Code Explanation

To solve the orientation problem for a person in x,y plan to have a visual of maximum number of trees at the set location we need to do as following

1. Calculate the angular difference from each tree to the person.
2. If the difference is less than half of the field of view angle alpha then the tree is in sight.
3. Rota the person by one degree and calculate the number of trees in sight.
4. Choose the orientation with the maximum number of trees in sight.

Now I implemented the code as follows

Declare the necessary variables

```
double alpha;
double maxTreesInSight = 0;
double bestOrientation = 0;  // Best orientation for the person
int treesInSight_PV_angle=0;
int maxTreesInSight_PV_angle=0;
```

As required ask the user for the input person x,y and orientation in the form of tuple of real numbers using std::cin

```
// Read input from stdin
std::tuple<double, double,double> person;
std::vector<std::tuple<double, double>> trees;
std::cout<<"Please enter Field of view angle alpha"<<std::endl;
std::cin>>alpha;
std::cout<<"Please enter person x ,y and Person_view_angle values"<<std::endl;
// Read person's position and  person_view_angle from input
std::cin >> std::get<0>(person) >> std::get<1>(person) >> std::get<2>(person);
std::cout<<"person x\t"<<std::get<0>(person)<<"person
y\t"<<std::get<1>(person)<<"person view
angle\t"<<std::get<2>(person)<<std::endl;
//////
int numTrees;
std::cout<<"Please enter total number of tress\n";
std::cin >> numTrees;
```

Once the persons details and make number of trees is collected from the user , I created a vector of tuple and added values of x,y coordinates of tree in the vector using the following code

```
for (int i = 0; i < numTrees; ++i) {
   std::tuple<double, double> tree;
   std::cout<<"Please tree "<<i+1<<" x and y coordinates\n";
   std::cin >> std::get<0>(tree) >> std::get<1>(tree);
   trees.push_back(tree);
}
std::cout<<"All tress added"<<std::endl;
```

Once all the required inputs are acquired I called `isTreeInFieldOfView` which I defined in a separate function.cpp file as follows

```
bool isTreeInFieldOfView(const std::tuple<double, double,double>& person, const
std::tuple<double, double>& tree, double orientation, double halfAlpha) {
   double angleToTree = atan2(std::get<1>(tree) - std::get<1>(person),
std::get<0>(tree) - std::get<0>(person)) * 180.0 / M_PI;
   double angleDifference = fabs(orientation - angleToTree);
  // std::cout<<"angle diff"<<angleDifference<<std::endl;
   return angleDifference <= halfAlpha;
```

This function receives the person data and tree data in the form of tuple. Than we calculate the arctan2 function to get the angle to the tree and from this we calculate the angle difference form the orientation of the person. The function returns a bool value of true if the angle difference is less the half of the field of view angle alpha.

Now in main function first we send the input values from user to this function to get the amount of trees in the field of view as follows

```
double orint=std::get<2>(person);
for (const auto& tree : trees) {

   double halfAlpha = alpha / 2.0;
       if (isTreeInFieldOfView(person, tree, orint, halfAlpha)) {
           treesInSight_PV_angle++;
           std::cout<<"here\n";
       }
   }
maxTreesInSight_PV_angle=treesInSight_PV_angle;
std::cout<<"max tress at person view angle
"<<maxTreesInSight_PV_angle<<std::endl;
```

The reason of doing this is to know if its possible to have a maximum number of trees insight without rotating

Then after this we traverse through all the possible 360 degrees to look if there is any better orientation in which we get more trees in sight.

```
// Iterate through possible orientations
for (double orientation = 0; orientation < 360; orientation += 1.0) {
   int treesInSight = 0;
   double halfAlpha = alpha / 2.0;

   // Check each tree if it's within the field of view
   for (const auto& tree : trees) {
       if (isTreeInFieldOfView(person, tree, orientation, halfAlpha)) {
           treesInSight++;
       }
   }

   // Update the best orientation if more trees are in sight
   if (treesInSight > maxTreesInSight ) {
       maxTreesInSight = treesInSight;
       bestOrientation = orientation;
       if(treesInSight <=maxTreesInSight_PV_angle){
       bestOrientation=orint;
   }
   }

}
```

In this block while traversing we keep on checking if the max number of trees in sight have changed in each iteration or not. If they have increased we set maxtress insights to the number of trees in view in the iteration and update the best orientation to the iteration orientation. We also check if the trees in sights are less or equal to trees in sight at the initial location. If this condition is true we do not update the orientation of the person.

Lastly we print the output using std::cout for best orientation, maximum number of tress in sight and lastly we if condition to look if the person needs to move to get the most number of tress insight or not.

```
std::cout << "Best orientation: " << bestOrientation <<" degress from +iv x
axis "<< "\nmax number of tree in sight "<<maxTreesInSight<< std::endl;
//Outputs does the person need to rotate or not
if(bestOrientation==orint)
{
   std::cout<<"The person is at the optimal point of view"<<std::endl;
}
else{
   std::cout<<"The person need to move to "<<bestOrientation<<" degrees from
the +iv X axis to have a view with most number of tress"<<std::endl;
}
```