Aplicaţii elementare cu caractere şi şiruri de caractere

Tipul `char`

Tipul `char` este un tip întreg, cu semn, care se reprezintă pe un octet. Acest tip suportă un singur modificator – unsigned.

Tip	Valori	Reprezentare
`char`	[-128,127]	1 octet, cu semn
`unsigned char`	[0,255]	1 octet, fără semn

Caracterele grafice se pot reprezenta încadrând caracterul respectiv între apostrofuri.
Exemple: ' a ', ' 0 ', ' * ', ' . '.
Caracterele negrafice (dar şi cele grafice) se pot specifica încadrând între apostrofuri o secvenţă de evitare, numită şi secvenţă escape. Aceste secvenţe sunt formate din caracterul \ urmat de codul ASCII al caracterului negrafic respectiv, exprimat în octal sau hexazecimal. Caracterele negrafice mai des utilizate au asociate secvenţe escape speciale, formate din \ şi o literă sugestivă.
Exemple: ' \n ' – caracterul newline (determină trecerea cursorului la linie nouă), ' \t ' – caracterul tab orizontal, ' \b ' – caracterul backspace (deplasează cursorul pe ecran cu o poziţie la stânga).

Şiruri de caractere

Un şir de caractere este o structură de date care este formată dintr-o mulţime ordonată de caractere, în care fiecare caracter se identifică prin poziţia sa în cadrul mulţimii. Un şir de caractere este, de fapt, o succesiune de caractere.

În limbajul C/C++ şirurile de caractere pot fi implementate ca vectori de caractere.
După cum ştiţi, în general, vectorii au două lungimi: o lungime fizică şi o lungime logică, ceea ce se aplică şi vectorilor de caractere. Ceea ce deosebeşte un vector de caractere de alte tipuri de vectori este posibilitatea de a marca sfârşitul logic al vectorului prin folosirea caracterului **NULL** (care are codul ASCII 0).

Exemplu:

```
char s [256];
```

Am declarat o variabilă s ce reprezintă un şir în care vor putea fi memorate maxim 255 caractere, deoarece un element al vectorului se va folosi pentru a memora caracterul NULL.

Putem iniţializa un şir de caractere la declararea lui (şi în acest caz lungimea şirului poate lipsi) atribuindu-i o constantă de tip şir de caractere (o succesiune de caractere delimitată de ghilimele).
Exemplu:

```
char s [ ] = "Buna ziua";
```

Pentru şirul s declarat compilatorul va rezerva 9 octeţi pentru memorarea constantei şir de caractere şi 1 octet pentru caracterul NULL, deci în total 10 octeţi.

Observaţie:
O constantă de tip şir de caractere ce conţine un singur caracter este diferită de o constantă de tip caracter (prima e stocată pe doi octeţi, a doua doar pe un octet).

Citirea unui şir de caractere

a) În limbajul C++

Fie următoarea declarație:

```
char s [100];
```

Citirea şirului s se poate face utilizând operatorul uzual de citire >> :

```
cin>>s;
```

În acest caz se vor citi în şirul s toate caracterele până la primul caracter alb (spațiu, tab, enter). De exemplu, dacă fluxul de intrare conține caracterele `Buna ziua`, dupa citire, şirul s va fi `Buna`. Pentru a elimina acest dezavantaj se pot folosi funcțiile get() sau getline() (diferența între ele este că getline() extrage din fluxul de intrare caracterul delimitator, în timp ce get() nu îl extrage).
Funcția getline() se poate utiliza cu următoarea formă:

```
cin.getline(sir,nr,ch)
```

unde **sir** este şirul de caractere în care se citeşte, **nr** reprezintă numărul maxim de caractere care se pot citi, iar **ch** este caracterul delimitator (este opțional, implicit este ' \n ').
Efectul este următorul: se citesc de la tastatură în variabila sir mai multe caractere, inclusiv caractere albe, până au fost citite maxim nr-1 caractere sau până a fost citit caracterul delimitator ch.

