Laboratorio di Elementi di Architetture e Sistemi Operativi

Esercizi del 2 Maggio 2012

Esercizio 1. *Scrivere un programma che dichiari le seguenti variabili ed esegua le operazioni specificate di seguito:*

```
double tassointeresse = 7.3, tassosconto;
double *dblptr;
```

1. *Stampa il valore di* `tassointeresse` *sullo schermo;*
2. *Assegna l'indirizzo della variabile* `tassointeresse` *al puntatore* `dblptr`*;*
3. *Stampa il valore puntato da* `dblptr` *sullo schermo;*
4. *Assegna il valore puntato da* `dblptr` *alla variabile* `tassosconto`*;*
5. *Stampa il valore di* `tassosconto` *sullo schermo;*
6. *Stampa l'indirizzo di* `tassointeresse` *sullo schermo.*
7. *Stampa l'indirizzo di* `tassosconto` *sullo schermo.*
8. *Stampa l'indirizzo memorizzato in* `dblptr` *sullo schermo.*
9. *Il valore stampato è lo stesso dell'indirizzo di* `tassointeresse`*? Oppure è lo stesso dell'indirizzo di* `tassosconto`*?*

Per stampare un indirizzo di memoria usare la direttiva `%p` *di* `printf`*.*

```
#include<stdio.h>
#include<stdlib.h>

int main(int argc, char* argv[]){
  double tassointeresse = 7.3, tassosconto;
  double *dblptr;

  printf("valore di tassointeresse: %f\n", tassointeresse);
  dblptr = &tassointeresse;
  printf("valore puntato da dblptr: %f\n", *dblptr);
  tassosconto = *dblptr;
  printf("valore di tassosconto: %f\n", tassosconto);
  printf("indirizzo di tassointeresse: %p\n", &tassointeresse);
  printf("indirizzo di tassosconto: %p\n", &tassosconto);
  printf("indirizzo contenuto in dblptr: %p\n", dblptr);
}
```

L'indirizzo contenuto in `dblptr` è uguale all'indirizzo di `tassointeresse`.

Esercizio 2. *Scrivete una funzione con prototipo*

`void split-time ( long int sec, int *h, int *m, int *s )`

che, dato un orario fornito in numero di secondi dalla mezzanotte (`00:00:00`*), calcoli l'orario equivalente in ore, minuti, secondi, e lo memorizzi nelle tre variabili puntate da* `h`*,* `m` *e* `s` *rispettivamente. Scrivere un programma di esempio che usi la funzione.*

```
#include <stdio.h>

void splittime ( long int sec, int *h, int *m, int *s )
{
  *s = sec % 60;
  sec /= 60;
  *m = sec % 60;
```

```
  *h = sec / 60;
}

main()
{
  int sec, h, m, s;

  printf("Inserire l'ora in secondi dalla mezzanotte: ");
  scanf("%d", &sec);
  splittime(sec, &h, &m, &s);
  printf("L'ora esatta è %2d:%2d:%2d\n",h,m,s);
}
```

Esercizio 3. *Scrivete una funzione con prototipo*
`int *smallest( int a[], int n )`
che, dato un vettore `a` *di lunghezza* `n`*, restituisca un puntatore all'elemento più piccolo dell'array. Scrivere un programma di esempio che usi la funzione.*

```
#include <stdio.h>

int *smallest( int a[], int n )
{
  int *min = &a[0];
  int i;

  for(i = 1; i < n; i++) {
    if(a[i] < *min)
      min = &a[i];
  }
  return min;
}

main()
{
  int v[] = {4, 3, 4, 5, -3, 4, -3, 1};
  int *min;

  min = smallest(v, 8);
  printf("Il minimo di v è in posizione %d, ed ha valore %d\n", (min - v), *min);
}
```

