



Politecnico di Milano

Dipartimento di Elettronica, Informazione e Bioingegneria

Informatica A - Prof. A. Fuggetta - a.a 2023/2024 - 22 gennaio 2024

Cognome: _____ Matricola: _____
Nome: _____ Firma: _____

Istruzioni

- Non separate questi fogli. Scrivete la soluzione **solo sui fogli distribuiti**, utilizzando il retro delle pagine in caso di necessità. **Cancellate le parti di brutta** con un tratto di **penna**.
- Ogni parte non cancellata a penna sarà considerata parte integrante della soluzione.
- **NON è possibile scrivere a matita.**
- Scrivere nome e cognome su tutti i fogli. Non verranno corretti compiti non firmati o con nome illeggibile
- È **vietato** utilizzare **calcolatrici, telefoni o pc**. Chi tenti di farlo vedrà **annullata** la sua prova.
- **Non è ammessa la consultazione di libri e appunti.**
- Qualsiasi **tentativo** di comunicare con altri studenti comporta **l'espulsione** dall'aula.
- È possibile ritirarsi senza penalità.
- Non è possibile lasciare l'aula conservando il tema della prova in corso.
- Tempo a disposizione: **2h**

Valore indicativo degli esercizi, voti parziali e voto finale:

Esercizio 1	5 punti	_____
Esercizio 2	6 punti	_____
Esercizio 3	7 punti	_____
Esercizio 4	14 punti	_____
Totale(32)		_____

Esercizio 1 - Algebra di Boole, Aritmetica Binaria, Codifica delle Informazioni (5 punti)

- (a) Si costruisca la tabella di verità della seguente espressione booleana in tre variabili, badando alla precedenza tra gli operatori logici. Eventualmente si aggiungano parentesi.
Scrivere l'espressione semplificata. Non si accetteranno soluzioni senza il procedimento. (2 punti)

$$(A \text{ AND NOT } B) \text{ OR } (\text{NOT } A \text{ AND } B \text{ AND } C) \text{ OR } A \text{ AND NOT } C$$

$$A\bar{B} + \bar{A}B C + A\bar{C} \rightarrow \text{es'is simpl.}$$

A	B	C	0
0	0	0	0
0	1	0	0
1	0	1	1
1	1	0	0

A	B	C	0
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	0

A	C	0
0	0	0
0	1	0
1	0	1
1	1	0

- (b) Si stabilisca il minimo numero di bit sufficiente a rappresentare in complemento a due i numeri $A = -47_{10}$ e $B = 82_{10}$ li si converta, se ne calcolino la somma $(A+B)$ e la differenza $(A-B)$ in complemento a due e si indichi se si genera riporto sulla colonna dei bit più significativi e se si verifica overflow. Non si accetteranno soluzioni senza il procedimento. (2 punti)

$$A = -47_{10} \rightarrow 47 = 00101111_2 \rightarrow \text{invertire: bit a pos} + 1$$

↓

$$11010000 + 1 \rightarrow 11010001 = -47$$

$$82 = \begin{matrix} 64 & 32 & 16 & 8 & 4 & 2 & 1 \\ 0 & 1 & 0 & 1 & 0 & 0 & 10 \end{matrix}$$

$$A+B = \begin{array}{r} 11010001 \\ + 01010010 \\ \hline \end{array}$$

$$\textcircled{1} 00100011$$

Riporto

$$A+(-B) \rightarrow 10101101 + 1$$

$$A-B = \begin{array}{r} 11010001 \\ + 10101110 \\ \hline \end{array}$$

$$\begin{array}{r} 10111111 \\ \hline \end{array} \textcircled{1}$$

- (c) Si converta il numero 12,3 in virgola fissa e in virgola mobile con codifica IEEE 754 con precisione singola. Non si accetteranno soluzioni senza il procedimento. (1 punti)

$12,3 \rightarrow \boxed{S = 0}$
 $12 = 1100$; $0,3 \rightarrow$

0.3	0	$\rightarrow 01001101 \dots$
0.6	1	
0.2	0	
0.4	0	
0.8	1	
0.6	1	
0.2	0	
0.4	0	

