen arteko asignazio edo esleipena burutzeko ere osagaiz osagai egin behar da. Ezin da asignazio soil batez burutu.

Dena den, taulekin erabiltzeko joera handia badago ere erakusleak datu-mota guztiekin erabil daitezke. Erakusleak aldagaiak dira; beraz, definitu egin behar dira eta memorian kokatuko dira, memoria zati bat hartuko dutelarik. Pascalez erakusleek, definitu behar badira ere, ez dute memoria zatirik hartzen, dinamikoak bait dira, baina C-z estatikoak direnez definiziorako * karaktere berezia erabiltzen da.

Erakusleak (pointer ingelesez) datu baten memoriako helbidea adierazten du eta beti lau bytetan kokatzen da. Definizioan ondoko datuak zehaztu behar dira: erakusten duen datu-mota, * eta erakusle-aldagaiaren izena.

Erabilpenerako * eta **&** eragileak erabiltzen dira. *-k Pascal-eko ↑ eragilearen funtzio bera duen bitartean (erakusleak erakutsitako aldagaiaren balioa alegia), & erakusle ez diren aldagaien aurrean zehazten da, beraien erreferentzia (edo helbidea) eta ez balioa (edo edukina) aipatzeko.

2. programan bi aldagaien arteko balioak trukatzeko funtzio bat definitzen da. Bi parametroak datuak eta emaitzak dira aldi berean. Beraz, ezin da balioaren bidez erreferentziaz baizik trukatu (beste edozein lengoaiatan bezala).

Kontu handia izan behar da erakusleek eragiketa aritmetikoetan parte hartzen dutenean, zeren gehitzen edo kentzen zaien unitatea bat izan beharrean erakutsitako aldagaiaren luzera bait da.

Horregatik ondoko programa-zatian

```
char *p1; short *p2; long *p3;
...
p1++; p2++; p3++;
```

p1 aldagaiari bat (karaktere baten luzera) p2-ri bi eta p3-ri lau (long aldagaien luzera) gehitzen zaio. Bestetik erakusleen arteko batuketak eta kenketak onartzen dira, erakutsitako datu-motak berberak badira.

Taulak funtzioen parametro

Lehen programan irakur-taula funtzioa definitu gabe utzi dugu. Bere lehen parametroa taula denez bi aukera daude parametroaren zehaztapena egiteko: array bezala edo erakusle bezala, 3. eta 4. prometan ikus daitezkeenez (biak baliokideak dira).

```
void swap (x,y)
int *x, *y; /* parametroak bi erakusle dira */
{
int tmp; /* bertako aldagaia */
tmp = *x; /*x-k erakutsitako balioa → tmp */
*x = *y; /*y-k erakutsitakoa  x-k erakutsitakoaren ordez */
*y = tmp; /*tmp-n gordetakoa y-k erakutsitakoaren ordez */
}
main ()
{
int a, b;
scanf ("%d %d", &a, &b); /* irakurtzeko erreferentziaren
bidez */
swap (&a, &b); /* erreferentziaren bidez
printf ("%d %d\n", a, b);
}
```

2. programa.

```
void irakur-taula (taula, os-kop)
long taula [ ];
int os-kop;
{
int i;
for (i= Ø; i < os-kop; i++)
 scanf ("%d", & taula [i]);
}
```

3. programa.

```
void irakur-taula (p, os-kop)
long *p;
int os-kop;
{
long * paux;
for (paux = p; paux < (p + os-kop);
paux ++)
  scanf ("%d", paux);
}
```

4. programa.

Ondoko desberdintasunak hartu behar dira kontutan:

- Lehenengo soluziobidean indizeak erabil daitezke eta ondorioz laneko aldagaia (i) osoa da. Bigarrenean aldiz, indizerik gabe burutuko da eta laneko aldagaia (paux) erakuslea izango da.

- scanf funtzio estandardarrak datua erreferentziaz eskatzen duenez, lehenengo ebazpidean **&** erabiltzen da (taula [i] long motakoa bait da), baina bigarrenean ez (paux erakuslea, eta beraz erreferentzia, bait da).

- Bigarren ebazpidean paux ++ egiten denean paux-i lau gehitzen zaio (long-i dagokion luzera) eta p + os--kop espresioa ebaluatzean erakusleari os-kop bider lau gehitzen zaio.

- Programa nagusian irakur-taula (salmen, 12) jarri beharrean irakur-taula (& salmen [0], 12) idatz daiteke.

Karaktere-kateak

Karaktere-katea edo string bat, karakterez osatutako '**\Ø'** karaktere berezia bukaeran duen taula da. Gainerako taulekiko ondoko diferentziak dituzte:

- String motako konstante batez hasiera daiteke. Adibidez, char izenburua [2Ø] = "URTEKO SALMENTAK";

- Ez da luzera jakin behar (funtzioetan eta makroetan), bukaera detektatzeko '\Ø' karakterea bilatzea nahikoa bait da.

- Scanf eta printf liburutegiko funtzioetan formatu berezi bat dago kateetarako eta ez gainerako tauletarako.

- Liburutegi estandarrean funtzio-multzo bat dago string maneiurako (strlen, strcpy, strcmp, ...)

5. eta 6. programetan karaktere bat kate batean zenbat aldiz agertzen den kalkulatzeko funtzioa azaltzen da.

```
int kont-kar (katea, kar)
char katea [ ], kar;
{
int kont = Ø, i;
for (i = Ø; katea [i] ! = '\Ø'; i++)
  if (katea [i] = = kar) kont++;
return (kont);
}
```

5. programa.

```
int kont-kar (p, kar)
char *p, kar;
{
int kont = Ø;
char * paux;
for (paux = p; * paux! = '\Ø'; p++)
  if (*paux = = kar) kont ++;
return (kont);
}
```

6. programa. Kont-kar erakusleen bidez.

Bestelakoak

- Dimentsio anitzeko tauletan, definizioak zein erabilpenak zenbaki bat mako artean eskatzen du dimentsio bakoitzeko.

  Adibidez,

  int matrize [1Ø] [10]; /* 1Ø ilada bider1Ø zutabe*/

- Erakusleez osatutako taulak defini daitezke. Adibidez **char * hil-izen [12]** definizioaren bidez. Hil-izen taulak 12 erakusle dauzka, bakoitzak string bat erreferentziatzen duelarik.

- Erakusle batek erreferentziatzen duena beste erakusle bat izan daiteke.

  Adibidez:

```
int r, *p, **q;
r = 1Ø
p = &r; /* *p = 1Ø */
q = &p  /* **q = 1Ø */
```