Skip List

S.R.PUNEETH
1BM18CS087
30/9/20

```
int randomLevel(){
float r = (float) rand() / RAND_MAX;
int lvl = 0;
while (r < P && lvl < max lvl)
{
 lvl++;
 r = (float)rand()/RAND_MAX;
 }
 return lvl;
};
void insertElem(int key) {
 Node* current = header;
 Node * update (MAX LVL+1);
 memset (update, 0, sizeof(Node * ) * (MAXLVL+1));
 for (int i = level ; i>=0 ; i--) {
 while (current -> forward (i) ! = NULL &&
         current -> forward (i) -> key < key)
   Current = current-> forward(i);
  update(i) = current;
 }
 current = current -> forward[0];
 if(current == NULL // current -> key != key)
 {
 int r level = random Level();
 if (r lvel > level)
 {
  for(int i = level+1 ; i < r level + 1 ; i++)
   update [i] = header;
  level = rlevel;
 }
Node * n = CreateNode (key, a level);
for (int i = 0 ; i <= a level ; i++)
 {
 n -> forward[i] = update [i] -> forward [i] ;
 update [i] -> forward [i] = n ;   }
 Cout << "Successfully inserted key "<< key << "\n";
}} ;

void deleteElement (int key) {
 Node * current = header ;
 Node * update (MAXLVL + 1);
 memset (update, 0, sizeof (Node *) * (MAXLVL + 1));
 for (int i = level ; i >= 0 ; i--) {
 while (current -> forward [i] != NULL &&
        current => forward [i] -> key < key )
     current = current -> forward [i];
     update [i] = current ; }
    current = current -> forward [0] ;
  if (current != NULL and current -> key == key )
  {
  for (int i = 0 ; i <= level ; i++) {
   if (update [i] -> forward [i] != current )
    break ;
    update [i] -> forward [i] = current -> forward [i];
  }
 while (level > 0 && header -> forward (level ) == 0)
  level -- ;
  cout << "Deleted " << key << "\n" } } ;
```

```
void search file (int key)
{
 Node * current = header ;
 for (int i = level ; i >= 0 ; i --)
 {
 while (current → forward [i] &&
          current → forward [i] → key < key)
        current = current → forward (i) ;
  }
  current = current → forward (0) ;
  if (current and current → key = = key)
  cout << "Found" << key << "\n" ;
 } ;
```