CORSO DI LAUREA IN INGEGNERIA AEROSPAZIALE

Prof. G. Agosta
Prof. Gerardo Pelosi

Informatica – a.a.2022/2023 – 3zo Appello – 30 Giugno 2023

Cognome _____ **Matricola o Cod. Persona** _____

Nome_____ **Firma** _____

Istruzioni

Non separate questi fogli. Scrivete la soluzione **solo sui fogli distribuiti**, utilizzando il retro delle pagine se necessario. **Cancellate le parti di brutta** (o ripudiate) con un tratto di **penna**.

Ogni parte non cancellata a penna sarà considerata parte integrante della soluzione.

È possibile scrivere a matita e non occorre ricalcare al momento della consegna.

Qualsiasi **tentativo** di comunicare con altri studenti comporta **l'espulsione** dall'aula.

Non è possibile lasciare l'aula conservando il tema della prova in corso.

È possibile ritirarsi senza penalità.

Tempo a disposizione: **2h:30**

Valore indicativo degli esercizi, voti parziali e voto finale:

Esercizio 1	(4 punti)	_____
Esercizio 2	(6 punti)	_____
Esercizio 3	(10 punti)	_____
Esercizio 4	(4 punti)	_____
Esercizio 5	(8 punti)	_____
Totale	(32 punti)	_____

Voto finale

Esercizio 1 [4 punti]

Il seguente programma acquisisce come *input* tre numeri interi nell'insieme `{0, 1}` dal terminale e riporta come *output* a video `1` se due e solo due di essi corrispondono al valore di verità "vero", altrimenti `0`.
Nel caso in cui uno o più degli *input* siano scorretti, riporta invece `-1`.
Nel programma sono presenti alcuni errori di sintassi e logici che ne pregiudicano il corretto funzionamento. Correggere tali errori.

Esempio:

```
Input: 0 0 0
Ouput: 0

Input:  0 1 1
Output: 1

Input:   1 0 2
Output: -1
```

```c
#include <stdio.h>

int main() {
    int a; b, c;
    fscanf(stderr, "%d%d%d", &a, &d, &c);
    if (a != 1 && a != 0 || b != 1 && b != 0 || c != 1 && c != 0)
        fprintf(stdin, "-1\n");
    else if ((!a && b && c) || (a && !b && c) || (a && b && c))
        fprintf(stdout, "1\n");
    else
        fprintf(stdin, "0\n")

    return 0;
} // end main
```

Soluzione:

```c
#include <stdio.h>

int main() {
    int a, b, c;
    fscanf(stdin, "%d%d%d", &a, &b, &c);
    if ((a != 1 && a != 0 )|| (b != 1 && b != 0) || (c != 1 && c != 0))
        fprintf(stdout, "-1\n");
    else if ( (!a && b && c) || (a && !b && c) || (a && b && !c) )
        fprintf(stdout, "1\n");
    else
        fprintf(stdout, "0\n");

    return 0;
} // end main
```

Se si continua sul retro di qualche foglio, indicare quale

Esercizio 2 [6 punti]

Scrivere un programma C che acquisisca da terminale una stringa i cui caratteri sono da interpretare come cifre di un numero intero espresso in base **dodici** (`0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 'a' = 10`$_{dec}$, `'b' = 11`$_{dec}$).
La lunghezza della stringa può variare da 1 a 8 caratteri alfanumerici.
Il programma deve calcolare il valore decimale del numero in base 12 memorizzato come stringa e riportare a video sia il numero minimo di bit necessari per rappresentarlo in binario naturale sia, sulla riga successiva, il valore del numero in formato esadecimale.
Nel caso la stringa non rispetti la specifica, il programma riporterà a video il numero `-1`.
In entrambi i casi, il programma restituisce il valore `0`.

Esempi:

Input: 4	*Input:* a3b	*Input:* c
Output: 3	*Output:* 12	*Output:* -1
4	5cf	

Soluzione:

```c
#include <stdio.h>
#include <string.h>

int daBase12aDecimale(char c) {
    if (c == 'a' || c == 'A') return 10;
    if (c == 'b' || c == 'B') return 11;
    if (c < '0'  || c > '9' ) return -1;
    return (int) c - (int) '0';
} // end daBase12aDecimale

int numBit(int v) {
    int cont = 0;
    do {
        v = v / 2;
        cont += 1;
    } while (v != 0);
    return cont;
} // end numBit

int main() {
    char s[9]; int v, n = 0;
    scanf("%s", s);
    if (strlen(s) < 1 || strlen(s) > 8) {
        printf("-1\n");
        return 0;
    }
    for (int i = 0; s[i] != '\0'; i++) {
        v = daBase12aDecimale(s[i]);
        if (v < 0) { printf("-1\n");
            return 0;
        }
        n = n*12 + v;
    }
    printf("%d\n%x\n", numBit(n), n);
    return 0;
} // end main
```

