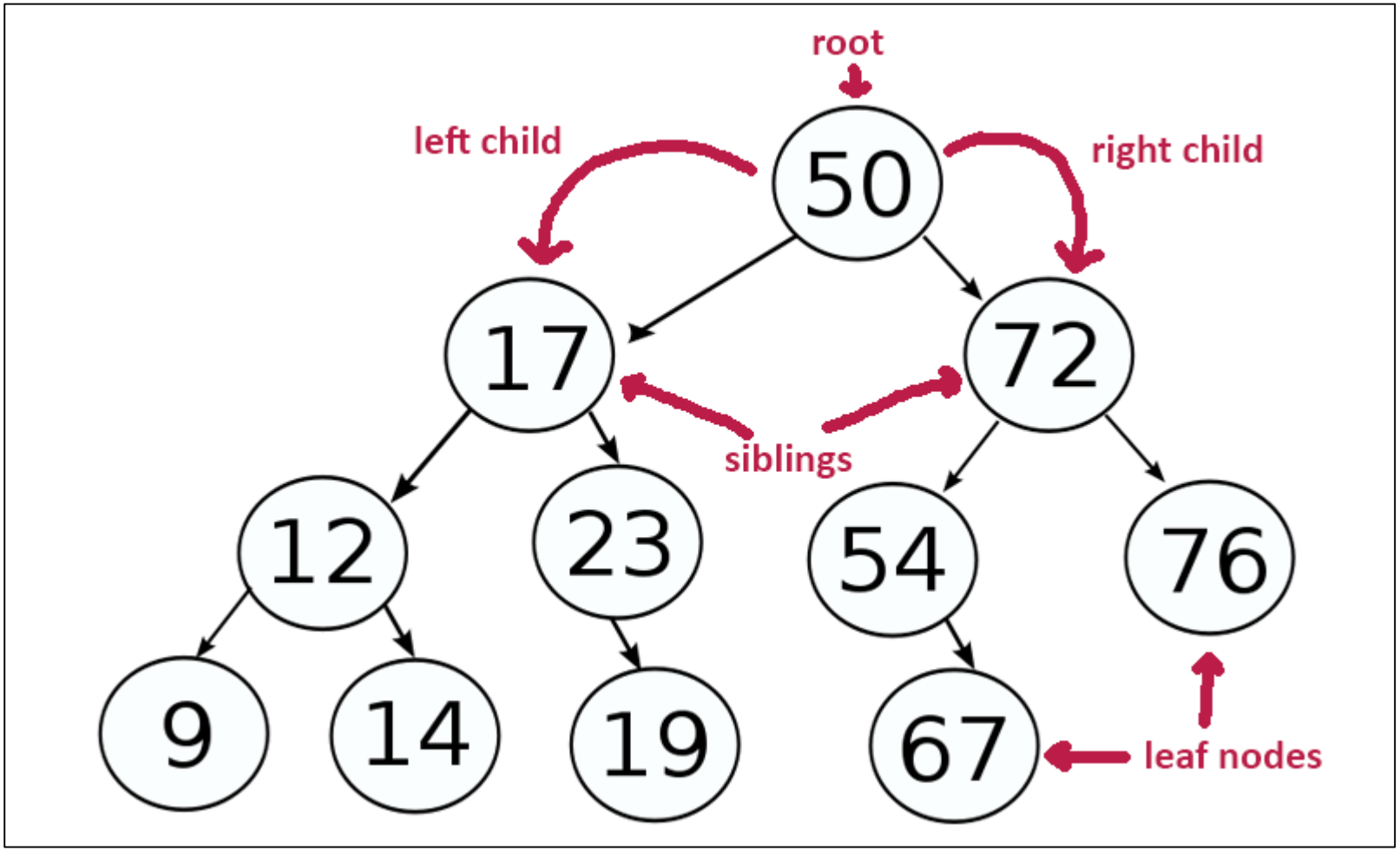
S08 – SD

Alin Zamfiroiu
alin.zamfiroiu@csie.ase.ro

Binary tree



http://www.bluesharktutorials.com/

Used structures

```
struct tara{
    int id;
    char* nume;
    int nr_locuitori;
    float suprafata;
};
```

```
struct nod{
    tara info;
    nod* st;
    nod* dr;
};
```

- To create a new node we need the useful information, the left root and the right root.
- In many cases the left root and the right root will be NULL.

