

Prova scritta di Programmazione II del 13/07/2021

NOME:

COGNOME:

MATRICOLA:

Domanda 3

Risposta non ancora data

Punteggio max.: 2,00

Indicare la o le risposte corrette

Scegli una o più alternative:

- ☐ a. I file **.h** si usano per offrire un'interfaccia del modulo
- ☐ b. I file **.h** vanno sempre inseriti in ordine alfabetico, avendo cura di inserire solo quelli necessari per il file, altrimenti viene generato un errore di tipo **ERROR_USELESS_HEADER_FILE**
- ☐ c. Dividere il codice in moduli rende più manutenibile il progetto
- ☐ d. Un file **nomefile.h** può essere inserito anche in diversi file **.cc** e non solo in **nomefile.cc**

NOME:

COGNOME:

MATRICOLA:

Domanda 4

Risposta non ancora data

Punteggio max.: 3,00

Dire cosa stampa il seguente codice

```
#include <iostream>
```

```
using namespace std;
```

```
int f1(int n) {  
    return n-1;  
}
```

```
int f2(int n) {  
    return n-2;  
}
```

```
int f(int n, int (*f1)(int)) {  
    if (n < 0)  
        return n;  
    if (n % 2 == 0)  
        return f(f1, f1);  
    else  
        return f(f1, f2);  
}
```

```
int main() {  
    cout << f(6,f2);  
}
```

Risposta:

NOME:

COGNOME:

MATRICOLA:

Domanda 5

Risposta non ancora data

Punteggio max.: 3,00

Dire cosa stampa il seguente programma

```
#include <iostream>
```

```
using namespace std;
```

```
int main() {  
    int a[5] = {0,3,4,5,6,5};  
    int *p = &a[2];  
    a[*p] = a[*{p-1}];  
    a[3] = *a;  
    p = p - 2;  
    *p = 18;  
    *(p+1) = *(p+3);  
    for (int i = 0; i < 6; i++) {  
        cout << a[i] << " ";  
    }  
}
```

Risposta:

Domanda 6

Risposta non ancora data

Punteggio max.: 3,00

Spiegare brevemente a cosa servono i breakpoint e i watch operator nel debugging.

Risposta:

NOME:

COGNOME:

MATRICOLA:

Domanda 7

Risposta non ancora data

Punteggio max.: 5,00

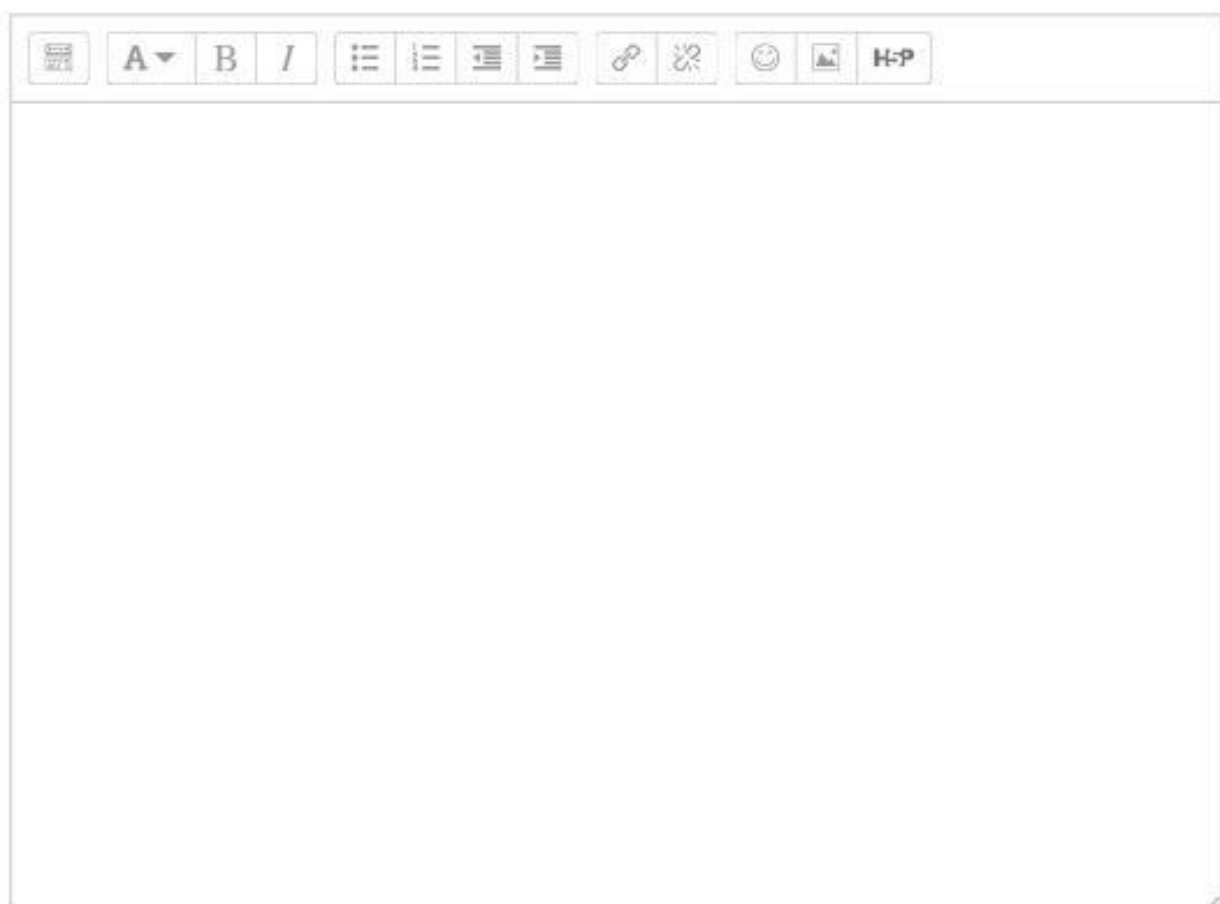
Date le seguenti dichiarazioni per un binary search tree con chiave rappresentata da una stringa, assumendo che i nodi siano ordinati in ordine alfanumerico:

```
struct bnode {  
    char* key;  
    bnode* left;  
    bnode* right;  
    bnode* parent;  
};  
typedef bnode* bst;
```

Scrivere il codice della primitiva per la ricerca di un nodo

```
bnode* bst_search(bst, char*);
```

in forma ricorsiva



NOME:

COGNOME:

MATRICOLA:

Domanda 8

Risposta non ancora data

Punteggio max.: 6,00

Date le seguenti dichiarazioni per un binary search tree con chiave intera e le relative primitive, assumendo che i nodi siano ordinati in *ordine decrescente* (dal più grande al più piccolo):

```
typedef int tipo_key;
```

```
struct bnode {
```

```
    tipo_key key;
```

```
    bnode* left;
```

```
    bnode* right;
```

```
    bnode* parent;
```

```
};
```


```
tipo_inf get_value(bnode*); //restituisce il valore del nodo in ingresso
```

```
bst get_left(bst); //restituisce il sottoalbero sinistro dell'albero in ingresso
```

```
bst get_right(bst); //restituisce il sottoalbero destro dell'albero in ingresso
```

```
bnode* bst_search(bst, tipo_key);
```


Scrivere la funzione void print_subtree(bst b, int x, int y) che dato un valore intero x presente nel BST b, stampi i valori nel sottoalbero radicato nel nodo che contiene x minori di y. La funzione *deve* sfruttare le caratteristiche del BST.





A ▼


B


I























NOME:

COGNOME:

MATRICOLA:

Domanda 9

Risposta non ancora data

Punteggio max.: 6,00

Data una lista *l* semplice di interi definita come sotto, scrivere una funzione *lista split(lista& l)* che restituisca una lista *l'* contenente gli elementi di *l* con valore pari mentre nella lista *l* rimangono tutti gli elementi con valore dispari. Ad esempio per la lista [1,3,2,5,6], *l'* sarà così rappresentato [2,6] e *l* diventerà [1,3,5]. La funzione deve far uso delle primitive specificate sotto, non deve creare o distruggere elementi ma effettuare operazioni sui puntatori.

```
struct elem
```

```
{
```

```
    int v;
```

```
    elem* pun ;
```

```
};
```

```
typedef elem* lista ;
```

```
tipo_inf head(lista);
```

```
lista tail(lista);
```

