Universidade do Minho

Transcrição de aula

(30 alunos)

Disciplina	Programação Imperativa – 1º ano - LEI	
Secretário	Número: 61084	Nome: Helder Gonçalves
Data: 2011-04-26 TP2		Nº Página 4

SUMÁRIO

Resolução de exercícios c/ listas ligadas: Salas de cinema e parques de estacionamento.

```
typedef char Filme[60];
typedef struct sBilhete
{ int lugar;
  struct sLBilhete * seg;
} *LBilhete, NLBilhete;
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()
{ sala 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 = vendebilhete (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, LBilhete 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 c)
{ int res;
  if (c)
  { res = (c->S.nlugares - count(c->S.Vendidos));
    printf("%S - %d\n", c->S.filme, res);
    listardisponibilidades (c->seg);
  }
}
```

```
typedef struct sLNotas
{ float nota;
  struct sLNotas * seg;
} *LNotas, nodoNotas;

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

```
typedef struct sGroups
{ LAlunos a;
  LNotas n;
  struct sGroups * seg;
} * LGroups, nodoGroups
```

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

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