# ASD 23/24 - Laboratorio 6

In questo laboratorio viene richiesto di proporre una nuova implementazione per il TDD Dizionario di coppie (chiave: `string`, valore: `string`) usando alberi binari di ricerca nei quali le chiavi saranno usate per ordinare i nodi del albero.

## 1 Materiale dato

Nel file `asd-lab6-traccia.zip`, trovate:

- Un file `dictionary.h` contenente le intestazioni delle funzione da implementare
- Un file `dictionary-bst.cpp` dove dovete scrivere l'implementazione delle funzioni richieste e anche definire il tipo della struttura
- Un file `dictionary-main.cpp` contenente un programma principale per testare le funzione via menù (lo stesso che al Laboratorio 5)
- Un file `dictionary-test.cpp` contenente un programma principale che avvia una sequenza di test automatici
- Nella cartella `engeng` si trovano diversi file `.txt` con la descrizione di 'dizionario' chiave-valore'.

## 2 Funzioni da implementare

Lo scopo di questa laboratorio è di implementare, come nel Laboratorio 5, tutte le operazioni 'standard' del TDD Dizionario e anche la funzione di stampa corrispondente ad una visita Depth First Search (DFS) simmetrica dell'albero. Le funzioni di lettura da file e stdin vi sono fornite e sono uguali a quelle date al Laboratorio 5. Dovete in più fornire nel file `dictionary-bst.cpp` l'implementazione della struttura `struct bstNode` che corrisponde ai nodi dell' albero.

L'implementazione delle operazioni dovrà essere fatta con funzioni/procedure ricorsive che operino in $\theta(h)$ nel caso peggiore dove $h$ è l'altezza dell'albero. È inoltre vietato aggiungere un puntatore al padre alla struct che rappresenta un nodo.

I prototipi delle funzioni da implementare sono forniti nel file `dictionary.h` come descritto qui sotto e dovrete realizzare l'implementazione nel file `dictionary-bst.cpp`. Notate che la struttura `struct bstNode` non è data e deve essere scritta da voi sempre in questo ultimo file.

```
namespace dict {

  // Codici di errore
  enum Error {OK, FAIL};

  // Tipi e costanti
  typedef string Key;         // tipo base
  typedef string Value;       // tipo base

  const Key emptyKey = "###RESERVED KEYWORD### EMPTY KEY";
  const Value emptyValue = "###RESERVED KEYWORD### EMPTY VALUE";

  typedef struct {
    Key key;
    Value value;
  } Elem;

  // Implementazione basata su albero binario di ricerca
  struct bstNode; // forward declaration

  typedef bstNode* Dictionary;

  const Dictionary emptyNode = NULL;
  const Dictionary emptyDictionary = NULL;

  //Prototipi di funzione da implementare
  Error insertElem(const Key, const Value, Dictionary&);
  Error deleteElem(const Key, Dictionary&);
  Value search(const Key, const Dictionary&);
  Dictionary createEmptyDict();
  bool isEmpty(const Dictionary&);
```

```
}  // end namespace Dict

//Funzione da implementare
void print(const dict::Dictionary&);
```

## 3 Tests manuali

Il file `dictionary-main.cpp` contiene il `main` di un programma per aiutarvi a svolgere dei tests, che è lo stesso che avete visto nel Laboratorio 5.
Per potere usare questo programma con la vostra nuova implementazione, potete compilarlo così:
`g++ -std=c++11 -Wall dictionary-bst.cpp string-utility.cpp dictionary-main.cpp -o dictionary-main`
e poi eseguirlo con `./dictionary-main`.

Dovete usare questo programma per aggiungere al file `tempi-di-esecuzione-operazioni-dict.xls`, che vi è stato dato al Laboratorio 5, i tempi di esecuzione delle singole operazioni relative all'albero binario di ricerca.

## 4 Tests automatici

Nel file `dictionary-test.cpp`, abbiamo programmato una sequenza di tests che si eseguono automaticamente grazie i quali verifichiamo che le funzioni implementate si comportino bene. Per usare questo programma invece, potete compilare così:
`g++ -std=c++11 -Wall dictionary-bst.cpp string-utility.cpp dictionary-test.cpp -o dictionary-test`
e poi eseguirlo con `./dictionary-test`.

## 5 Consegna

Per la consegna, creare uno `zip` con tutti i file forniti; in particolare con i file `dictionary.h`, `dictionary-bst.cpp` da voi modificati e il file `tempi-di-esecuzione-operazioni-dict.xls` completato con i nuovi dati.