# პროგრამირების დავალება ბიარული ძებნის ხეებზე

ეს პროგრამა მოითხოვს თქვენგან შეიმუშაოთ მინარული ძების ხე რომლისთვისაც განსაზღვრული იქნება შემდეგი ოპერაციები:

1) კვანძის დამატება
2) კვანძის წაშლა
3) ელემენტის ძიება
4) ხის შემოვლა 3 ნაირად: Preorder, Inorder და Postorder
5) ხის ელემენტების ბეჭდვა preorder შემოვლის გამოყენებით

დავალება უნდა შედგენოდეს შემდეგი ფაილებისაგან:
TreeNode.h
BST.h
BST.cpp
Driver.cpp

TreeNode.h ფაილში უნდა გქონდეთ კლასი TreeNode. მას უნდა ჰყავდეს BST როგორც მისი მეგობარი კლასი, დამატებით მასში უნდა იყოუს კლასის კონსტრუქტორი და აქსესორ ფუნქციები (სე ჩვენ თავიდან ავიცილებთ TreeNode.cpp - ს).
მაგალითად შესაძლებელია ქონდეს ფორმა:

```
Class TreeNode {
                    Friend class BST;
Public:
  TreeNode();  //უპარამეტრო კონსტრუქტორი
  TreeNode(int i, TreeNode* L = 0; TreeNode* R = 0);
//პარამეტრიანი კონსტრუქტორი
  int getItem () const; // ფუნქცია-აქსესორი

private:
 int item;
  TreeNode *Lchild;
  TreeNode *Rchild;

};

TreeNode::TreeNode()
{
   Lchild = Rchild = NULL;
}

TreeNode::TreeNode(int i, TreeNode *L = 0, TreeNode *R = 0) : item(i), Lchild(L),
Rchild(R)
{}
```

```
int TreeNode::getItem() const
{ return item;}
```

BST.h უნდა გვქონდეს  BST კლასის აღწერა, დაგჭირდებათ ასევე გქონდეთ 4 ე.წ. უტილიტი ფუნქციები რომლებიც ძებნისა და ხის შემოვლისათვის დაგჭირდებათ.
BST.cpp ში უნდა იყოს ყველა იმ public ფუნქციებისიმპლემენტაცია რომლებიც BST.h გაქვთ დეკლარირებული
driver.cpp -ში უნდა იყოს ყველა იმ ფუნქციონალის ტესტირება რომელიც BST-სთვის გაქვთ
P.S. კლასი უნდა იყოს ზოგადი - ანუ TEMPLATE (ყურადღება მიაქციეთ რომ მაგალითში მოყვანილი გაქვთ არატემპლეიტ ვარიანტი…)