1 Esercizio 3_1

Il file `esercizio3.cc` contiene la definizione di una struttura dati rappresentante il nodo di un albero binario, insieme all'implementazione della funzione `generaAlbero` che crea un albero binario (non ordinato) contenente numeri interi positivi.

Scrivere nel file `esercizio3.cc` le dichiarazioni e le definizioni delle seguenti tre funzioni:

- `creaAlberoDiRicercaBinario`: questa funzione prende come unico argomento la radice dell'albero creato dalla funzione `generaAlbero`. La funzione `creaAlberoDiRicercaBinario` deve creare un *albero di ricerca binario* contenente solo i numeri **dispari** contenuti nell'albero fornito come argomento e ritornare la radice dell'albero di ricerca binario appena creato. **Suggerimento**: utilizzare la struttura dati già definita nel file `esercizio3.cc` per costruire l'albero di ricerca binario;

- `stampaAlberoInOrdine`: questa funzione prende come unico argomento la radice dell'albero di ricerca binario appena creato. La funzione `stampaAlberoInOrdine` deve stampare a video (su un'unica riga) gli elementi dell'albero di ricerca binario in ordine **crescente**;

- `deallocaAlbero`: questa funzione prende come unico argomento la radice di un albero binario e ne dealloca **tutti** i nodi.

Questi sono due esempi di esecuzione:

```
computer > ./a.out
L'albero iniziale è: 4 5 5 6 6 4 5 1 3 0
L'albero di ricerca binario è: 1 3 4 5 5 5

computer > ./a.out
L'albero iniziale è: 6 2 2 8 5 3 2 9 4 6
L'albero di ricerca binario è: 3 5 6 9
```

Note:

- Scaricare il file `esercizio3.cc`, modificarlo per inserire la corretta definizione e implementazione delle funzioni `creaAlberoDiRicercaBinario`, `stampaAlberoInOrdine` e `deallocaAlbero` e infine caricare il file risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito;

- Notare che il file `esercizio3.cc` contiene già la definizione della funzione `main`, insieme a un'ulteriore funzione ausiliaria `stampaAlbero` che stampa graficamente a video un albero binario (nel caso in cui possa esservi utile visualizzare l'albero);

- E' consentito definire e implementare funzioni ausiliarie che possano aiutarvi nella soluzione del problema;
- All'interno di questo programma non è ammesso l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `iostream` e `cstdlib`.

2 Esercizio 3_2

Il file `esercizio3.cc` contiene la definizione di una struttura dati rappresentante il nodo di un albero binario, insieme all'implementazione della funzione `generaAlbero` che crea un albero binario (non ordinato) contenente numeri interi positivi.

Scrivere nel file `esercizio3.cc` le dichiarazioni e le definizioni delle seguenti tre funzioni:

- `creaAlberoDiRicercaBinario`: questa funzione prende come unico argomento la radice dell'albero creato dalla funzione `generaAlbero`. La funzione `creaAlberoDiRicercaBinario` deve creare un *albero di ricerca binario* contenente solo i numeri **pari** contenuti nell'albero fornito come argomento e ritornare la radice dell'albero di ricerca binario appena creato. **Suggerimento**: utilizzare la struttura dati già definita nel file `esercizio3.cc` per costruire l'albero di ricerca binario;

- `stampaAlberoInOrdine`: questa funzione prende come unico argomento la radice dell'albero di ricerca binario appena creato. La funzione `stampaAlberoInOrdine` deve stampare a video (su un'unica riga) gli elementi dell'albero di ricerca binario in ordine **crescente**;

- `deallocaAlbero`: questa funzione prende come unico argomento la radice di un albero binario e ne dealloca **tutti** i nodi.

Questi sono due esempi di esecuzione:

```
computer > ./a.out
L'albero iniziale è: 4 0 5 0 0 7 3 7 0 6
L'albero di ricerca binario è: 0 0 0 0 4 6

computer > ./a.out
L'albero iniziale è: 9 3 5 2 6 5 4 6 6 0
L'albero di ricerca binario è: 0 2 4 6 6 6 9
```

Note:

- Scaricare il file `esercizio3.cc`, modificarlo per inserire la corretta definizione e implementazione delle funzioni `creaAlberoDiRicercaBinario`, `stampaAlberoInOrdine` e `deallocaAlbero` e infine caricare il file risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito;

- Notare che il file `esercizio3.cc` contiene già la definizione della funzione `main`, insieme a un'ulteriore funzione ausiliaria `stampaAlbero` che stampa graficamente a video un albero binario (nel caso in cui possa esservi utile visualizzare l'albero);

- E' consentito definire e implementare funzioni ausiliarie che possano aiutarvi nella soluzione del problema;
- All'interno di questo programma non è ammesso l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `iostream` e `cstdlib`.

