AYED II
Lautaro Gastón Peralta

## *Práctico 3.3*

1. Modifique el código del algoritmo que resuelve el problema de la moneda utilizando backtracking, de manera que devuelva qué monedas se utilizan, en vez de sólo la cantidad.

```
fun cambio(d:array[1..n] of nat, i,j: nat) ret r: Set of nat
    var cambio_with_k : set of nat
    var cambio_without_k : set of nat
    if j=0 then r:= emptyset()
    else if i = 0 then r:= add(∞,emptyset())
    else if d[i] > j then r:= cambio(d,i-1,j)
    else
        cambio_without_k := cambio(d,i-1,j)
        cambio_with_k := add(i,cambio(d,i,j-d[i]))
        if(set_length(cambio_with_k) < set_length(cambio_without_k)) then
            r := cambio_with_k
        else
            r:= cambio_without_k
    fi
end fun
```

2. En un extraño país las denominaciones de la moneda son 15, 23 y 29, un turista quiere comprar un recuerdo pero también quiere conservar el mayor número de monedas posibles. Los recuerdos cuestan 68, 74, 75, 83, 88 y 89. Asumiendo que tiene suficientes monedas para comprar cualquiera de ellos, ¿cuál de ellos elegirá? ¿qué monedas utilizará para pagarlo? Justificar claramente y mencionar el método utilizado.

= 29-29 (58 no hay valor para comprar) - 29 (87 no hay valor para comprar)
**29-23 (52 puedo seguir pagando) -23 (75 puedo pagar el objeto) ***
29-23 (52 puedo seguir pagando) -15 (67 no hay valor para comprar)
**29-15-15 (59 puedo seguir pagando) - 15 (74 puedo pagar el objeto) -15 (89 puedo pagar el objeto)***
**23-15-15-15 (68 puedo pagar el objeto) - 15 (83 puedo pagar el objeto)**
**le conviene comprar el recuerdo de 75 (usa las monedas: 29,23,23)**

---

Para cada uno de los siguientes ejercicios:

- Identifique qué parámetros debe tomar la función recursiva que resuelve el problema.
- Describa con palabras **qué calcula** la misma, en función de sus argumentos.
- Defina la función recursiva en notación matemática y opcionalmente en código.
- Indique cuál es la llamada principal que obtiene el resultado pedido en el ejercicio.

---

3. Una panadería recibe $n$ pedidos por importes $m_1, \ldots, m_n$, pero sólo queda en depósito una cantidad $H$ de harina en buen estado. Sabiendo que los pedidos requieren una cantidad $h_1, \ldots, h_n$ de harina (respectivamente), determinar el máximo importe que es posible obtener con la harina disponible.

***Datos:**

- **n pedidos**
- **m1…mn importes (1<=i<=n)**
- **H => harina en buen estado**
- **los pedidos usan una cantidad de h1…hn de harina**
- **determinar maximo importe que es posible obtener con la harina disponible**

***Funcion:**

- **max_importe(H,m) = calcula la mayor cantidad de importes que es posible obtener con la harina disponible en deposito {h1,...hn}. Restando cuando sea posible, H-hi.**

***Definición de función recursiva:**

max_importe(H, i) = {

| | |
|---|---|
| 0 | i = 0 |
| ∞ | H = 0 ^ i > 0 |
| max_importe(H,m-1) | hi > H > 0 ^ i > 0 |
| max (max_importe(H-hi,m-1),max_importe (H,m-1)) | H > hi > 0 ^ i > 0 |

}

4. Usted se encuentra en un globo aerostático sobrevolando el océano cuando descubre que empieza a perder altura porque la lona está levemente dañada. Tiene consigo $n$ objetos cuyos pesos $p_1, \ldots, p_n$ y valores $v_1, \ldots, v_n$ conoce. Si se desprende de al menos $P$ kilogramos logrará recuperar altura y llegar a tierra firme, y afortunadamente la suma de los pesos de los objetos supera holgadamente $P$. ¿Cuál es el menor valor total de los objetos que necesita arrojar para llegar sano y salvo a la costa?