$1100 \ 01001101 \dots$
 $1,10001001101 \dots$
 $\boxed{e = 3 + 127 = 130}$

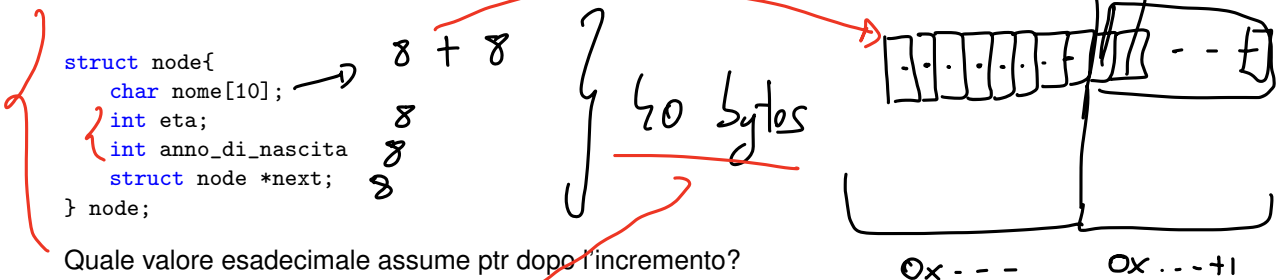
$0 \mid 100000010 \mid 10001001101 \dots$
 $\quad \quad \quad 8 \quad \quad \quad 23 \text{ bit}$

Esercizio 2 - Teoria (6 punti)

Segnare con una crocetta le risposte che si ritengono corrette. Per ogni domanda, possono essere presenti da 1 a 4 soluzioni corrette.

ogni parola è di 8 byte

- (a) In un computer con processore a 64bit, qual è il risultato dell'operazione 'ptr++' se 'ptr' è un puntatore relativo ad una lista i cui nodi sono composti come descritti qui sotto e contiene l'indirizzo esadecimale di memoria '0x1000'?



Quale valore esadecimale assume ptr dopo l'incremento?

- ☐ 0xFD8
☐ 0x1000
☐ 0x102C
☒ 0x1028

- (b) Qual è il valore della variabile 'value' alla fine dell'esecuzione della seguente funzione?

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

```
void elaboratoComplesso(int *result) {
    *result = 3;

```

```

    for (int k = 1; k <= 8; k++) {
        if (k % 2 == 0) {
            if (k % 4 == 0) {
                *result += k * 3;
            } else {
                *result -= k / 2;
            }
        } else {
            if (k % 3 == 0) {
                *result *= k;
            } else {
                *result += k;
            }
        }
    }
}
    
```

```

int main() {
    int value = 7;
    elaboratoComplesso(&value);

    printf("Il valore finale della variabile : %d\n", value);

    return 0;
}
    
```

- ☒ 54
☐ 42

result	k
3	X
4	1
3	2
9	3
21	4
26	5
23	6
30	7
54	8

- ☐ 30
- ☐ 15
- ☐ 23

(c) Qual è il valore decimale del numero 10100110 ad 8 bit complemento a due?

- ☐ 129
- ☐ 166
- ☒ -90
- ☐ 90

$$\begin{array}{r}
 \downarrow \\
 10100110 - \\
 \hline
 10100101 \\
 \begin{array}{ccccccc}
 64 & 32 & 16 & 8 & 4 & 2 & 1
 \end{array} \\
 01011010 \quad \text{invertito} \\
 \begin{array}{ccccccc}
 \nearrow & \uparrow & \uparrow & \uparrow & & &
 \end{array} \\
 64 + 16 + 8 + 2 = 90
 \end{array}$$

(d) Quali delle seguenti affermazioni sull'utilizzo delle liste in programmazione sono corrette?

- ☒ Le liste sono strutture dati dinamiche, il cui numero di elementi può variare durante l'esecuzione del programma, offrendo flessibilità nell'aggiunta o rimozione di elementi.
- ☐ Le liste sono strutture dati che permettono l'accesso casuale ai loro elementi perché sono salvate in RAM, garantendo una gestione efficiente della memoria.
- ☐ Le liste sono strutture dati statiche, il cui numero di elementi deve essere noto a priori, rendendo l'allocazione di memoria più efficiente.
- ☒ Le liste sono strutture dati che permettono l'inserimento e la rimozione di elementi in posizioni arbitrarie, facilitando la gestione dinamica della struttura dati.

