作业 2（BST 排序算法实现）

陈俊铭 3210300364 （信息与计算科学）

2022 年 10 月 19 日

1 创建要求

编写 BSTsorting 函数实现对数组的排序，且必须使用二叉搜索树排序算法。有两种不同的模式，为乱序后排序和不乱序直接排序，使用两种方法测试函数运行时间。

2 设计思路

1. 编写 BSTSorting 函数的数组排序。
2. 编写 main 函数对头文件进行测试。
3. 进行脚本文件编写。

3 添加函数

```
1.  #include <iostream>
#include <vector>
#include <ctime>
#include "BinarySearchTree.h"
using namespace std;
template <typename Comparable>

void BSTSorting(vector<Comparable> &_arr, int _mode = 0)
{
  BinarySearchTree<Comparable> bst;
  int start, finish;
  int time = 0;
```

```
  //当 _mode = 0 时, 不乱序
  if(_mode == 0)
  {
    start = clock();
    for(int i = 0; i <_arr.size(); ++i)
    {
      bst.insert(_arr[i]);
    }
    finish = clock();
    time = (double)(finish-start)/CLOCKS_PER_SEC;
  }
  //当 _mode = 1 时, 先对数组 _arr 乱序后再排序。
  else if(_mode == 1)
  {
    for(int j = 0; j < 100; ++j)
    {
      bst.makeEmpty();
      start = clock();
      for(int i = _arr.size()-1; i >= 1; --i)
      {
        int k = rand()%i;
        Comparable t = _arr[k];
        _arr[k] = _arr[i];
        _arr[i] = t;
      }
      for(int i = 0; i < _arr.size(); ++i)
      {
        bst.insert(_arr[i]);
      }
      finish = clock();
      time += (double)(finish-start)/CLOCKS_PER_SEC;
    }
  }

  cout << "BStSorting time: " << time << "s" << endl;
}
```

```
2. int main()
{
    int t, mode;
    cout << "Enter Array:"; cin >> t;
    cout << "Enter mode:"; cin >> mode;
    if(mode!=0 && mode !=1)
    {
        cout << "Enter 0 or 1.\n";
    }
    vector<int> _arr={};
    for(int t = 0; t<=1; t++)
    {
        _arr.push_back(t);
    }
    BSTSorting(_arr,mode);
        return 0;
}
```