Politecnico di Milano - Anno Accademico 2003-2004 - Informatica C Seconda prova in itinere – 5 Luglio 2004 – **Elaborato A**	
COGNOME e NOME	
Matricola	

DOMANDA SU TEMI DI LABORATORIO

Si supponga che il seguente programma venga scritto esattamente come sotto riportato e che si cerchi poi di compilarlo e di eseguirlo. Stabilire quanti e quali errori contiene e indicarli, dopodichè, per ogni errore trovato:

- indicare come lo si potrebbe correggere.
- Specificare se si tratta di un errore che il compilatore rileverebbe non appena si tentasse di compilare il programma, oppure di un errore che si manifesterebbe successivamente, in fase di esecuzione.

```
int main()
{

  printf("ciao/n");            /n invece di \n (run time)

  int a=2;                     le dichiarazioni vanno all'inizio
                               (compile time)
  printf("a vale: %d",a);

}                              la funzione dovrebbe ritornare un int
                               (compile time)
```

Politecnico di Milano - Anno Accademico 2003-2004 - Informatica C Seconda prova in itinere – 5 Luglio 2004 – **Elaborato A**	
COGNOME e NOME	
Matricola	

Domande su argomenti della II parte del corso

1. Files (punti 5)

E' stato richiesto di scrivere un programma che copia il contenuto di un file su un altro file, con le seguenti caratteristiche:

- I nomi dei due file devono essere forniti su riga di comando; prima il nome del file di origine e poi quello del file di destinazione.
- Se non vengono dati esattamente due parametri il programma termina subito. NON è richiesto stampare un messaggio d'errore.
- Se il file di origine non esiste o non può essere aperto, per qualsiasi motivo, il programma termina subito. NON è richiesto stampare un messaggio d'errore.
- Se il file di destinazione non esiste, viene creato; se esiste già viene sovrascritto quello che c'era, senza avvisare con alcun messaggio.
- Se fallisce per qualsiasi motivo la creazione del file di destinazione il programma termina subito. NON è richiesto stampare un messaggio d'errore.

Si assuma che la parte restante del programma, quella che esegue il ciclo di copiatura, sia già stata scritta.

Completare il programma sotto riportato in modo tale che si comporti esattamente come sopra descritto. E' ammesso solo fare delle aggiunte e solo nei punti del programma ove compaiono gli appositi riquadri; le parti fuori dai riquadri non possono assolutamente essere modificate.

```
#include <stdio.h>

int main(          int argc, char * argv[]          )
{
  FILE * infile;
  FILE * outfile;
  int count=1;
  char c;

  if(argc!=3)
    return 0;

  infile=fopen(argv[1],"r");
  if(infile == NULL)
    return 0;

  outfile=fopen(argv[2],"w");
  if(outfile == NULL)
    return 0;

  /* .. la parte finale del programma, comprendente il ciclo di copiatura, è già
stata scritta .. */
}
```

Dich. param. main() – punti 0.5

Controllo n. parametri – punti 1

Apertura file origine – punti 1.5

Apertura file destinazione – punti 2

Politecnico di Milano - Anno Accademico 2003-2004 - Informatica C Seconda prova in itinere – 5 Luglio 2004 – **Elaborato A**	
COGNOME e NOME	
Matricola	

2. Liste (punti 5)

Un archivio di dati personali utilizza come struttura dati una lista doppia che contiene attualmente 4 nodi. Ogni nodo contiene nome, cognome, indirizzo ed età di una persona. Il puntatore alla testa della lista è la variabile globale **`head`**, il puntatore alla coda della lista è la variabile globale **`tail`**.

I nodi sono di un tipo così definito:

```
struct nodo_s {
      char nome[30];
      char cognome[30];
      char indirizzo[50];
      int eta;
      struct nodo_s * next;
      struct nodo_s * prev;
};
```

Percorrendola dalla testa alla coda, la lista contiene i dati delle quattro persone i cui cognomi sono Bianchi, Gialli, Neri e Rossi (in quest'ordine).

Data una variabile x di tipo int e una variabile y di tipo char,

- **scrivere un'istruzione che assegna a x il valore dell'età di Neri. Non usare il puntatore head. (scandire la lista dalla coda) –** ***punti 1***

```
x=tail->prev->eta;
```

- **scrivere un'istruzione che assegna a x la lunghezza della stringa contenente l'indirizzo di Rossi. Non usare il puntatore tail. (scandire la lista dalla testa) –** ***punti 1***

```
x=strlen(head->next->next->next->indirizzo);
```

- **scrivere un'istruzione che assegna a y il terzo carattere del nome di Bianchi. (si assuma che tale nome sia sicuramente lungo almeno 3 caratteri) –** ***punti 1***

```
y=head->nome[2];
```

- **supponendo che sia già stato creato (allocato e inizializzato) un nodo per il sig. Arrigoni, raggiungibile mediante il puntatore `nuovonodo`, scrivere le istruzioni necessarie per inserire tale nodo in prima posizione nella lista. –** ***punti 2***

```
head->prev=nuovonodo;
nuovonodo->next=head;
head=nuovonodo;
nuovonodo->prev=NULL;
```

Politecnico di Milano - Anno Accademico 2003-2004 - Informatica C Seconda prova in itinere – 5 Luglio 2004 – **Elaborato A**	
COGNOME e NOME	
Matricola	

3. Algoritmi (punti 3)

Scrivere una funzione **`int cercaA(char * stringa)`** che prende una stringa come parametro e ritorna 1 se la stringa contiene almeno una volta il carattere A, altrimenti ritorna 0.
Implementare la stessa funzione in due modi diversi, utilizzando prima un algoritmo iterativo (punti 1) e poi uno ricorsivo (punti 2).
Suggerimento per l'implementazione ricorsiva: *data una stringa di N (N>0) caratteri, esaminare il primo carattere: se è A, si può senz'altro ritornare 1. Altrimenti controllare se A è presente nel resto della stringa e ritornare il risultato di tale controllo…*

Implementazione iterativa

```
int cercaA_iterativa(char * stringa)
{
    int i, count=0;
    for(i=0; stringa[i]!='\0'; i++)
      if(stringa[i]=='A')return 1;
    return 0;
}
```

Implementazione ricorsiva

```
int cercaA_ricorsiva(char * stringa)
{
    if(stringa[0]=='\0')return 0;
    else if(stringa[0]=='A') return 1;
    else return cercaA_ricorsiva(stringa+1);
}
```

4. Puntatori (punti 2)

La variabile **`buffer`** è stata dichiarata come un array di caratteri (destinato a contenere una stringa). Scrivere un ciclo for in cui la stringa viene esaminata e, qualora vengano trovate delle "A", queste vengono trasformate in "Z"; scrivere il ciclo SENZA FARE USO dell'operatore [] per l'accesso all'array e SENZA FARE USO della funzione strlen.

```
for(i=0; *(buffer+i)!=0; i++)
    if(*(buffer+i) == 'A') *(buffer+i) = 'Z';
```

5. Teoria (punti 2)

spiegare che cosa si intende con l'espressione "dangling reference"

Se un puntatore punta ad un'area di memoria che ha un tempo di vita diverso dal suo si ha una dangling reference.
Es. un puntatore in un main punta ad un dato dichiarato in una funzione, quando la funzione termina l'indirizzo non vale più. Oppure se un puntatore punta ad una memoria allocata dinamicamente (malloc), quando viene eseguita un FREE dell'area, l'indirizzo contenuto nel puntatore non ha più senso.

```
Int * P,Q;  P=malloc ……ecc …; Q=P; free(P); /*Q non punta a nulla */
```