(e) Quali sono le differenze principali tra le variabili globali e le variabili locali in linguaggio C?

- ☐ Le variabili globali sono dichiarate all'interno di una funzione, mentre le variabili locali sono dichiarate al di fuori di qualsiasi funzione.
- ☐ Le variabili globali possono essere utilizzate solo in un file sorgente specifico, mentre le variabili locali possono essere utilizzate in qualsiasi file sorgente del programma.
- ☐ Le variabili globali sono accessibili solo all'interno della funzione in cui sono dichiarate, mentre le variabili locali possono essere utilizzate da qualsiasi parte del programma.
- ☒ Le variabili globali mantengono il loro valore anche al di fuori della funzione in cui sono dichiarate, mentre le variabili locali perdono il loro valore una volta che la funzione è terminata.

(f) Dato il seguente codice c:

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

typedef struct node {
    int data;
    struct node *next;
} node;

typedef node *ptrNode;

ptrNode insert(ptrNode head, int data) {
    ptrNode new_node = NULL; malloc
    new_node->data = data;
    new_node->next = head;
    return new_node;
}

void print_list(ptrNode head) {
    ptrNode current = head;
    while (current != NULL) {
        printf("%d ", current->data);
        current = current->next;
    }
    printf("\n");
}

int main() {
    ptrNode lista = NULL;
```

```

insert(lista, 5);
insert(lista, 10);
insert(lista, 15);

print_list(lista);

return 0;
}

```

marco
: | salvataggio
della
head nuova

Cerchia/sottolinea o indica con una freccia l'errore

Inserisci qui sotto le eventuali righe mancanti per correggere il codice:

```

ptrNode insert (ptrNode head, int data) {
    ptrNode new Node = (ptrNode) malloc (sizeof(Node));
    if (newNode == NULL) {
        exit(1);
    }
    new Node -> data = data;
    new Node -> next = head;
    return new Node;
}

int main () {
    ptrNode lista = NULL;
    lista = insert(lista, 5);
    ...
    print_list (lista);
    return 0;
}

```

(g) **Matlab (7 punti)**

Scrivere il codice Matlab che restituisca i valori richiesti. Attenersi al numero massimo di righe di codice indicato.

1. Creare la matrice A di dimensione quadrata casuale tra 3 e 7 contenete solo valori pari 10(1riga)(1 punto)

$$A = \text{ones}(\text{randi}([3,7]) * 2 * \text{randi}([1,5]))$$

2. Inserire nelle righe pari valori casuali tra -10 e 10 (ogni elemento delle righe deve avere un valore diverso) (1 riga)(1 punto)

$$A(2:2:\text{end}, :) = \text{randi}([-10, 10], \text{sum}(\text{mod}(\underbrace{1:\text{size}(A,1)}_{\text{\# di righe}}, 2) == 0, \underbrace{\text{size}(A,2)}_{\text{indica righe pari}}), \text{size}(A,2));$$

costo la righe pari

3. Eliminare le righe dispari (1 riga)(1 punto)

$$A(1:2:\text{end}, :) = [];$$

4. Aggiungere una riga ed una colonna contenente la media delle colonne e la media delle righe (1 riga)(1 punto)

$$A = [A; \text{mean}(A, 1)]; A = [A, \text{mean}(A, 2)]$$

5. Calcolare la somma totale delle sole righe dispari (1 riga - 1 punti)

$$\text{sum_righe_dispari} = \text{sum}(A(1:2:\text{end}, :), 'all');$$

6. Scrivere una funzione che presa in ingresso la matrice ed un vettore, calcoli e ritornifin quanti sono gli elementi dell'array contenuti nella matrice. Si dia per scontato che tutti gli elementi della matrice e dell'array sono presenti una sola volta (non ci sono doppi) (4 righe)(2 punti).

$$\text{function count} = \text{count_elements_in_matrix}(M, v)$$

$$\begin{aligned} &\text{count} = 0; \\ &\text{for } i = 1:\text{length}(v) \\ &\quad \text{if } \text{any}(M(:, i) == v(i)) \rightarrow \\ &\quad \quad \text{count} = \text{count} + 1; \\ &\text{end} \end{aligned}$$

(h) Programmazione C Liste (14 punti)

Si presuma di dovere implementare un applicativo per la gestione del menu di un ristorante. Il menu è da considerarsi come la lista dei piatti che lo chef è in grado di offrire ai suoi clienti. Ogni piatto è caratterizzato da un nome, da una descrizione, e da un costo. Inoltre, un piatto può essere vegetariano oppure no.

Svolgere l'esercizio attenendosi a quanto richiesto. **NON E' RICHiesto SCRIVERE IL MAIN.**

1. Si definiscano le strutture dati necessarie allo sviluppo di questo programma. (1 punto)
2. Si implementi una funzione che crea e restituisce una lista contenente solo i piatti del menu che sono vegetariani. (4 punti)
3. Si implementi una funzione RICORSIVA che riceve una lista dei piatti scelti dai clienti e calcola il costo totale della consumazione. (3 punti)
4. Scrivere una funzione RICORSIVA che, presa in ingresso la lista dei piatti e i dati relativi ad un nuovo piatto, aggiunga il nuovo nodo alla lista in ordine crescente di prezzo. (3 punti)
5. Scrivere una funzione che, presa in ingresso la lista ed il nome di un piatto, elimini dalla lista il piatto che corrisponde al nome inserito. La funzione restituisce un valore per indicare se il piatto è stato trovato/eliminato, oppure se il piatto non è stato trovato. (3 punti)

~~x. (SOLO PER CHI NON HA SUPERATO IL LABORATORIO)~~

~~Scrivere una funzione presa in ingresso un prezzo, crei una nuova lista contenente solamente i piatti che costano più del prezzo inserito.~~

