Treap Rotation

Problem Statement

This is a function-only problem. You need to implement the following `rotate_dir` function.

```
node *rotate_dir(node *r, bool dir);
```

Description:

- If `dir` is true,
  then the function performs a right rotation **on the subtree rooted at `r`**,
  i.e., on the node `r` and its left child node.
- If `dir` is false,
  then the function performs a left rotation **on the subtree rooted at `r`**,
  i.e., on the node `r` and its right child node.

It is guaranteed that the corresponding child node of `r` exists for the required operation.
When the operation is done, the function **returns the pointer to the new root of the subtree**.

You should assume the following declaration of the `node` struct (without redeclaring it).
Note that, you **should not redelare this struct** in your submitted program code.

```
struct node {
    char c;
    int pri, l_sz, r_sz;
    // pri, size of left child and right child
    node *lc, *rc, *p;
    // pointer to left child, right child, parent
};
```

Requirements:

- The function should run in $O(1)$ time.

- Along with the rotation, your function must update the corresponding fields, e.g., the pointers and the auxiliary fields such as `l_sz`, `r_sz`, etc.
- The function must return a pointer to new root of the subtree (which was originally rooted at `r`). Note that this is exactly the parent node of `r` after the rotation.

Note that, if a pointer does not reference any node, it is set to be `NULL` or `nullptr`.

Submission Instructions

This is a function implementation task. Your submitted code must include the following identifier:

```
/* probID: W9-A1-Rotation */
```

Your submitted code must include the implementation of the `rotate_dir` function and additional functions/declarations if necessary, but must not the `main` function.

When submitting, choose the language `C++ - function only`.