Exemple:

```
cin.getline(s,100);  //  al treilea parametru lipseşte, '\n' este delimitator

cin.getline(s,100,'.'); // se citesc de la tastatură toate caracterele până la
întâlnirea caracterului '.'
```

Observație:
La citire, marcajul de sfârşit de şir NULL este automat plasat la sfârşitul şirului.

b) În limbajul C

Pentru a citi şiruri ce nu conțin caractere albe se poate folosi funcția scanf():

```
scanf("%s",s);
```

Pentru citirea şirurilor care conțin caractere albe se poate utiliza funcția gets(), declarată în fişierul antet stdio.h, care citeşte în şirul s caracterele introduse de la tastatură până la întâlnirea caracterului ' \n ' (care este citit şi convertit automat în caracterul NULL):

```
gets(s);
```

Afişarea unui şir de caractere

a) În limbajul C++

Se face cu ajutorul operatorului de scriere <<, caracterele fiind afişate până la întâlnirea marcajului de sfârşit de şir (NULL):

```
cout<<s;
```

b) În limbajul C

Afişarea se poate face cu ajutorul funcției printf(), utilizând specificatorul de format %s :

```
printf("%s",s);
```

Afişarea se poate face şi cu funcția puts(), cu prototipul în stdio.h, care afişează caracterele şirului s până la întâlnirea caracterului NULL şi apoi afişează un caracter newline:

```
puts(s);
```

Şirurile de caractere **pot fi prelucrate la nivel de caracter** (pot fi parcurse caracter cu caracter, ca un vector de caractere), sau **pot fi prelucrate la nivel de structură** (cu ajutorul funcţiilor existente în bibliotecile limbajului).

Funcţii standard de lucru cu şiruri de caractere

Chiar dacă folosirea funcţiilor standard de lucru cu şiruri de caractere presupune cunoaşterea tipului pointer, iată totuşi exemple de apeluri pentru câteva dintre cele mai utilizate funcţii (declarate în fişierul antet string.h, respectiv cstring):

`strlen(s)` - returnează lungimea şirului s

`strcpy(destinatie,sursa)` - copie şirul sursa, caracter cu caracter, în şirul destinatie, până după copierea caracterului NULL şi returnează adresa de început a şirului destinatie.

`strcat(destinatie, sursa)` - copie caracterele şirului sursa la sfârşitul şirului destinatie şi returnează adresa de început a şirului destinatie.

`strcmp(s1,s2)` - compară din punct de vedere **lexicografic** cele două şiruri şi returnează 0 dacă cele două şiruri sunt egale, o valoare negativă dacă s1<s2 sau o valoare pozitivă dacă s1>s2.

`strlwr(s)` - transformă toate literele mari din şirul s în litere mici.

`strupr(s)` - transformă toate literele mici din şirul s în litere mari.

`tolower(ch)` - transformă caracterul ch din literă mare în literă mică, altfel îl lasă neschimbat

`toupper(ch)` - transformă caracterul ch din literă mică în literă mare, altfel îl lasă neschimbat

Bibliografie

- E. Cerchez, M. Serban - Programarea în limbajul C/C++. Volumul I. Editura Polirom
- E. Cerchez –Informatica, Culegere de probleme pentru liceu. Editura Polirom
- M. Miloşescu – Informatică – manual pentru clasa a X-a. Editura didactică şi pedagogică
- D.Lica, M.Paşoi – Informatică, Culegere de probleme pentru clasa a X-a. Editura L&S Infomat

PROBLEME PROPUSE

1. Se citeşte de la tastatură o succesiune de litere ale alfabetului englez. Citirea se încheie la apăsarea tastei Enter. Să se afişeze frecvența de apariție a literelor în text, fără a se ține cont de diferența dintre literele mari şi literele mici.

2. Se citeşte de la tastatură o secvență formată din maxim 100 de litere mici, până la întâlnirea caracterului punct. Dacă există în secvența citită litere alăturate egale, aceste litere vor fi eliminate. Dacă în urma eliminării se obțin din nou litere alăturate egale, se elimină şi acestea, eliminările efectuându-se până când în secvență nu există litere alăturate egale. Să se afişeze caracterele rămase după ce s-au executat toate aceste operații de eliminare.
Exemplu:
Dacă se citeşte secvența **aacbooobcaxa**, se va afişa: **axa**

3. Se consideră fişierul text **date.in** ce conține numere întregi dispuse pe mai multe linii. Orice caracter ce nu reprezintă un caracter numeric este considerat separator. Scrieți un program care creează un fişier **date.out** ce conține pe fiecare linie media aritmetică a numerelor situate pe aceeaşi linie în fişierul date.in. Media aritmetică va fi scrisă cu două zecimale. Pe fiecare linie a fişierului de intrare se află maxim 200 de caractere.
Exemplu:

date.in	date.out
2..3a 403bx 2..2 A,..5 1.92	136.00 3.00 46.50

4. grad
http://campion.edu.ro/arhiva/index.php?page=problem&action=view&id=143

5. alfabetar
http://campion.edu.ro/arhiva/index.php?page=problem&action=view&id=1200

6. paritate
http://campion.edu.ro/arhiva/index.php?page=problem&action=view&id=843