Laboratorio di Informatica L-A
Prova d'Esame 5– 27 Giugno 2007

Prima di cominciare: si scarichi il file **`StartKit5.zip`** contenente i file necessari.
Avvertenze per la consegna: nominare i file sorgenti come richiesto nel il testo del compito, apporre all'inizio di ogni file sorgente un commento contenente i propri dati (**cognome, nome, numero di matricola**) e il **numero** della prova d'esame. Al termine, **consegnare tutti i file sorgenti** ed i file contenuti nello StartKit.
Rispettare **ALLA LETTERA** le specifiche, in particolare inserire le funzioni nei file specificati fra parentesi dopo il nome della funzione. Chi non rispetta le specifiche sarà opportunamente penalizzato. **NON SARANNO CORRETTI** gli elaborati che presenteranno un numero "non affrontabile" di errori di compilazione.
Consiglio: per verificare l'assenza di *warnings*, effettuare di tanto in tanto un *Rebuild All*.

La società Autostrade, commissiona alla vostra azienda la progettazione di un software per il controllo elettronico della velocità. Sulla strada sono disposti dei varchi a distanze note; quanto un autoveicolo passa sotto un varco, viene annotato il passaggio nella forma **`targa;ora`** (es: **`XY123YZ;12:23:44`**) in un file di testo il cui nome è dato da una combinazione fra il nome del varco e un numero progressivo (un contatore) che si resetta al passaggio del giorno. Ad esempio il varco di nome **`V1`** genererà, nell'ordine, i file **`V1-1.txt`**, **`V1-2.txt`**, ecc. L'applicazione da progettare deve, per ogni coppia di varchi per cui è definita una distanza, caricare tutte le informazioni di passaggio di ognuno dei due varchi (quindi deve caricare tutte le informazioni su tutti i file presenti) e, tramite le informazioni caricate, calcolare le velocità medie di percorrenza dei veicoli per poi stampare tali velocità sullo schermo.
Nello **`StartKit5.zip`** vengono forniti i file di testo relativi alle informazioni di passaggio, il file **`gatePair.h`** contenente la definizione della coppia di varchi (**`GatePair`**), il file **`configuration.h`** contenente la configurazione relativa alle coppie di varchi da considerare (**`GatePairArray`**), i file **`tests.h/tests.c`** (completamente commentati e da decommentare selettivamente) contenenti alcune funzioni di test che consentono di verificare il buon funzionamento del codice scritto.

NOTA BENE:

- **Affiché i test funzionino è assolutamente necessario utilizzare i nomi suggeriti nel testo.**
- **La non deallocazione della memoria allocata dinamicamente non sarà considerata errore, ma una corretta deallocazione potrà generare punti "bonus".**

Passo 1 – Tipo Time e alcune funzioni correlate (time.h/time.c)

Definire tramite **`typedef`** Il tipo **`Time`** (**`time.h`**) come una struttura di tre **`short`** (**`hour`**, **`minute`**, **`second`**) che rappresenta un'ora della giornata. Definire una funzione **`stringToTime`** (**`time.h/time.c`**) che presa in ingresso una stringa di caratteri rappresentante un'ora (tipo hh:mm:ss), restituisca il **`Time`** corrispondente. Definire una funzione **`timeDifference`** (**`time.h/time.c`**) che date due strutture **`Time t1`** e **`t2`**, restituisca la differenza fra **`t1`** e **`t2`** (**`t1`** - **`t2`**) in ore come **`float`**. Per fare ciò si consiglia di trasformare **`t1`** e **`t2`** in due **`float`** rappresentanti le ore (es: 9:30:00 → 9,5 ore) per poi effettuare la differenza. Si ponga attenzione al fatto che se **`t1`** < **`t2`** occorre aggiungere a **`t1`** 24 ore per far tornare i calcoli. Una volta terminata l'implementazione, si decommentino dai file di test le funzioni **`test_stringToTime`** e **`test_timeDifference`** e si eseguano (nel **`main`**) per verificare che l'implementazione sia corretta.