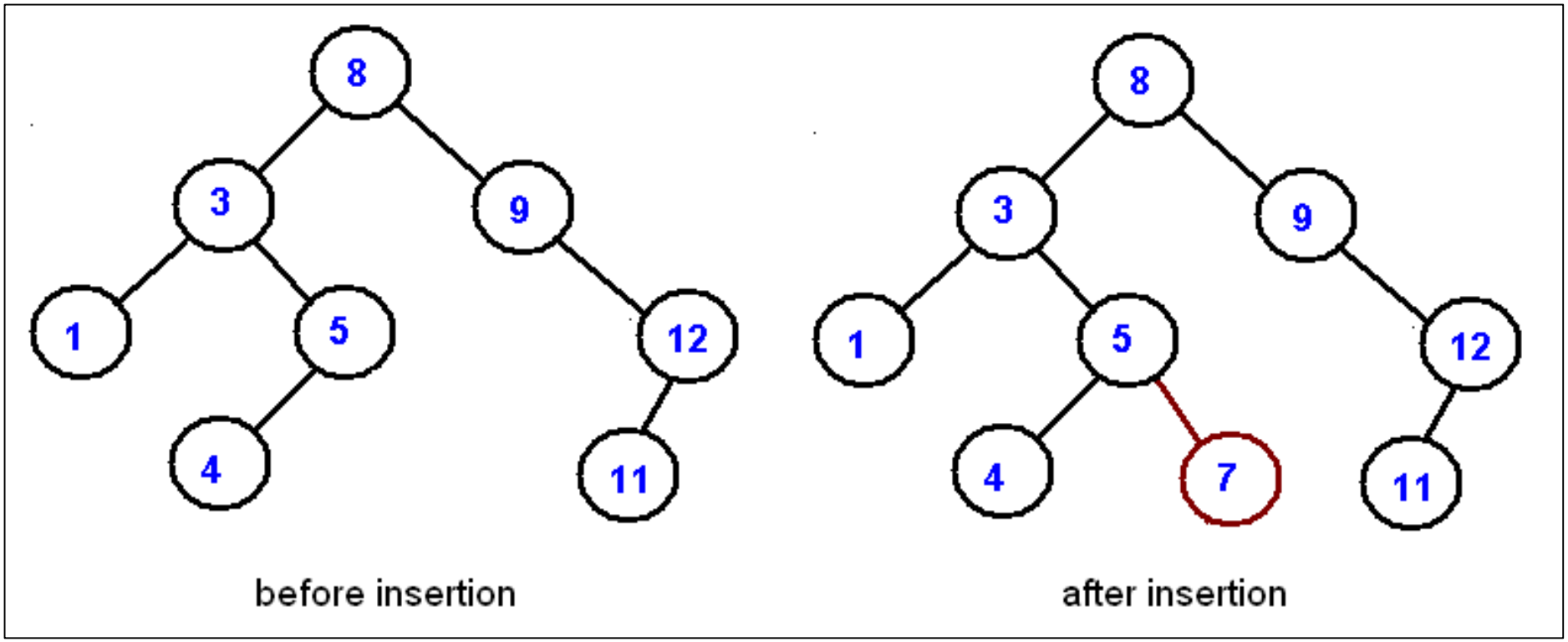
Creation node function

```
nod* creareNod(tara info, nod* st,nod*dr){
        nod* temp=(nod*)malloc(sizeof(nod));
        temp->info.id=info.id;
        temp->info.nume=(char*)malloc((strlen(info.nume)+1)*sizeof(char));
        strcpy(temp->info.nume,info.nume);
        temp->info.nr_locuitori=info.nr_locuitori;
        temp->info.suprafata=info.suprafata;

        temp->dr=dr;
        temp->st=st;

        return temp;
}
```

Insertion



http://www.cs.cmu.edu/

Insertion

```
nod* inserare(nod*radacina, tara info){
    if(radacina==NULL)
    {
        radacina=creareNod(info,NULL,NULL);
        return radacina;
    }
    else
    {
        if(info.id < radacina->info.id)
        {
            radacina->st=inserare(radacina->st,info);
        }
        else
        {
            radacina->dr=inserare(radacina->dr,info);
        }
        return radacina;
    }
}
```

