Practical 1 (A)

**Parallel BFS using OpenMP**

#include<iostream> #include<stdlib.h> #include<queue> using namespace std;

class node

{

public:

node \*left, \*right; int data;

};

class Breadthfs

{

public:

node \*insert(node \*, int); void bfs(node \*);

};

node \*insert(node \*root, int data)

// inserts a node in tree

{

if(!root)

{

root=new node; root->left=NULL; root->right=NULL; root->data=data; return root;

}

queue<node \*> q; q.push(root);

while(!q.empty())

{

node \*temp=q.front(); q.pop();

if(temp->left==NULL)

{

}

else

{

temp->left=new node; temp->left->left=NULL; temp->left->right=NULL; temp->left->data=data; return root;

q.push(temp->left);

}

if(temp->right==NULL)

{

temp->right=new node; temp->right->left=NULL; temp->right->right=NULL; temp->right->data=data; return root;

}

else

{

q.push(temp->right);

}

}

}

void bfs(node \*head)

{

queue<node\*> q; q.push(head);

int qSize;

while (!q.empty())

{

qSize = q.size(); #pragma omp parallel for

//creates parallel threads

for (int i = 0; i < qSize; i++)

{

node\* currNode; #pragma omp critical

{

currNode = q.front(); q.pop();

cout<<"\t"<<currNode->data;

}// prints parent node #pragma omp critical

{

if(currNode->left)// push parent's left node in queue q.push(currNode->left);

if(currNode->right)

>right);

node in queue

q.push(currNode-

}// push parent's right

node?";

root=insert(root,data);

cout<<"do you want insert one more cin>>ans;

}

} }while(ans=='y'||ans=='Y');

} bfs(root);

int main(){

node \*root=NULL; int data;

char ans;

do

{

cout<<"\n enter data=>"; cin>>data;

return 0;

}

Run Commands:

1. g++ -fopenmp bfs.cpp -o bfs
2. ./bfs

OUTPUT:

