C PROGRAMAZIO-LENGOAIA (II)

OINARRIZKO DATUAK

Iñaki Alegria & Montse Maritxalar

BESTE lengoaietan bezala C lengoaia erabiliz programak idazten ditugunean, datuak definitu egin behar dira. Konstanteak edo aldagaiak izanik, dagokien mota azaldu beharko da konpiladoreak memori zatia eta bit-segida egokia eslei diezaien.

Datu-motak

C-k ondoko oinarrizko datuak ezagutzen ditu: karaktereak, zenbaki osoak, zenbaki errealak eta doitasun bikoitzeko zenbaki errealak. (1. taulan mota bakoitzeko ezaugarriak ikus daitezke).

MOTA	HITZ GAKOA (aldagaien definizioan)	BIT-KOPURUA	ADIERAZPIDEA
karakterea	char	8	ASCII
osoa	int	16 edo 32	koma finkoa
erreala	float	32	koma higikorra
doitasun bikoitzeko erreala	double	64	koma higikorra

1. taula. Oinarrizko datuen ezaugarriak.

Ezaugarri desberdineko zenbaki osoak defini daitezke zeinuaren erabilpenaren eta bit-kopuru desberdinaren arabera. Horrela, zeinurik gabeko zenbakiak, zenbaki laburrak edo luzeak erazagutu daitezke 2. taulan aztertzen diren ezaugarriak kontutan hartuz.

Mota errealen arteko desberdintasuna doitasunean datza; **float** motakoek 7 zifra hamartar dituzten bitartean **double** motakoek 15 zifrako doitasuna bait dute.

Beste mota batzuk ere erabil daitezke, baina beti aipatutako motak oinarritzat hartuz. Horrela karaktere-kateak karaktereen taulak izango dira, boolearrak bi balio baino erabiltzen ez dituzten osoak edo karaktereak, etab.

Konstanteak

Konstanteak ez dira erabili baino lehen erazagutu behar, zeren eta beren idazkeraren arabera dagokien mota konpiladoreak asmatzen bait du. 3. taulan adibide batzuk ikus daitezke eta bertan ikus daitekeenez • hamartarra da errealen ezaugarria, ‘ ‘ karaktereena, “ “ karaktere-kateena, **L** atzizkia **long** motakoena. Konstanteen adierazpidean aurrizki batzuk ere erabiltzen dira memoriaratu nahi dugun balioaren idazkera zehazteko. **Ø** aurrizkiak adibidez, balioa digitu zortzitarrez osatuta dagoela dio, **Øx** aurrizkiak digitu hamaseitarrez eta **E**k bi zenbakien artean mantisa eta berretzaileaz.

2. taula. Zenbaki osoen azpimotak.

HITZ GAKOA	LABURDURA	BIT-KOPURUA	ADIERAZPIDE-TARTEA
short int	short	16	[-32.768, 32.767]
long int	long	32	$[-2^{31}, 2^{31}-1]$
unsigned short int	unsigned short	16	[0, 65.535]
unsigned long int	unsigned long	32	$[0, 2^{32}-1]$
int	int	16 edo 32*	
usingned int	unsigned	16 edo 32*	

* Ordenadorearen arabera, beraz, ez da zeharo trukagarria

Karaktere bereziak adierazteko badaude beste konstante batzuk (emaitzak idazteko erabiliak batez ere). **\n** adibidez, lerro-bukaerako karakterea da, **\t** tabulaziokoa, **\b** *backspace*koa eta **\Ø** karaktere--katearen bukaerakoa.

ADIBIDEA	ESANAHIA
'a'	a karakterea
25	25 int
25L	25 long
25.	25 float
12.5	12.5 float
12.5EØ	12.5 double
1.15E8	1.25×10^8 double
Ø123	123 zortzitar = 83 int
Øx53	53 hamaseitar = 83 int
"Cikastaroa"	11 karaktereko katea

3. taula. Konstanteen definizioa.

Konstanteen idazkera ikusi ondoren, programazio egituratuak gomendatzen duen erizpide bat gogoratu nahi dugu; konstante parametrizatuak erabiltzearena hain zuzen. Erizpide honen arabera konstanteak zuzenean erabili beharrean (konstanteei hasieran izen bat emanez) erabilpen guztietan izen hori erabiliko da. Horrela, programak irakurgarritasuna irabaziko du eta balioa aldatzekotan izen-ematean baino ez da aldatu behar. Konstanteei izena emateko C programetan **#define** sasiagindua erabiltzen da programen hasieran, konstanteen izenak maiuskulaz idatzi ohi direlarik (ikus 4. taula).

Aldagaien erazagupena eta hasieraketa

4. taulan programan ikus daitekeenez aldagai guztiak, erabili baino lehen, erazagutu egin behar dira, 2. taulan ikusitako hitz gakoak erabiliz dagokien mota

```
# define PI   3.14159
main ( )
{
float r, azalera, perimetroa;
printf ("sakatu erradioaren neurria\n");
scanf ("%f", &r); /*&→erreferentziaz*/
perimetroa = 2 * PI * r;
azalera = PI * r * r;
printf ("azalera: %f\n"; azalera);
printf ("perimetroa: %f\n", perimetroa);
}
```

4. taula. Konstante parametrizatuen erabilpena.

aurretik zehaztuko delarik. Horrela, konpiladoreak aldagaiari programa exekutagarrian zenbat bit gorde beharko dizkion jakingo du.

Aldagaiak erazagutzean hasierako balio bat eslei dakieke, horrela egiten ez bada konpiladoreak **Ø** balioa esleituko dielarik. Hasieraketa hori burutzeko erazagupen-sententzian aldagaiaren izenaren ondoren = karakterea eta ezarri nahi den balioa idatziko du. Adibidez, karaktere motako aldagai bat hasieratzean, a balioaren esleipena ondoko sententziaz egingo dugu:

char datua = 'a'

Bateragarritasuna eta bihurketak

Pascal bezalako lengoaietan ez bezala, C-zko programetan eragigaiek motari dagokionean ez dute bateragarriak izan behar.

Esleipenetan edota eragiketetan eragigaiak mota desberdinekoak badira, datu-bihurketa inplizitua gertatzen da, tarteko aldagaien mota ondoko sailkapenari jarraituz parte hartzen dutenen arteko luzeena eta konplexuena izanik:

char → short → int → long → float → double

Hau dela eta, karaktere bat maiuskula bihurtzeko (minuskulaz dagoela suposatuz) 5. taulako programaren bidez egin daiteke.

Programan ikusten denez, karaktereen batuketa eta kenketa gertatzen da, baina benetan exekutatuko dena ondokoa da: karaktereak zenbaki oso bihurtu, osoen

```
main ( )
{
char kar–min, kar–maj;
printf ("sakatu karaktere bat minuskulaz \n")
scanf ("%c", &kar–min);
kar–maj = kar–min–'a' + 'A'
printf ("%c", kar–maj);
}
```

5. taula. Minuskula maiuskula bihurtzea.

arteko batuketa eta kenketa, eta azkenik emaitza karaktere bihurtu.

Dena den, komenigarria da bihurketak esplizitatzea, erroreak ekidin eta irakurgarritasuna irabazteko, C konpiladore batzuetan esplizituki egiten ez bada abisu-mezuak sortzen direlarik.

Bihurketa esplizitua adierazteko *cast* (izenburu) izeneko elementua erabiltzen da; parentesi artean datu--motaren hitz gakoa zehazten duena. Adibidez, karaktere bat zenbaki oso bihurtzeko **(int)** izenburua jarriko da karaktere-aldagaiaren erreferentziaren aurrean.