Binary tree crossing

- Preorder → **R**SD;
- Inorder → S**R**D;
- Postorder →SD**R.**

Binary tree crossing

```
void SRD(nod*rad)
{
    if(rad)
    {
        SRD(rad->st);
        printf("%d.%s are %d locuitori si o supr
        SRD(rad->dr);
    }
}
```

Tree height

- The height of the current tree will be maximum of height of left tree and height of left tree plus 1.

```
int max(int a,int b)
{
    if(a>b) return a;
    else
        return b;
}
```

Tree height

```
int nivele(nod*rad)
{
    if(rad)
        return 1+max(nivele(rad->st),nivele(rad->dr));
    else
        return 0;
}
```

Display nodes from a level

- Create a function with three parameters: tree root, the level that we want to display information and the level of the root (level 1).
- Every call for both the left subtree and the right subtree current level is increased by one.
- When the current level is equal to the searched level we display the information from that node.

Display nodes from a level

```
void afisareNivel(nod*rad,int nivel,int nivelCurent)
{
    if(rad)
    {
        if(nivel==nivelCurent)
        {
            printf("%d.%s are %d locuitori si o suprafat
        }
        else
        {
            afisareNivel(rad->st,nivel,nivelCurent+1);
            afisareNivel(rad->dr,nivel,nivelCurent+1);
        }
    }
}
```

