Laboratorio n. 5

28 maggio e 4 giugno 2004

Conversione in maiuscole

- Scrivere un programma che legge da tastiera una stringa, ne converte tutti i caratteri in maiuscolo e stampa il risultato
- Sfruttare la funzione toupper(char c) che converte in maiuscolo un singolo carattere
- Fornire **due** soluzioni distinte:
  1. Manipolare la stringa come un array
  2. Manipolare la stringa usando solo puntatori.

Conteggio vocali

Scrivere un programma con:

- Una funzione Vocale che dato un carattere stabilisce se è una vocale (ritorna 1) o una consonante (ritorna 0)
- Una funzione ContaVocali che data una stringa conta e ritorna quante vocali contiene, usando la funzione Vocale.
  - Realizzare **due distinte implementazioni** della funzione ContaVocali:
  1. "Classica", iterativa
  2. Ricorsiva (*il numero di vocali contenute in una stringa è pari al numero di vocali contenute nella sottostringa dal secondo carattere in poi, più 1 se il primo carattere è una vocale…)*
- Una funzione main che legge da tastiera una stringa e la passa alla funzione ContaVocali per conoscere il numero di vocali contenute nella stringa, poi stampa il numero ritornato da tale funzione.

Conteggio vocali: possibile soluzione (iterativa)

```
int Vocale(char c)
  { c=tolower(c);
    if ((c=='a') || (c=='e') ||\
     (c=='i') || (c=='o') || (c=='u'))
       return(1);    else return(0);  }
int ContaVocali(char *s)
  { int ret=0, i;
    for(i=0;s[i]!='\0';i++)
      if (Vocale(s[i]))
            ret++;
    return ret;}
main()
  {char s[30];
   printf("dammi una stringa\n");
   gets(s);
   printf("vocali %d\n",ContaVocali(s));return 0;}
```

Conteggio vocali: variante per soluzione ricorsiva

```
int ContaVocali(char *s)
  { int ret=0, i;
    for(i=0;s[i]!='\0';i++)
      if (Vocale(s[i]))
            ret++;
    return ret;
  }
```

```
int ContaVocali(char *s)
  { if(!*s)return 0;
      else return Vocale(*s)+ContaVocali(s+1);
  }
```

Strutture e array (1)

Dato l'assegnamento C

```
x[2].a = x.b;
```

indicare i tipi, se esistono, che lo rendono possibile; altrimenti spiegare perché non esistono.

Strutture e array (2)

Dato l'assegnamento C

```
y.a[2] = y.a;
```

indicare i tipi, se esistono, che lo rendono possibile; altrimenti spiegare perché non esistono.

Puntatori: trova errore

- Correggere il programma e stabilire il valore finale delle variabili

```
main()
   {   int a, b, *c;
       a=8; b=9;
       c=&a;
       *c=a+5;
       b=c;
   }
```

Funzioni e passaggio parametri

- Quanto varranno i parametri nella funzione chiamante dopo l'esecuzione della funzione?

```
void f1 (int a, int b)
   {  int c;
      a = a + 4;
      c = a + b;
      b = b - 3;  }
```

Parametri: trova errore

- La funzione dovrebbe scambiare due valori (ma non lo fa)

```
void scambia(int *a, int *b)
   {
      int *temp;
      temp = a;
      a = b;
      b = temp;
   }
```

Parametri: trova errore

- Trovare l'errore :

```
#define N 12
#define M 4
void carica (int *v)
  {  int i;
     for(i=0; i<M; i++)
      *(v+i)=i;}
main()
  { int v[N];
    carica(&v);    carica(&v[4]);
  }
```

Parametri: trova errore

- Trovare l'errore

```
#define N 20
typedef int vett[N];
void carica (vett v)
  { int i;
    for(i=0; i<N; i++)
       scanf("%d",v[i]);}
main()
  {vett v;
     carica(v);
  }
```

Parametri: trova errore

- Trovare l'errore logico e correggerlo :

```
typedef int VT[8];
int *carica (void)
  { int i;    VT v;
    for(i=0;i<8; i++)
      v[i] = i, printf("%d\n",v[i]);
    return(v);  }
main()
  { int *v,i;
    v=carica();    printf("\n");
  for(i=0;i<8;i++) printf("%d\n",v[i]);
}
```

Parametri: trova errore

- Trovare gli errori :

```
#define M 4
typedef struct{int a; char b[9];} RC;
void carica (RC *r)
  {r.a=12;
   strcpy(r->b,"prova");  }
main()
  { int i; RC vr[M];
    for(i=0;i<M;i++)
     { carica(vr[i]);
  printf("%d\t%s\n",vr[i].a,vr[i].b);}}
```

Parametri: trova errore

- Trovare gli errori :

```
typedef struct{int a; int b;} RC;
RC carica (void)
  {  RC r;
     r->a=12;      r->b=20;
     return(r);  }
main()
  { int i;    RC rec;
    rec=carica();
    printf("%d\t%d\n",rec.a,rec.b);  }
```

Operazioni su vettori

Scrivere un programma che:

- Dichiara un vettore di 10 numeri interi
- Carica tale vettore con 10 valori letti da tastiera
- Verifica se il vettore contiene (in qualsiasi posizione) almeno 2 valori uguali e lo segnala dicendo qual è il valore ripetuto e in quali elementi del vettore si trova.

Possibile soluzione (frammento)

```
int trova (vett A, int ret)
{ int i,j;
 for(i=0; (i<99) && (ret==0); i++)
  for(j=0; j<100; j++)
     if (A[i] == A[j])
                ret=1;
 return(ret); }
```