Universidade do Minho

Transcrição de aula

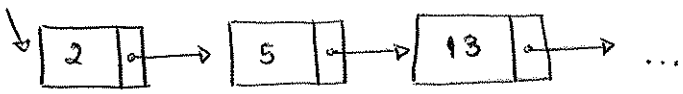
Disciplina	Programação Imperativa – 1º ano - LEI	
Secretário	Número:	Nome: Diana Lemos
Data: 2011-05-03		Nº Página 3
Turno: T2		Nº Alunos 61031

90 (alunos)

SUMÁRIO

Algoritmos c/ listas ligadas.

```
typedef struct slInt
{ int valor;
  struct slInt *seg;
} *Lista, NodoLista;
```

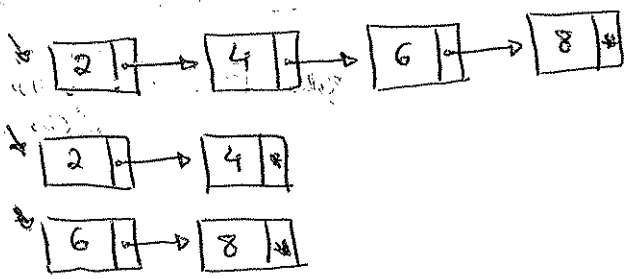



```
Lista SearchElem (Lista l, int m)
{ Lista aux;
  if (!l || (l->valor > m)
        aux = NULL;
  else
     if (l->valor == m)
           aux = l;
     else
           aux = SearchElem (l->seg, m);
return aux;
}
```

Transcrição (folha de continuação)

```
Lista Remove (Lista l, int m)
{
  Lista res;
  if (!l || (l->valor > m))
        res = l;
  else
        if (l->valor == m)
        {  res = l->seg;
           free(l);
         }
        else
          { l->seg = remove(l->seg, m);
            res = l;
          }
  return res;
}
```

FrontBacksplit




```
typedef struct sDuaslistas
{ Lista a,b;
} Duaslistas;
```

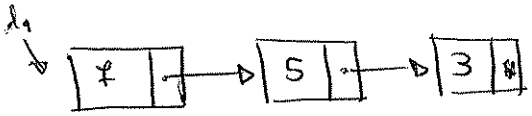
```
Duaslistas FrontBacksplit (Lista l)
{ int nelems = Count(l), i;
  Duaslistas res = {NULL, NULL};
for(i=1; i<= nelems/2 + nelems%2; i++)
  { res.a = Enqueue(res.a, l->valor);
    l->l->seg;
  }
for(i=1; i<= nelems/2; i++)
  { res.b = Enqueue(res.b, l->valor);
    l->l->seg;
  }
return res;
}
```

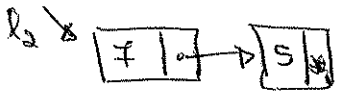
Transcrição (folha de continuação)

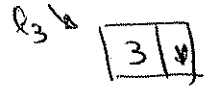
```
Void FrontBackSplit (Lista l, Lista *a, Lista *b)
{ int nelems = Count(l); i;
for (i=1; i<= nelems/2 + nelems%2; i++)
   { *a = Enqueue (*a, l->valor);
     l = l->seg;
   }
for(i=1; i< nelems/2; i++)
  { *b = Enqueue (*b, l->valor);
    l = l -> seg;
  }
}

int maim( )
{ Lista l1 = NULL, l2 = NULL, l3 = NULL;
 l1 = Push(Push(Push(l1, 3), 5), 7);
 FrontBackSplit (l1, &l2, &l3);
 Lista(l2);
 Lista(l3);
  ...
}
```

tpc:







RemoveDuplicados
AlternatingSplit
Merge
Reverse