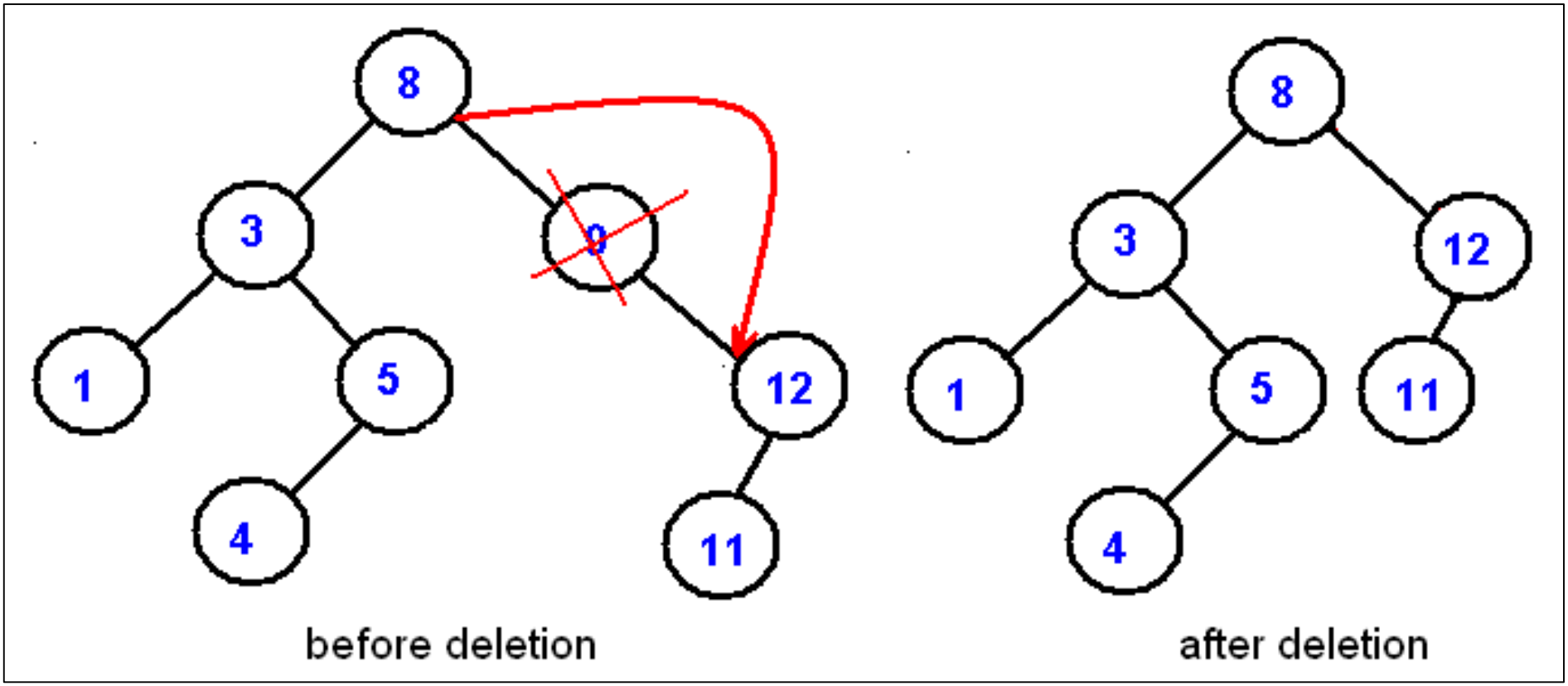
Search a node

- We search in a tree by country ID.
- Searching is made by country ID as the tree is sorted by this ID.

Search a node

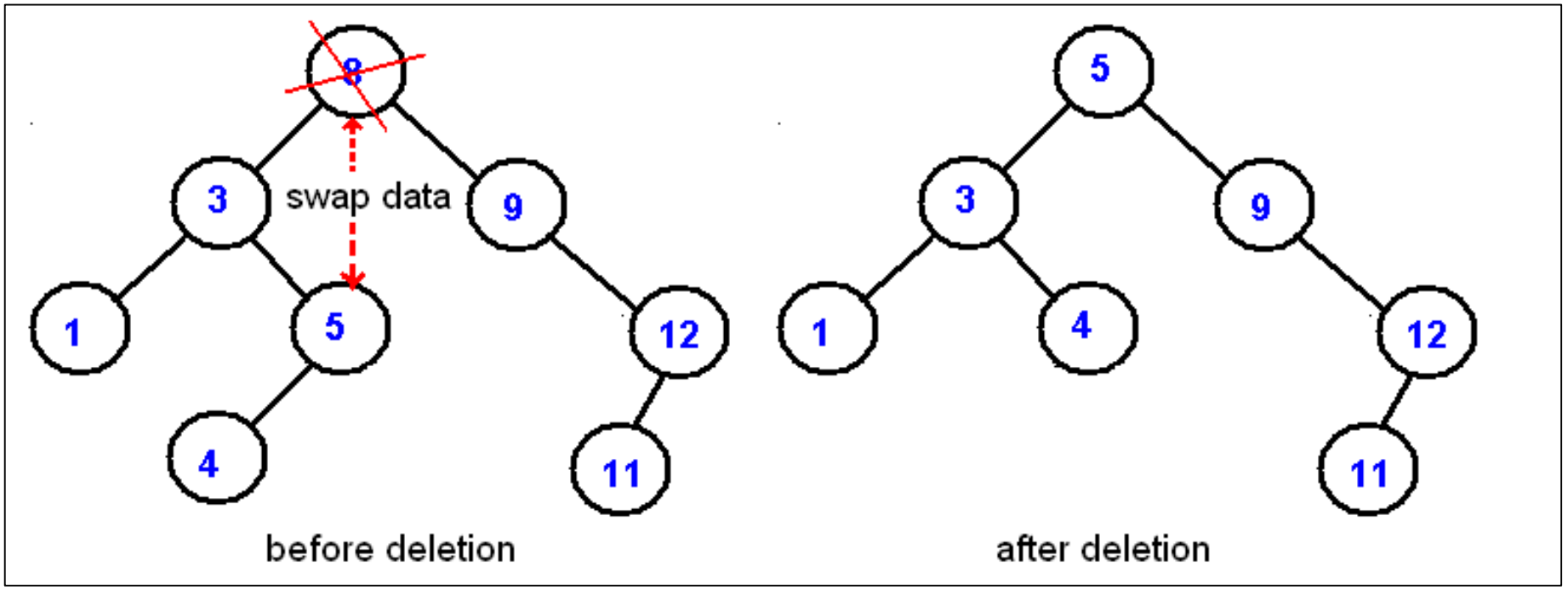
```
nod* cauta(nod*rad, int cod)
{
    if(rad)
    {
        if(rad->info.id==cod)
        {
            return rad;
        }
        else
        {
            if(rad->info.id>cod)
                return cauta(rad->st,cod);
            else
                return cauta(rad->dr,cod);
        }
    }
}
```

Delete a node



http://www.cs.cmu.edu/

Delete a node



http://www.cs.cmu.edu/