***Datos:**

- **Estamos en un globo y perdemos altura.**
- **n objetos (1<=i<=n).**
- **cada objeto tiene pesos p1…pn y valores v1..vn.**
- **P kg para recuperar altura y llegar a tierra firme.**
- **la suma de p1..pn >= P.**
- **Buscar el menor valor total de los objetos que necesita arrojar.**

***Funcion:**

- **min_obj(P, i) = "Es una funcion que calcula el menor valor total de objetos a tirar entre 1 a i, de manera tal que la suma sea mayor o igual a P."**

***Definicion de funcion recursiva:**

min_obj (P,i) = {

| | |
|---|---|
| 0 | P = 0 |
| ∞ | P > 0 ^ i = 0 |
| min_obj(P,i-1) | pi > P > 0 ^ i > 0 |
| min (vi + min_obj (P-pi, i-1) , min_obj(P, i-1) | P > pi > 0 ^ i > 0 |

}

5. Sus amigos quedaron encantados con el teléfono satelital, para las próximas vacaciones ofrecen pagarle un alquiler por él. Además del día de partida y de regreso ($p_i$ y $r_i$) cada amigo ofrece un monto $m_i$ por día. Determinar el máximo valor alcanzable alquilando el teléfono.

***Datos:**

- **Tenemos n amigos (1<=i<=n).**
- **Prestamos un celular, conocemos dia de partida y regreso de cada uno p_i y r_i.**
- **Ahora nos quieren pagar un monto mi cada amigo.**

***Funcion: max_monto(i,) = "Funcion que calcula el maximo monto obtenido de prestar el celular a mis amigos segun los dias de partida y regreso).**
**d es un arreglo de amigos en donde se conoce sus p_i, r_i y m_i.**

***Definicion de funcion recursiva:**

max_monto(d) = {
0 -> p_i < d
max (m_i * (r_i - p_i + 1) + max_monto(r_i + 1), max_monto(d+1))
-> p > 0 **v** r_i > 0
}

6. Un artesano utiliza materia prima de dos tipos: $A$ y $B$. Dispone de una cantidad $MA$ y $MB$ de cada una de ellas. Tiene a su vez pedidos de fabricar $n$ productos $p_1, \ldots, p_n$ (uno de cada uno). Cada uno de ellos tiene un valor de venta $v_1, \ldots, v_n$ y requiere para su elaboración cantidades $a_1, \ldots, a_n$ de materia prima de tipo $A$ y $b_1, \ldots, b_n$ de materia prima de tipo $B$. ¿Cuál es el mayor valor alcanzable con las cantidades de materia prima disponible?

***DATOS :**

- **Materias primas : A y B.**
- **Cada materia prima tiene cantidades MA y MB.**
- **pedidos a fabricar => n productos (p1…pn) (UNO DE CADA UNO).**
- **cada producto tiene un valor de venta (v1…vn).**
- **requieren cada uno para su elaboración cantidades (a1…an) de materia prima A y (b1…bn) de materia prima B.**
- **Buscar el mayor valor alcanzable con la cantidad de materia prima disponible.**

***Funcion: max_art(i,MA,MB) = "Es una funcion que calculara el maximo valor de venta alcanzable de p1…pn productos en base al uso de materia prima de estos."**

***Definicion de funcion recursiva:**

max_art(i,MA,MB) = {
0 -> MA=0 ^ MB = 0 ^ i = 0
max_art(i-1,MA,MB) -> a_i > MA **^** b_i > MB ^ i > 0
max (v_i + max_art(i-1,MA-a_i,MB-b_i), max_art(i-1,MA,MB))
-> a_i<=MA ^ b_i<=MB ^ i >0
}

7. En el problema de la mochila se buscaba el máximo valor alcanzable al seleccionar entre $n$ objetos de valores $v_1, \ldots, v_n$ y pesos $w_1, \ldots, w_n$, respectivamente, una combinación de ellos que quepa en una mochila de capacidad W. Si se tienen dos mochilas con capacidades $W_1$ y $W_2$, ¿cuál es el valor máximo alcanzable al seleccionar objetos para cargar en ambas mochilas?

***Datos:**

- **n objetos.**
- **cada objeto tiene valores v1..vn y pesos w1..wn.**
- **la mochila tiene capacidad W.**
- **DOS MOCHILAS W1 Y W2.**

***Función: mochila(i,W1,W2) : "Funcion que calcula el máximo alcanzable de dos mochilas, con objetos 1…n, tal que no sobrepasen sus pesos."**

***Definición recursiva de la función:**

mochila(i,W1,W2) = {

0 -> i=0 ^ W1=0 ^ W2=0

mochila(i-1,W1,W2) -> i > 0 ^ wi>W1 ^ wi>W2

max(mochila(i-1,W1,W2),vi + mochila(i-1,W1-wi,W2))
-> i>0 ^ wi<=W1 ^ wi>W2

max(mochila(i-1,W1,W2),vi + mochila(i-1,W1,W2-wi))
-> i>0 ^ wi>W1 ^ w<=W2

max(mochila(i-1,W1,W2),vi + mochila(i-1,W1-wi,W2), vi + mochila(i-1,W1i,W2-wi))
-> i>0 ^ wi<=W1 ^ w<=W2

}