Se si continua sul retro di qualche foglio, indicare quale

Esercizio 3 (10 punti)

Scrivere un programma C che legga da terminale una stringa di lunghezza arbitraria, considerando soltanto i caratteri alfabetici (cioè quelli che rappresentano lettere maiuscole o minuscole, escluse quelle dotate di accenti o altri segni diacritici), e stampi a terminale:

1) sulla prima riga, la **mediana** del numero di occorrenze dei caratteri alfabetici nella stringa letta (considerando identici i caratteri maiuscoli e minuscoli), o `0` in caso di nessun carattere valido letto;
   La mediana in una sequenza numerica è definita come il "valore centrale" della sequenza ordinata in ordine ascendente. Se la sequenza è di lunghezza pari, la mediana è calcolata come la media aritmetica dei due valori centrali.
   Esempio: `[1, 4, 5, 7]`, mediana = `(4+5)/2= 4.5`
   Esempio: `[6, 11, 16]`, mediana = `11`
   Attenzione: per ogni esecuzione del programma la mediana deve essere calcolata considerando le lettere inserite (quindi quelle con num. di occorrenze >= 1) e non necessariamente tutto l'alfabeto.
2) sulle righe successive, i caratteri (minuscoli, uno per riga, e in ordine alfabetico) il cui numero di occorrenze è diverso da zero, seguiti dal carattere `':'` e dal proprio numero di occorrenze;
3) sull'ultima riga, la sequenza di caratteri (minuscoli, in ordine alfabetico, e senza spaziature) il cui numero di occorrenze è maggiore della mediana.

Soluzione:

```c
#include <stdio.h>
#include <ctype.h>

#define N 26

float mediana(int a[N]);

int main() {
    int occorrenze[N] = {0};
    char c;
    int i;

    do {
        c = (char) getchar();
        if (c >= 'A' && c <= 'Z')
            c = (char)((int)c-(int)'A'+(int)'a');
        if (c >= 'a' && c <= 'z')
            occorrenze[(int)c-(int)'a'] += 1;
    } while (!isspace(c) && c != '\0' && c != EOF);

    float m = mediana(occorrenze);
    printf("%f\n", m);

    for (i = 0; i < N; i++) {
        if (occorrenze[i] != 0)
            printf("%c:%d\n", (char)((int)'a'+i), occorrenze[i]);
    } // end for

    for (i = 0; i < N; i++) {
        if (occorrenze[i] > m)
            printf("%c", (char)((int)'a'+i) );
    }  // end for
    printf("\n");

    return 0;
} // end main
```

```
float mediana(int a[N]) {
    int array[N], cont = 0;
    int i, j;

    for (i = 0; i < N; i++) {
        if (a[i] == 0) cont++;
        array[i] = a[i];
    }

    if (cont == N) {
        return 0;
    }

    for (i = 0; i < N; i++) {
        for (j = 0; j < N; j++) {
            if (array[i]<array[j]) {
                int t = array[i];
                array[i] = array[j];
                array[j] = t;
            }
        } // end for
    } // end for

    if ((N-cont) % 2 == 0)
        return ( (float) array[cont+(N-cont)/2] +
                 (float) array[cont+(N-cont)/2-1] ) / 2;
    return (float) array[cont+(N-cont)/2];
} // end mediana
```

Se si continua sul retro di qualche foglio, indicare quale

Esercizio 4 (4 punti)

Si progetti una funzione

`...aggiornaMatrice(...)`

che riceve in ingresso una matrice `NxN` contenente valori interi nell'insieme `{0, 1}`, e che modifica la matrice in modo tale che ogni cella della matrice contenga, dopo l'esecuzione della funzione, `1` se il conteggio dei valori non nulli nelle celle adiacenti (inclusa la cella corrente) è maggiore o uguale al conteggio dei valori nulli, `0` altrimenti.

