Name : Satyam Sharma
Project Report : Optimal Binary Search Tree
Course : CSC 220
Prof. Stephen Lucci

Files in the project

BST.cpp (contains ADT for BSTNode)

```
class BSTNode {...};
//Node with a key value, and pointer to left and right BSTNode
```

optimal.cpp (contains ADT for BSTNode)

Global variables

```
int keys[] = {10, 12, 20};
//Keys have to be ordered in a  non-decreasing manner

double p[] = {.4, .1, .5};
//Probablities corresponding to the keys
//where sum of p equals 1
//however, it can also be used a frequncy array

const int n = 3;
//number of elements in the array

double costMatrix[n][n];
// costMatrix[i][j] stores optimal cost for key_i to key_j

int    rootMatrix[n][n];
// rootMatrix[i][j] stores optimal root for key_i to key_j
```

Functions

```
void optsearchtree( );
//Finds cheapest cost from Key_i to Key_j, and stores them in costMatrix
//and stores the key that creates cheapest root in rootMatrix
//Based on dynamic programmin (Bottom-Up), starts filling the diagonal values first
//first
// O(n^4)

double pSum(int i, int j);
//To calculate Sum of probablities required by optsearchtree.
// O(n)

BSTNode* createBST(int i, int j);
//Creates an optimal Binary serach tree based on info from rootMatrix[n][n]
//Possible Bug: Sometimes crashes the program
// O(n)

BSTNode* search (BSTNode* tree, int key);
//searches a key in a tree. If it is found it returns a pointer to it, otherwise NULL.
//O(lg n)
```