Corrigé du devoir n°2

Exercice1:

```
int  taille_liste (liste tete)
{ int n=0 ;
  for ( ; tete !=NULL ; tete=tete->svt)
         n++;
return n;
}
```

```
int  recherche_dicho (liste tete, int val)
{ int bi=1, bs=taille_liste(tete), milieu ;
  liste p ;
  while(bi<=bs)
    { milieu=(bi+bs)/2; p=tete;
      for (int i=1 ; i<=milieu ;i++) p=p->svt ;
     if (p->element==val) return 1;
     else if (p->element>val) bs=milieu-1;
            else bi=milieu+1;
    }
return 0;
}
```

Exercice 2 :

```
void  constTab (liste tete, liste *T, int *n)
{ int i=0, *n=taille_liste(tete) ;
 T=(liste *)malloc( (*n) * sizeof(liste)) ;
for ( ; tete !NULL; tete=tete->svt)
   { T[i]=tete; i++; }
}
```

```
void  triTab( liste *T, int n)
{ int i, j ; liste x ;
 for (i=0; i<n-1; i++)
  for (j=i+1; j<n; j++)
    if (T[i] -> element > T[j] -> element)
     { x=T[i]; T[i]=T[j]; T[j]=x; }
}
```

```
void affiche (liste *T, int n)
{  for (int i=0 ; i<n ; i++)
       printf("%d", T[i]->element);
}

main()
{  liste tete=creer_listeLiFo();
   int n;
  liste *T ;
 constTab(tete, T, &n) ;
triTab(T, n) ;
affiche(T, n) ;
}
```