J. Anukesh
1BM18CS038

Hashing

```
class HashN
 {
    public:
        int k,v;
        HashN(int k,int v)
        {
          this->k=k;
          this->v=v;
        }
  };
 class   HashM
 {
    private:
        HashN **t ;
    public:
        HashM()
        {
          t= new  HashN * [Tsize];
          for (int i=0;i<Tsize;i++)
          {
            t[i]=NULL;
          }
        }
        int HashF(int k)
        {
          return k%Tsize;
        }
        void Ins(int k,int v)
        {
          int h=HashF(k);
          while (t[h]!=NULL&&t[h]->k!=k)
          {
             h=HashF(h+1);
          }
          if (t[h]!=NULL)
             delete t[h];
          t[h]=new HashN(k,v);
        }
        int SearchK(int k)
        {
          int h=HashF(k);
          while (t[h]!=NULL&&t[h]->k!=k)
          {
             h=HashF(h+1);
          }
        if(t[h]==NULL)
            return -1;
        else
            return t[h]->v;

    }
    void Rem(int k)
     {
        int h=HashF(k);
        while(t[h]!=NULL)
        {
            if(t[h]->k==k)
                break;
            h=HashF(h+1);
        }
        if(t[h]=NULL)
        {
            cout<<"No element found at key"<<k<<endl;
            return;
        }
        else
        {
            delete t[h];
        }
        cout<<"Element deleted"<<endl;

};
```