```

① type def struct Piatto {
    char nome[50];
    char descrizione[200];
    float costo;
    int vegetariano;
} Piatto;

```

```

typedef struct Node {
    Piatto piatto;
    struct Node * next;
} Node;

```

```

② Node * filtraVegetarian (Node * head) {
    Node * newHead = NULL;
    Node * temp = head;
    Node * prev = NULL;

```

```
while (temp != NULL) {
```

```
if (temp->path Vegetarian == 1) {
```

NOME e COGNOME: _____

```
Node * new_node = (Node*) malloc (sizeof(Node));
```

```
new_node->path = temp->path;
```

prev = [P2]

```
new_node->next = NULL;
```

```
if (prev == NULL) {
```

```
new_head = new_node;
```

```
} else {
```

```
prev->next = new_node;
```

```
}
```

```
prev = new_node;
```

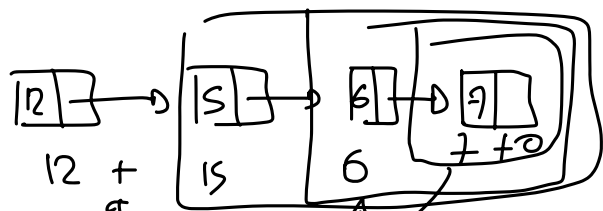
```
}
```

```
temp = temp->next;
```

if (new_head == NULL) →

```
}
```

```
return new_head;
```



③

```
float calcTot (Node* head) {
```

```
if (head == NULL) {
```

```
return 0;
```

```
} else {
```

```
return head->path + calcTot (head->next);
```

```
}
```

```

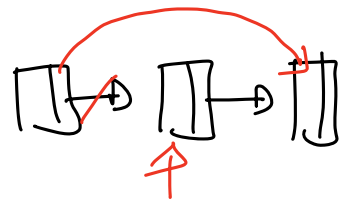
④ Node * aggPathsOrdine (Node * head, Paths new_paths) {
    if (head == NULL || head->paths.cost > new_paths.cost) {
        Node * new_node = (Node *) malloc (sizeof (Node));
        new_node->paths = new_paths;
        new_node->next = head;
        return new_node;
    }
    head->next = aggPathsOrdine (head->next, new_paths);
    return head;
}

```

```

⑤ int eliminatePaths (Node * head, const char * name) {
    Node temp = Node head;
    Node prev = NULL;
    while (temp != NULL) {
        if (strcmp(temp->paths.name, name) == 0) {
            if (prev == NULL) {
                head = temp->next;
            } else {
                prev->next = temp->next;
            }
            free(temp);
        }
        prev = temp;
        temp = temp->next;
    }
}

```



return 1;

}
return 0;

}