Esercizio 4. *Una parola* palindroma *è una parola che, letta a rovescio, rimane identica. Scrivete una funzione che stabilisca se il suo argomento è una parola palindroma oppure no, usando due puntatori per scorrere la parola partendo dall'inizio e dalla fine, e* senza usare le funzioni di libreria per le stringhe*. Quindi scrivete un programma che stabilisca, per ciascun argomento* fornito da linea di comando*, se si tratta di una parola palindroma oppure no.*

```
#include <stdio.h>

int palindroma(char *s) {
  char *t;
  for(t = s; *t != '\0'; t++) ;
  for(t--; s < t && *s == *t; t--, s++)
    ;
  return !(s < t);
}
```

```
int main(int argc, char *argv[])
{
  int i;
  if(argc == 1) {
    fprintf(stderr, "Uso: %s parola parola ....\n", argv[0]);
    return 1;
  }
  for(i = 1; i < argc; i++) {
    if(palindroma(argv[i])) {
      printf("%s è palindroma\n", argv[i]);
    } else {
      printf("%s non è palindroma\n", argv[i]);
    }
  }
  return 0;
}
```

Esercizio 5. *Scrivere le seguenti funzioni, utilizzando i puntatori e* senza usare le funzioni di libreria per le stringhe*:*

- *`void concat(char *s, char *t)` che aggiunge la stringa `t` al termine della stringa `s` (supponete che la lunghezza ddi `s` sia sufficiente a contenere la concatenazione di `s` e `t`);*
- *`void reverse(char *s)` che inverte la stringa `s` (e.g. `abc` $\rightsquigarrow$ `cba`);*

Utilizzate le funzioni `strcat` e `reverse` per scrivere i seguenti comandi:

1. *un comando `uniscipar` che unisce in una riga sola tutti i paragrafi di un testo, eliminando le linee vuote. Un paragrafo è un blocco di righe di testo separato dagli altri da una o più linee vuote;*
2. *un comando `inverti` che scrive alla rovescia le righe di un testo.*

Ogni comando deve leggere l'input da un file passato come primo argomento *del `main`, e stampare il risultato sullo schermo. Si supponga che le linee del testo di output non siano mai più lunghe di 100 caratteri.*

Comando `uniscipar`:

```
#include <stdio.h>

void concat(char *s, char *t) {
  for( ; *s != '\0'; s++) ;
  for( ; *t != '\0'; t++, s++) {
    *s = *t;
  }
  *s = '\0';
}

void togliacapo(char *s) {
  for( ; *s != '\0' && *s != '\n'; s++) ;
  *s = '\0';
}

int main(int argc, char *argv[])
{
  char s[100] = "";
  char t[100];
  char *res;
  FILE *input;

  if(argc != 2) {
    fprintf(stderr, "Uso: uniscipar nomefile\n");
```

```
    return 1;
  }
  input = fopen(argv[1], "r");
  if(input == NULL) {
    fprintf(stderr, "errore nell'apertura del file %s\n", argv[1]);
    return 1;
  }
  res = fgets(t,100,input);
  while(res != NULL) {
    if(t[0] != '\n') {
      togliacapo(t);
      concat(s, t);
      concat(s, " ");
    } else if(s[0] != '\0') {
      puts(s);
      s[0] = '\0';
    }
    res = fgets(t,100,input);
  }
  if(fclose(input) != 0) {
    fprintf(stderr, "Errore nella chiusura del file %s\n", argv[1]);
    return 1;
  }
  return 0;
}
```

Comando `inverti`:

```
#include <stdio.h>

void swap(char *a, char *b) {
  char tmp;
  tmp = *a;
  *a = *b;
  *b = tmp;
}

void reverse(char *s) {
  char *t;
  for(t = s; *t != '\0'; t++) ;
  for(t--; s < t; t--, s++) {
    swap(s, t);
  }
}

void togliacapo(char *s) {
  for( ; *s != '\0' && *s != '\n'; s++) ;
  *s = '\0';
}

int main(int argc, char *argv[])
{
  char s[100];
  char *res;
  FILE *input;
```

```
  if(argc != 2) {
    fprintf(stderr, "Uso: inverti nomefile\n");
    return 1;
  }
  input = fopen(argv[1], "r");
  if(input == NULL) {
    fprintf(stderr, "errore nell'apertura del file %s\n", argv[1]);
    return 1;
  }
  res = fgets(s,100,input);
  while(res != NULL) {
    togliacapo(s);
    reverse(s);
    puts(s);
    res = fgets(s,100,input);
  }
  if(fclose(input) != 0) {
    fprintf(stderr, "Errore nella chiusura del file %s\n", argv[1]);
    return 1;
  }
  return 0;
}
```