Exercicio 1 7 p

Imaxina que para subir uha escaleira podes subir peldaño a peldaño ou subir dous peldaños a un tempo. Coa lista `[1, 2, 1, 1, 2, 2]` denotamos unha posible maneira de subir unha escaleira de 9 peldaños: primerio subimos 1 peldaño, despóis 2 peldaños xuntos, logo dous peldaños de 1 en 1 finalmente catro peldaños de 2 en 2.

2p **a)** Representa con MAPLE a colección das listas correspondentes a todas as formas posibles de subir unha escaleira de 9 peldaños se subes tres de 1 en 1 e seis de 2 en 2.

Serache de axuda o paquete **`combinat`**, do que lembrarás que usamos comandos como **`permute`**, **`choose`**, **`numperm`**, e **`numcomb`**. Tamén che pode ser de axuda usar **`$(x, n)`**, que replica **`x`** *n* veces.

```
> with(combinat):
> permute([1,2,1,1,2,2]);
```

$$[[1,2,1,1,2,2],[1,2,1,2,1,2],[1,2,1,2,2,1],[1,2,2,1,1,2],[1,2,2,1,2,1],[1,2,2,2,1,1],[1,1,2,1,2,2],[1,1,2,2,1,2],[1,1,2,2,2,1],[1,1,1,2,2,2],[2,1,1,1,2,2],[2,1,1,2,1,2],[2,1,1,2,2,1],[2,1,2,1,1,2],[2,1,2,1,2,1],[2,1,2,2,1,1],[2,2,1,1,1,2],[2,2,1,1,2,1],[2,2,1,2,1,1],[2,2,2,1,1,1]] \quad \textbf{(1.1)}$$

2p **b)** Escribe un procedemento que tome como parámetro un número $n \geq 6$ e devolva todas as formas de subir unha escaleira de *n* peldaños, de un en un e dando tres saltos de dous peldaños.

```
> ForSub_3x2:=proc(n)
  permute([$(2,3),$(1,n-2*3)])
  end;
```

$$ForSub\_3x2 := \textbf{proc}(n)\ combinat:\text{-}permute([2\$3, 1\$n-6])\ \textbf{end proc} \quad \textbf{(1.2)}$$

```
> ForSub_3x2(9);
```

$$[[2,2,2,1,1,1],[2,2,1,2,1,1],[2,2,1,1,2,1],[2,2,1,1,1,2],[2,1,2,2,1,1],[2,1,2,1,2,1],[2,1,2,1,1,2],[2,1,1,2,2,1],[2,1,1,2,1,2],[2,1,1,1,2,2],[1,2,2,2,1,1],[1,2,2,1,2,1],[1,2,2,1,1,2],[1,2,1,2,2,1],[1,2,1,2,1,2],[1,2,1,1,2,2],[1,1,2,2,2,1],[1,1,2,2,1,2],[1,1,2,1,2,2],[1,1,1,2,2,2]] \quad \textbf{(1.3)}$$

3p **c)** Escribe un procedemento que toma como parámetro un número par *n* e devolve todas as formas posibles de subir unha escaleira de *n* peldaños, podendo subir peldaños de 1 en 1 e de 2 en 2.

Pódenche resultar de utilidade **`[…]`**, **`op(…)`**, **`for…from…to…by…do…end do`** ou **`if…then…elif…else…end if`**.

```
> ForSub:=proc(n)
  local L, i;
```

```
  L:=[];
     for i from 0 to n/2 do
        L:=[op(L), op(permute([$(2,i),$(1,n-2*i)]))]
     end do;
  end;
```

$$ForSub := \mathbf{proc}(n) \quad \textbf{(1.4)}$$

```
    local L, i;
    L := [ ];
    for i from 0 to 1/2*n do
        L := [op(L), op(combinat:-permute([2$i, 1$n - 2*i]))]
    end do
end proc
```

```
> ForSub(8);
```

$$[[1,1,1,1,1,1,1,1],[2,1,1,1,1,1,1],[1,2,1,1,1,1,1],[1,1,2,1,1,1,1],[1,1,1,2,1,1,1],[1,1,1,1,2,1,1],[1,1,1,1,1,2,1],[1,1,1,1,1,1,2],[2,2,1,1,1,1],[2,1,2,1,1,1],[2,1,1,2,1,1],[2,1,1,1,2,1],[2,1,1,1,1,2],[1,2,2,1,1,1],[1,2,1,2,1,1],[1,2,1,1,2,1],[1,2,1,1,1,2],[1,1,2,2,1,1],[1,1,2,1,2,1],[1,1,2,1,1,2],[1,1,1,2,2,1],[1,1,1,2,1,2],[1,1,1,1,2,2],[2,2,2,1,1],[2,2,1,2,1],[2,2,1,1,2],[2,1,2,2,1],[2,1,2,1,2],[2,1,1,2,2],[1,2,2,2,1],[1,2,2,1,2],[1,2,1,2,2],[1,1,2,2,2],[2,2,2,2]] \quad \textbf{(1.5)}$$

Exercicio 2 3 p

2p Escribe un **algoritmo recursivo** para contar de cantas formas distintas podes subir escaleiras de *n* peldaños, podendo subir peldaños de 1 en 1 e de 2 en 2.

```
> NumSub:=proc(n)
  if n=1 then 1
  elif n=2 then 2
  else NumSub(n-1) + NumSub(n-2)
  end if;
  end;
```

$$NumSub := \mathbf{proc}(n) \quad \textbf{(2.1)}$$

```
    if n = 1 then 1 elif n = 2 then 2 else NumSub(n - 1) + NumSub(n - 2) end if
end proc
```

```
> NumSub(8);
```

$$34 \quad \textbf{(2.2)}$$

1p Compara os valores dados por **NumSub** e os que poderías obter a partir de **ForSub** para *n* = 10, 20, 30 e 40.

Pódenche ser de utilidade **nops(…)**, **for…from…to…by…do…end do**, **if…then…elif…else…end if** ou **option remember**.

```
> nops(ForSub(10)) = NumSub(10);
```

$$89 = 89 \quad \textbf{(2.3)}$$

```
> nops(ForSub(20)) = NumSub(20);
```

(2.4)

$$10946 = 10946 \tag{2.4}$$

```
> NumSub:=proc(n) option remember;
  if n=1 then 1
  elif n=2 then 2
  else NumSub(n-1) + NumSub(n-2)
  end if;
  end;
```

$$NumSub := \mathbf{proc}(n) \tag{2.5}$$

$$\quad \mathbf{option}\ remember;$$

$$\quad \mathbf{if}\ n = 1\ \mathbf{then}\ 1\ \mathbf{elif}\ n = 2\ \mathbf{then}\ 2\ \mathbf{else}\ NumSub(n-1) + NumSub(n-2)\ \mathbf{end\ if}$$

$$\mathbf{end\ proc}$$

```
> nops(ForSub(30)) = NumSub(30);
```

$$1346269 = 1346269 \tag{2.6}$$

```
> nops(ForSub(20)) = NumSub(20)
```