Esercizio 1

La compagnia telefonica TT, utilizza le seguenti strutture dati per memorizzare i dati delle chiamate effettuate dai propri 100 clienti nell'ultimo mese:

```
#define MAX 1000

typedef char stringa[50];

typedef struct
{
     int ora;     // ora di inizio della chiamata telefonica (da 0 a 23)
     int minuti; // minuti di inizio della chiamata telefonica (da 0 a 59)
     int durata; // durata della chiamata telefonica in secondi
} chiamata;

typedef struct
{
     stringa CF;       //codice fiscale del cliente
     int n;            //numero chiamate effettuate dal cliente nell'ultimo mese
     chiamata c[MAX]; //chiamate effettuate dal cliente nell'ultimo mese
} cliente;

// database dei 100 clienti della compagnia
cliente db[100];
```

1) Le telefonate iniziate dalle 22:00 alle 07:59 (estremi inclusi) hanno un costo di 0.005 euro al secondo e le altre di 0.01 euro al secondo. Dichiarare un array di 100 **float** di nome **bolletta** e scrivere un frammento di codice C che riempia l'array **bolletta**, in modo tale che la posizione i-esima di bolletta contenga il costo complessivo delle chiamate effettuate dal cliente i-esimo nell'ultimo mese.

2) Dichiarare un array di nome **premium** di tipo **cliente** e riempirlo (senza lasciare spazi vuoti) con i dati dei clienti che hanno speso più di 100 euro nel corso dell'ultimo mese.

Note. Si ipotizzi che la variabile **db** sia già stata inizializzata con i dati dei clienti della compagnia. Rispondere ai punti 1) e 2) riportando solo i frammenti di codice necessari a svolgere le operazioni richieste e le eventuali dichiarazioni di variabili aggiuntive utilizzate.

Esercizio 3 (3 punti). Le seguenti dichiarazioni definiscono tipi di dati relativi alla categoria degli impiegati di un'azienda (gli impiegati possono essere di prima, seconda, …, quinta categoria), agli uffici occupati da tali impiegati, all'edificio che ospita tali uffici (l'edificio è diviso in 20 piani ognuno con 40 uffici).

```
/* definizioni dei tipi */
typedef struct { char nome[20], cognome[20];
                 int cat;  /* contiene valori compresi tra 1 e 5 */
                 int stipendio;
               } impiegato;
typedef enum {nord, nordEst, est, sudEst, sud, sudOvest, ovest, nordOvest} esposizione;
typedef struct { int superficie;  /* in metri quadri */
                 esposizione esp;
                 impiegato occupante;
               } ufficio;

/* definizioni delle variabili */
ufficio torre[20][40]; /* rappresenta un edificio di 20 piani con 40 uffici per piano */
```

Si scriva un frammento di codice, che includa eventualmente anche le dichiarazioni di ulteriori variabili, che, per tutte e sole le persone che occupano un ufficio (tra quelli memorizzati nella variabile torre) orientato a sud oppure a sudEst e avente una superficie compresa tra 20 e 30 metri quadri, stampi il cognome, lo stipendio e la categoria.

```
int p, u;  /* indice di piano nell'edificio e di ufficio nel piano */
```

Esercizio 2 (2 punti). Trasformare il frammento di codice incluso nella parte sinistra della tabella in un equivalente frammento, da scrivere nella colonna destra, che contenga istruzioni while invece che for. Specificare, infine, qual è il valore stampato quando il codice viene eseguito, giustificando adeguatamente la risposta.

…
int i, j, s;
s = 1;
for (i=3; i>=0; i--)
  for (j=3; j>=0; j--)
  if ((i+j) % 2 != 0)
  s = s * 2;
printf("%d", s);
…	

Esercizio 3 (6 punti)

Si consideri il seguente programma in linguaggio C:

```
#define N -17

int main() {
   int a = N/10;
   float b = N/10.0;
   double c = N/10.0;
   double d = b;
}
```

Sapendo che i tipi delle variabili utilizzate sono codificati in binario in questo modo:

- `int`: complemento a due a 32 bit
- `float`: virgola mobile a precisione singola secondo lo standard IEEE 754-1985 (1 bit per il segno, 23 bit per la mantissa, 8 per l'esponente (K = 127))
- `double`: virgola mobile a precisione doppia secondo lo standard IEEE 754-1985 (1 bit per il segno, 52 bit per la mantissa, 11 per l'esponente (K = 1023))

Si risponda alle seguenti domande:

1. Qual è il valore in decimale e in binario della variabile **a** alla fine dell'esecuzione dell'ultima istruzione? Giustificare la risposta mostrando i calcoli effettuati.
2. Qual è il valore in decimale e in binario della variabile **b** alla fine dell'esecuzione dell'ultima istruzione? Giustificare la risposta mostrando i calcoli effettuati.
3. Considerando il risultato ottenuto nel punto 2, alla fine dell'esecuzione dell'ultima istruzione le variabili **c** e **d** contengono lo stesso valore binario? La risposta cambierebbe se la costante **N** fosse definita pari a 10 anziché a -17? Non è necessario ricavare il valore in binario di **c** e **d**, ma la risposta deve essere adeguatamente motivata.

Esercizio 1 (10 punti)

Una fattoria gestisce una stalla con 10 mucche, ciascuna identificata da un codice alfanumerico di 8 caratteri.

Il gestore della fattoria vuole realizzare un programma in C che consenta la contabilizzazione del latte prodotto in un giorno dalle 10 mucche.

In questo contesto, si risponda ai seguenti punti:

1. Si definisca in C il tipo **ProduzioneGiornaliera** che rappresenti la quantità di latte prodotta da ognuna delle 10 mucche della stalla. La produzione giornaliera di ogni mucca deve essere associata al suo codice identificativo. Si definiscano quindi ulteriori tipi se necessario. La quantità di latte prodotta da ciascuna mucca è espressa in litri.

2. Si definisca la variabile **produzione**, utilizzando in modo appropriato il tipo **ProduzioneGiornaliera** definito al punto 1, per la memorizzazione della produzione giornaliera delle 10 mucche della stalla. Si supponga inoltre che la variabile **produzione** sia stata opportunamente inizializzata con la produzione giornaliera delle 10 mucche della stalla.
   Si scriva un frammento di programma C che stampi a video il codice identificativo e la quantità di latte prodotta da ciascuna mucca della stalla. Si stampi inoltre la quantità totale di latte prodotto nella stalla.

Esercizio 1 (7 punti)

Siano date le seguenti definizioni di strutture dati:

```
#define MAXDB 80

typedef struct
{
  int giorno;
  int mese;
  int anno;
} data;

typedef char stringa[20];

typedef struct
{
  data      data_di_nascita;
  int       giorno_dell_anno;
  int       codice_crimine;
  stringa   stringa_crimine;
} arresto;

arresto database[MAXDB];
int arresti_per_giorno[365];
```

ove `database` contiene gli arresti effettuati in un determinato anno mentre `arresti_per_giorno` è una variabile array ausiliaria (i cui elementi sono inizializzati tutti a `0`) utilizzabile per i calcoli intermedi.

Scrivere una porzione di codice C che ricavi e stampi a video il massimo numero di arresti giornalieri dei nati nel 1979 con `codice_crimine` pari a `555` utilizzando i dati contenuti in `database`. Si definiscano e inizializzino le eventuali altre variabili temporanee necessarie per tale calcolo. Si assuma che il database **non sia ordinato**.