Cognome ..... Nome ..... Matricola .....

(Se si sta sostenendo un' integrazione, indicare da quanti crediti: .....)

Programmazione - Appello del 20/7/2022

Istruzioni

- Prima di iniziare, scrivete nome, cognome e matricola su *ogni foglio*. Se state sostendo un'integrazione, indicatelo sopra.
- Dall'inizio della prova avete a disposizione 1h:30 (per quelli di voi che hanno diritto al 30% di tempo aggiuntivo 1:57h, 70 minuti per chi fa l'integrazione da 6-7 crediti) per consegnare il compito alla cattedra. Consegne in ritardo non verranno accettate. Se state facendo un'integrazione da x crediti, dovete completare solamente gli esercizi con indicato (integrazione $\geq$ x crediti).
- Potete usare solamente penne, matita e gomma. Nient'altro può essere presente sul banco. Non potete usare fogli aggiuntivi, nemmeno per la brutta copia. Appoggiate zaini e giacche a lato della stanza. Scrivete in modo leggibile, parti non comprensibili non verranno corrette.
- Vi chiedo di non parlare e non alzare lo sguardo dal vostro foglio. Una volta finito il compito non utilizzate il cellulare. Visto che siete seduti vicini gli uni agli altri, sarò intransigente, segnandomi la minima trasgressione e riservandomi successivamente come agire.

Cognome ..... Nome ..... Matricola .....

1. L'elemento A[0][0] dell'array sottostante è allocato nella cella di memoria di indirizzo 100. Calcolare la cella di memoria dove viene immagazzinato il valore A[5][4]. Potete assumere che ogni elemento dell'array occupi una singola cella di memoria. Motivare la risposta. **2 punti**

```
1     int A[6][8];
```

Risposta:

2. Osservare il seguente codice e specificare se compila e, nel caso, cosa stampi. Motivare la risposta (risposte corrette senza motivazione non verranno valutate). **5 punti**

```
1  #include <stdio.h>
2
3  int x = 10;
4
5  int fun(int y) {
6      static int x = 1;
7      x += y;
8      return x;
9  }
10
11 int main(void) {
12
13     int y = fun(2);
14     printf("%d\n", x/y*3);
15     {
16         int x = 3;
17         int y = fun(x);
18         printf("%d\n", fun(y)+x);
19     }
20     printf("%d\n", fun(y)+x);
21 }
```

Risposta:

Cognome ..... Nome ..... Matricola .....

3. Implementare una funzione ricorsiva che, dati due array A e B, restituisce:
   - 1 se tutti i valori di A sono presenti in B
     - ciascuno con lo stesso segno o con segno opposto
     - nello stesso ordine;
   - 0 altrimenti.

   Negli esempi seguenti i numeri sottolineati sono gli elementi di B che fanno si che la funzione restituisca 1:

   1. la funzione restituisce 1 per A={1,-3,2} e B={1,9,8,3,8,6,2,7,9} // 1 3 2 sono presenti in B nello stesso ordine (anche se 3 ha segno opposto);
   2. la funzione restituisce 0 per A={1,3,2} e B={1,9,8,4,8,6,2,7,9} // non c'è l'elemento 3 in B;
   3. la funzione restituisce 0 per A={1,3,2} e B={1,9,8,4,8,6,2,7,3} // il 2 ed il 3 in B non sono nello stesso ordine nel quale appaiono in A;
   4. la funzione restituisce 1 per A={1,3,2} e B={1,9,8,4,8,6,2,7,3,5,2} // da B si può estrarre la tripla 1,3,2 che corrisponde ai valori di A.

   N.B. Soluzioni iterative che richiamano la funzione stessa al termine senza calcolare niente ricorsivamente non saranno accettate. **8 punti** (integrazione $\geq$ 6 crediti)

Risposta:

Cognome ..... Nome ..... Matricola .....

4. Tenendo conto delle dichiarazioni e del frammento di codice riportato di seguito, scrivere la funzione *clone_list* che, ricevuta una lista collegata con puntatori, dovrà crearne una copia / clone. Vi è richiesto di scrivere soltanto la funzione. **6 punti** (integrazione ≥ 3 crediti)

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

4  struct nodo {
5      float value;
6      struct nodo *nextPtr;
7  };

9  typedef struct nodo Lista;

11 void clone_list(Lista *srcPtr, Lista **destPtr);
```

Risposta:

Scrivere adesso la porzione di codice necessaria nella funzione chiamante (ad es. nel *main*) per poter effettuare la copia della lista. **1 punto** (integrazione ≥ 3 crediti)

Risposta:

5. Tenendo conto delle dichiarazioni e del frammento di codice riportato di seguito, scrivere la funzione *search* che, dato un albero binario di ricerca, dovrà ricercare al suo interno un particolare valore *val*. La funzione restituirà 1 se l'elemento è presente. Vi è richiesto di scrivere soltanto la funzione. **6 punti** (integrazione ≥ 3 crediti)

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

Cognome ..... Nome ..... Matricola .....

```
4  struct btree {
5      int valore;
6      struct btree *leftPtr;
7      struct btree *rightPtr;
8  };

10 typedef struct btree BTree;

12 int search(BTree *ptr, int val);
```

Risposta:

6. Dato un albero binario di ricerca (come quello dell'esercizio precedente), si riporti la definizione di visita simmetrica, anticipata e posticipata, aiutandosi con un esempio. **3 punti** (integrazione $\geq$ 3 crediti)

Risposta: