

# Transcrição de aula

(30 alunos)

Disciplina	Programação Imperativa - 1º ano - LEI	
Secretário	Número: 61084	Nome: Helder Gonçalves
Data: 2011-04-26	TR2	Nº Página 4

## SUMÁRIO

Resoluções de exercícios c/ listas  
ligadas: sales de cinema e parques de  
estacionamento.

```
typedef char Filme [60];
typedef struct sBilhete
{
    int lugar;
    struct sBilhete * seg;
} * LBilhete, NBilhete;

typedef struct sSala
{
    int nlugares;
    LBilhete vendidos;
    Filme filme;
} Sala;

typedef struct sCinema
{
    Sala s;
    struct sCinema * seg;
} * Cinema, NCinema;
```

(

(



## Transcrição (folha de continuação)

```
int main ()
{
  sola s1 = { 150, NULL, "ET" },
    s2 = { 200, NULL, "Hannibal" };

  Cinema c1;

  c1 = inicializa (c1, s1);
  c1 = " " (c1, s2);
  ...
  listar (c1);
  ...
  if (disponivel (c1, "shark", 17))
    c1 = vende bilhete (c1, "shark", 17);
  ...
  listar disponibilidades (c1);

  void listar (Cinema c)
  {
    if (c)
      { printf ("%s", c -> s.filme);
        listar (c -> seg);
      }
  }
}
```

( )

( )

## Transcrição (folha de continuação)

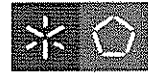
```
int disponivel (cinema c, Filme f, int l)
{
    if (c)
    {
        if (strcmp(f, c->s.filme) == 0)
        {
            return check(l, c->s.vendidos);
        }
        else
            return disponivel(c->seg, f, l);
    }
}

int check (int l, Bilhete B)
{
    if (B)
    {
        if (B->lugar == l)
            return 0;
        else
            return check(l, B->seg);
    }
    return 1;
}

Cinema vendeBilhete (cinema c, Filme f, int lg)
{
    if (!strcmp(f, c->s.filme))
    {
        c->s.vendidos = push(c->s.vendidos, lg);
        return c;
    }
    else
        return vendeBilhete(c->seg, f, lg);
}
```

( )

( )



## Transcrição (folha de continuação)

```
void listarDisponibilidades (cinema e)
{
    int res;
    if (e)
    {
        res = (e -> s. lugares - count (e -> s. vendidos));
        printf ("%i s - %i d / m", e -> s. filme, res);
        listarDisponibilidades (e -> seg);
    }
}
```

}

```
typedef struct sNotas
{
    float nota;
    struct sNotas * seg;
} * LNotas, modNotas;
```

```
typedef struct sAlunos
{
    char * numero;
    char * nome;
    struct sAlunos * seg;
} * LAlunos, modAlunos;
```

```
void listarAlunos (Lgroups lg)
```

```
{
    if (lg)
    {
        getreal (lg -> a);
        listarAlunos (lg -> seg);
    }
}
```

}

```
typedef struct sgroups
{
    LAlunos a;
    LNotas n;
    struct sgroups * seg;
} * Lgroups, modogroups;
```

```
void getreal (LAlunos la)
```

```
{
    if (la)
    {
        printf ("%i s \ t %i s", la -> nome, la -> numero);
        getreal (la -> seg);
    }
}
```

}

}