Esempio con `N == 4`

Input:

	[0]	[1]	[2]	[3]
[0]	1	0	0	1
[1]	0	1	0	1
[2]	1	0	1	0
[3]	1	1	0	1

Output:

	[0]	[1]	[2]	[3]
[0]	1	0	1	1
[1]	1	0	0	1
[2]	1	1	1	1
[3]	1	1	1	1

Soluzione:

```
void copiaMatrice(int r[N][N], int m[N][N]) {
    int i, j;
    for (i = 0; i < N; i++) {
        for (j = 0; j < N; j++) {
            r[i][j] = m[i][j];
        } // end for
    } // end for
} // end copiaMatrice

void aggiornaMatrice(int m[N][N]) {
    int r[N][N];
    int i, j, k, h;
    copiaMatrice(r, m);
    for (i = 0; i < N; i++) {
        for (j = 0; j < N; j++) {
            int somma = 0, contaAdiacenti = 1;
            for (k = -1; k <= 1; k++) {
                for (h = -1; h <= 1; h++) {
                    if (i + k >= 0 && i + k < N && j + h >= 0 && j + h < N) {
                        somma += m[i + k][j + h];
                        contaAdiacenti += 1;
                    }
                } // end for
            } // end for
            if (somma >= contaAdiacenti / 2)
                r[i][j] = 1;
            else
                r[i][j] = 0;
        } // end for
    } // end for
    copiaMatrice(m, r);
} // end aggiornaMatrice
```

Esercizio 5 (8 punti)

Si completi il programma seguente esibendo il codice dei sottoprogrammi

`...leggi_aerei(...);` e `...capacita_max(...);`

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

struct aereo {
    char modello[128];
    int velocita;
    int passeggeri;
};

typedef struct aereo aereo_t;

aereo_t *leggi_aerei(char *nome_file, int *numero_aerei);
void capacita_max(aereo_t *aerei, int numero_aerei);

int main() {
    int numero_aerei = 0;
    char file[200];

    scanf("%s", file);
    aereo_t *aerei = leggi_aerei(file, &numero_aerei);
    printf("%d\n", numero_aerei);

    if (!aerei) {
        return 0;
    }

    capacita_max(aerei, numero_aerei);
    free(aerei);

    return 0;
} // end main
```

Il programma legge da un *file* che contiene nomi, velocità e numero di passeggeri di vari modelli di aerei di linea e stampa a terminale:

(a) il conteggio di quanti aerei sono descritti nel *file*;

(b) i modelli di aereo con la massima capacità effettiva (ovvero la capacità di trasportare il massimo numero di passeggeri per unità di tempo).

Il *file* letto ha il seguente formato (il numero di righe del *file* è arbitrario):

```
Modello     Velocità    Passeggeri
Concorde    2179        100
Tu-144      2300        110
Boeing-747  1061        660
Boeing-707   987        194
Il-62        850        186
A380        1049        868
…
```

La prima riga contiene l'intestazione e deve essere scartata.
Ogni riga successiva alla prima contiene tre campi: modello, velocità e passeggeri.
I modelli sono stringhe prive di spazi composte da caratteri alfanumerici, con al più la presenza del carattere trattino (−). Le velocità e i passeggeri sono numeri interi.

Se si continua sul retro di qualche foglio, indicare quale

Soluzione:

```c
int conta_aerei(FILE *file){
    int i = 0;
    char a[128], b[128], c[128];
    int n, m;

    if (fscanf(file, "%s%s%s", a, b, c) != 3) {
        return 0;
    }

    while(fscanf(file, "%s%d%d", a, &n, &m) == 3) {
        i++;
    }

    rewind(file);
    fscanf(file, "%s%s%s", a, b, c);

    return i;
} // end conta_aerei

aereo_t* leggi_aerei(char *nome_file, int *numero_aerei) {
    int i;
    aereo_t *res = NULL;
    FILE *fin = fopen(nome_file, "r");

    if (!fin) {
        return NULL;
    }

    *numero_aerei = conta_aerei(fin);
    res = (aereo_t*) malloc(sizeof(aereo_t) * (*numero_aerei));

    if (!res) {
        fclose(fin);
        return NULL;
    }

    for(i = 0; i < *numero_aerei; i++) {
        if (fscanf(fin, "%s%d%d", res[i].modello, &res[i].velocita, &res[i].passeggeri)
!= 3) {
            return res;
        }
    } // end for

    fclose(fin);
    return res;
} // end leggi_aerei
```

```c
void capacita_max(aereo_t* aerei, int numero_aerei) {
    int max = 0;
    int i;

    for (i = 0; i < numero_aerei; i++) {
        if (max < aerei[i].velocita * aerei[i].passeggeri) {
            max = aerei[i].velocita * aerei[i].passeggeri;
        }
    } // end for

    for (i = 0; i < numero_aerei; i++) {
        if (max == aerei[i].velocita * aerei[i].passeggeri) {
            printf("%s\n", aerei[i].modello);
        }
    } // end for
} // end capacita_max
```

Se si continua sul retro di qualche foglio, indicare quale