**GROUP D**

|  |  |
| --- | --- |
|  | **Given sequence k = k1 <k2 < ... < kn of n sorted keys, with a search probability** |
|  | **pi for each key ki . Build the Binary search tree that has the least search cost given the access** |
|  | **probability for each key.** |
|  |  |
|  | #include<iostream> |
|  | using namespace std; |
|  | void con\_obst(void); |
|  | void print(int,int); |
|  | float a[20],b[20],wt[20][20],c[20][20]; |
|  | int r[20][20],n; |
|  | int main() |
|  | { |
|  | int i; |
|  | cout<<"\n\*\*\*\*\*\* PROGRAM FOR OBST \*\*\*\*\*\*\n"; |
|  | cout<<"\nEnter the no. of nodes : "; |
|  | cin>>n;cout<<"\nEnter the probability for successful search :: "; |
|  | cout<<"\n————————————————\n"; |
|  | for(i=1;i<=n;i++) |
|  | { |
|  | cout<<"p["<<i<<"]"; |
|  | cin>>a[i]; |
|  | } |
|  | cout<<"\nEnter the probability for unsuccessful search :: "; |
|  | cout<<"\n————————————————–\n"; |
|  | for(i=0;i<=n;i++) |
|  | { |
|  | cout<<"q["<<i<<"]"; |
|  | cin>>b[i]; |
|  | } |
|  | con\_obst(); |
|  | print(0,n); |
|  | cout<<endl; |
|  | } |
|  | void con\_obst(void) |
|  | { |
|  | int i,j,k,l,min; |
|  | for(i=0;i<n;i++) |
|  | { //Initialisation |
|  | c[i][i]=0.0; |
|  | r[i][i]=0; |
|  | wt[i][i]=b[i]; |
|  | // for j-i=1 can be j=i+1 |
|  | wt[i][i+1]=b[i]+b[i+1]+a[i+1]; |
|  | c[i][i+1]=b[i]+b[i+1]+a[i+1]; |
|  | r[i][i+1]=i+1; |
|  | } |
|  | c[n][n]=0.0; |
|  | r[n][n]=0; |
|  | wt[n][n]=b[n]; |
|  | //for j-i=2,3,4....,n |
|  | for(i=2;i<=n;i++) |
|  | { |
|  | for(j=0;j<=n-i;j++) |
|  | { |
|  | wt[j][j+i]=b[j+i]+a[j+i]+wt[j][j+i-1]; |
|  | c[j][j+i]=9999; |
|  | for(l=j+1;l<=j+i;l++) |
|  | { |
|  | if(c[j][j+i]>(c[j][l-1]+c[l][j+i])) |
|  | { |
|  | c[j][j+i]=c[j][l-1]+c[l][j+i]; |
|  | r[j][j+i]=l; |
|  | } |
|  | } |
|  | c[j][j+i]+=wt[j][j+i]; |
|  | } |
|  | cout<<endl; |
|  | } |
|  | cout<<"\n\nOptimal BST is :: "; |
|  | cout<<"\nw[0]["<<n<<"] :: "<<wt[0][n]; |
|  | cout<<"\nc[0]["<<n<<"] :: "<<c[0][n]; |
|  | cout<<"\nr[0]["<<n<<"] :: "<<r[0][n]; |
|  | } |
|  | void print(int l1,int r1) |
|  | { |
|  | if(l1>=r1) |
|  | return; |
|  | if(r[l1][r[l1][r1]-1]!=0) |
|  | cout<<"\n Left child of "<<r[l1][r1]<<" :: "<<r[l1][r[l1][r1]-1]; |
|  | if(r[r[l1][r1]][r1]!=0) |
|  | cout<<"\n Right child of "<<r[l1][r1]<<" :: "<<r[r[l1][r1]][r1]; |
|  | print(l1,r[l1][r1]-1); |
|  | **print(r[l1][r1],r1);**  **OUTPUT:**  **\*\*\*\*\*\* PROGRAM FOR OBST \*\*\*\*\*\***  **Enter the no. of nodes : 5**  **Enter the probability for successful search ::**  **————————————————**  **p[1]0.5**  **p[2]0.8**  **p[3]0.2**  **p[4]0.7**  **p[5]0.002**  **Enter the probability for unsuccessful search ::**  **————————————————–**  **q[0]0.2**  **q[1]0.30**  **q[2]0.2**  **q[3]0.40**  **q[4]0.8**  **q[5]0.3**  **Optimal BST is ::**  **w[0][5] :: 4.402**  **c[0][5] :: 9.904**  **r[0][5] :: 4**  **Left child of 4 :: 2**  **Right child of 4 :: 5**  **Left child of 2 :: 1**  **Right child of 2 :: 3** |