3 Esercizio 3_3

Il file `esercizio3.cc` contiene la definizione di una struttura dati rappresentante il nodo di un albero binario, insieme all'implementazione della funzione `generaAlbero` che crea un albero binario (non ordinato) contenente numeri interi positivi.

Scrivere nel file `esercizio3.cc` le dichiarazioni e le definizioni delle seguenti tre funzioni:

- `creaAlberoDiRicercaBinario`: questa funzione prende come unico argomento la radice dell'albero creato dalla funzione `generaAlbero`. La funzione `creaAlberoDiRicercaBinario` deve creare un *albero di ricerca binario* contenente solo i numeri **dispari** contenuti nell'albero fornito come argomento e ritornare la radice dell'albero di ricerca binario appena creato. **Suggerimento**: utilizzare la struttura dati già definita nel file `esercizio3.cc` per costruire l'albero di ricerca binario;

- `stampaAlberoInOrdine`: questa funzione prende come unico argomento la radice dell'albero di ricerca binario appena creato. La funzione `stampaAlberoInOrdine` deve stampare a video (su un'unica riga) gli elementi dell'albero di ricerca binario in ordine **decrescente**;

- `deallocaAlbero`: questa funzione prende come unico argomento la radice di un albero binario e ne dealloca **tutti** i nodi.

Questi sono due esempi di esecuzione:

```
computer > ./a.out
L'albero iniziale è: 9 6 9 7 1 2 7 4 9 3
L'albero di ricerca binario è: 9 9 9 7 7 3 1

computer > ./a.out
L'albero iniziale è: 6 5 4 5 6 2 7 6 4 8
L'albero di ricerca binario è: 7 6 5 5
```

Note:

- Scaricare il file `esercizio3.cc`, modificarlo per inserire la corretta definizione e implementazione delle funzioni `creaAlberoDiRicercaBinario`, `stampaAlberoInOrdine` e `deallocaAlbero` e infine caricare il file risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito;

- Notare che il file `esercizio3.cc` contiene già la definizione della funzione `main`, insieme a un'ulteriore funzione ausiliaria `stampaAlbero` che stampa graficamente a video un albero binario (nel caso in cui possa esservi utile visualizzare l'albero);

- E' consentito definire e implementare funzioni ausiliarie che possano aiutarvi nella soluzione del problema;
- All'interno di questo programma non è ammesso l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `iostream` e `cstdlib`.

4 Esercizio 3_4

Il file `esercizio3.cc` contiene la definizione di una struttura dati rappresentante il nodo di un albero binario, insieme all'implementazione della funzione `generaAlbero` che crea un albero binario (non ordinato) contenente numeri interi positivi.

Scrivere nel file `esercizio3.cc` le dichiarazioni e le definizioni delle seguenti tre funzioni:

- `creaAlberoDiRicercaBinario`: questa funzione prende come unico argomento la radice dell'albero creato dalla funzione `generaAlbero`. La funzione `creaAlberoDiRicercaBinario` deve creare un *albero di ricerca binario* contenente solo i numeri **pari** contenuti nell'albero fornito come argomento e ritornare la radice dell'albero di ricerca binario appena creato. **Suggerimento**: utilizzare la struttura dati già definita nel file `esercizio3.cc` per costruire l'albero di ricerca binario;

- `stampaAlberoInOrdine`: questa funzione prende come unico argomento la radice dell'albero di ricerca binario appena creato. La funzione `stampaAlberoInOrdine` deve stampare a video (su un'unica riga) gli elementi dell'albero di ricerca binario in ordine **decrescente**;

- `deallocaAlbero`: questa funzione prende come unico argomento la radice di un albero binario e ne dealloca **tutti** i nodi.

Questi sono due esempi di esecuzione:

```
computer > ./a.out
L'albero iniziale è: 2 7 5 1 4 9 2 1 9 7
L'albero di ricerca binario è: 4 2 2

computer > ./a.out
L'albero iniziale è: 7 4 8 7 3 0 2 2 1 0
L'albero di ricerca binario è: 8 7 4 2 2 0 0
```

Note:

- Scaricare il file `esercizio3.cc`, modificarlo per inserire la corretta definizione e implementazione delle funzioni `creaAlberoDiRicercaBinario`, `stampaAlberoInOrdine` e `deallocaAlbero` e infine caricare il file risultato delle vostre modifiche a soluzione di questo esercizio nello spazio apposito;

- Notare che il file `esercizio3.cc` contiene già la definizione della funzione `main`, insieme a un'ulteriore funzione ausiliaria `stampaAlbero` che stampa graficamente a video un albero binario (nel caso in cui possa esservi utile visualizzare l'albero);

- E' consentito definire e implementare funzioni ausiliarie che possano aiutarvi nella soluzione del problema;
- All'interno di questo programma non è ammesso l'utilizzo di variabili globali o di tipo `static` e di funzioni di libreria al di fuori di quelle definite in `iostream` e `cstdlib`.