Passo 2 – PlateTime e alcune funzioni correlate (plateTime.h/plateTime.c)

Definire (**`typedef`**) il tipo **`Plate`** (targa) come una stringa di al più 19 caratteri. Definire (**`typedef`**) il tipo **`PlateTime`** come una struttura contenente un campo **`Plate`** (di nome **`plate`**) e un campo **`Time`** (di nome **`time`**). Definire una struttura di nome **`PlateTimeListStruct`** che rappresenti un elemento di una lista contenente un campo di tipo **`PlateTime`** (di nome **`data`**); il puntatore al successivo elemento della lista deve chiamarsi **`next`**. Definire (**`typedef`**) il tipo **`PlateTimeList`** come puntatore ad una struttura **`PlateTimeListStruct`**. Definire la funzione **`addPlateTime`** che presi in ingresso una lista **`PlateTimeList`** e un elemento **`PlateTime`**, aggiunga alla lista l'elemento ricevuto e restituisca la nuova lista. Definire la funzione **`concatPlateTimeLists`** che prese in ingresso due liste **`PlateTimeList`** le concateni e restituisca il risultato della concatenazione nella forma di una nuova **`PlateTimeList`**. Definire la funzione **`getPlateTimeListLength`** che presa in ingresso una lista **`PlateTimeList`** calcoli e restituisca la lunghezza di tale lista. Definire la funzione **`findPlateTime`** che data una lista **`PlateTimeList`** e dato un

Plate, cerchi nella lista l'elemento **PlateTime** avente nel campo **plate** il valore ricevuto; nel caso la ricerca abbia successo, la funzione deve restituire un puntatore al PlateTime trovato, altrimenti deve restituire **NULL**. Terminata l'implementazione delle funzioni di cui sopra, decommentare le rimanenti funzioni di test ed invocarle per verificare il corretto funzionamento del programma.

Passo 3 – Lettura dei file (plateTime.h/plateTime.c)

Definire la funzione **readPlateTimeFile**; tale funzione deve prendere in ingresso un file di testo già aperto (uno dei file relativi ai passaggi sotto i varchi) e restituire una **PlateTimeList**. Per convertire il campo relativo l'ora del passaggio in un **Time** si usi la funzione **stringToTime** precedentemente definita. Si verifichi il corretto funzionamento aprendo e leggendo il file **V1-1.txt**. Definire la funzione **readGateFiles** che, dato in ingresso un parametro di tipo **GateName** (v. **gatePair.h**), legga tutti i file relativi ad un varco (ad es., per il varco **V1** i file **V1-1.txt**, **V1-2.txt**, **V1-3.txt**, …) e restituisca la **PlateTimeList** relativa. A questo scopo occorre predisporre un indice che parta da 1, costruire il nome del file partendo dal nome del varco e dall'indice stesso e tentare di aprire il file: se l'apertura ha successo, si legga il file (**readPlateTimeFile**) e si concateni la lista letta alla lista che si sta costruendo **concatPlateTimeLists**; se l'apertura del file non ha successo si termini la lettura e si restituisca la lista "globale". A titolo di verifica, nel **main** si stampino a video le informazioni lette.

Passo 4 – Calcolo e stampa delle velocità (plateVelocity.h/plateVelocity.c)

Definire un tipo struttura di nome **PlateVelocity** che contenga una targa (**Plate**) e una velocità (un campo **float**). Definire una funzione **calculateVelocities** che prenda in ingresso una **GatePair**, due liste **PlateTimeList** (una per ogni varco) e restituisca un *array* di **PlateVelocity** allocato dinamicamente e grande quanto la più piccola delle liste **PlateTimeList** ricevute in ingresso – poiché non è detto che tutte le automobili passate sotto un varco passino sotto quello successivo (errori di lettura, fermate degli autoveicoli, ecc.) il numero di strutture effettivamente presenti nell'*array* di **PlateVelocity** può essere inferiore alla sua dimensione fisica, pertanto è necessario restituire anche il numero di elementi effettivamente presenti nell'*array*. Per calcolare le velocità, per ogni elemento della prima lista **PlateTimeList**, si cerchi nella seconda lista un elemento **PlateTime** con lo stesso valore di **Plate** (usare la funzione **findPlateTime** scritta in precedenza); se tale elemento viene trovato, procedere al calcolo della velocità ed inserire un nuovo elemento con tutti i dati del caso nell'*array*. Definire la funzione **printVelocities** che presi in ingresso una **GatePair**, un *array* di **PlateVelocity** e il numero di elementi effettivamente contenuti nell'*array*, stampi a video i nomi dei varchi interessati e la loro distanza e, per ogni autovettura di cui è stata calcolata la velocità, la targa e la velocità stessa. Nel **main** si effettui un ciclo sul **GatePairArray** definito nel file **configuration.h**, si leggano i file relativi ad ognuno dei due varchi contenuti in ogni **GatePair** e si calcolino e stampino le velocità calcolate. Il risultato della stampa ottenuta con i dati forniti è contenuto nel file **result.txt**.