1.- Un inversor dispone de un capital de C euros (C≥0) que desea invertir durante un año en un conjunto de n fondos de inversión (n≥1), sabiendo que no hay límite mínimo de inversión en cada fondo, que solo puede distribuirlo en a lo sumo n/2 fondos y que no puede invertir más de C/2 euros en cada uno de ellos. Para cada uno de estos fondos conocemos la rentabilidad anual que ofrece (en %) en función del número de euros invertidos en él. En la tabla R de dimensión (C/2+1,n) se recogen esos datos, siendo Rij el rendimiento anual (en porcentaje) obtenido al invertir i€ en el fondo j (0≤i≤C/2, 1≤j≤n) .

	Fondo 1	**Fondo 2**	**Fondo 3**	**…..**	**Fondo n**
0 €	**0 %**	**0 %**	**0 %**	**…..**	**0 %**
1 €	**$R_{1,1}$ %**	**$R_{1,2}$ %**	**$R_{1,1}$ %**	**…..**	**$R_{1,1}$ %**
2 €	**$R_{2,1}$ %**	**$R_{2,2}$ %**	**$R_{2,1}$ %**	**…..**	**$R_{2,1}$ %**
…..	**…..**	**…..**	**…..**	**…..**	**…..**
C/2 €	**$R_{C/2,1}$ %**	**$R_{C/2,2}$ %**	**$R_{C/2,1}$ %**	**…..**	**$R_{C/2,1}$ %**

El objetivo es obtener una distribución del capital C entre los n fondos de inversión de forma que se optimicen los beneficios, es decir que el capital recuperado al cabo del año sea máximo. Utilizando la metodología de Backtracking, diseñar un algoritmo que resuelva este problema.
Deberá responderse a las siguientes cuestiones:

a) Restricciones explícitas **(10%)**
b) Restricciones implícitas **(15%)**
c) Función a optimizar **(15%)**
d) Identificar y escribir el esquema de Backtracking a utilizar **(20%)**
e) Identificar y escribir cada una de las operaciones del esquema en relación con los datos del problema **(40%)**

Solución

Tupla de tamaño fijo

- Secuencia de decisiones:

Tomamos n decisiones, tantas como fondos de inversión.
La decisión d_i (i:1..n) corresponde a un valor comprendido entre 0 y C/2. Si d_i=0, se entiende que no se invierte nada en el fondo i-ésimo.

- Restricciones explícitas: $\forall i: d_i \in \{0,1,2,\dots,C/2\}\ \text{con}\ (1 \le i \le n)$

Cada nivel del grafo corresponde a un fondo de inversión. Es decir, en el nivel i decidimos la cuantía a invertir en el fondo i.

- Restricciones implícitas:

$$(N(i): (d_i > 0)\ \ (1 \le i \le n)) \le n/2$$

$$\sum_{i=1}^{n} d_i \le C$$

- Función a optimizar:

$$maximizar \quad \sum_{i=1}^{n} \left(d_i * \left(1 + R_{d_i,i}/100\right)\right)$$

- Esquema de Backtracking a utilizar:

```
Procedimiento Backtracking_OPTIMA (D:datos_problema; k:entero; e/s x, x_mejor:tupla; e/s v_mejor:valor)
preparar_recorrido_nivel_k;
mientras ∃_hermano_nivel_k hacer
        siguiente_hermano_nivel_k;
        opción
            solución(D,x,k) ∧ correcto(D,x,k): si valor(D,x,k) > v_mejor entonces
                                                    x_mejor=x;
                                                    v_mejor=valor(D,x,k);
                                               fsi
        ¬solución(D,x,k) ∧ correcto(D,x,k): Backtracking_OPTIMA (D, k+1, x, x_mejor, v_mejor);
        fopción
fmientras
fprocedimiento
```

- Código correspondiente a cada una de las operaciones que aparecen en el esquema:

```
Preparar_recorrido_nivel_k :    x[k]=-1;
∃_hermano_nivel_k :   x[k]<C/2
siguiente_hermano_nivel_k :  x[k]++;
solución(D,x,k) :   k==(n-1)
```

Función Correcto :

```
int correcto(int *x,int k,float **R, int C, int n) {
int i,correcto=1, total=0,contador=0;
for (i=0;i<=k;i++) {
  total+=x[i];
  if (x[i]>0) contador++;
}
if ((contador>n/2)||(total>C)) correcto=0;
return correcto;
}
```

Función Valor :

```
float valor(int *x,int k,float **R) {
int i;
float total=0;
for (i=0;i<=k;i++)
    total+=x[i]*(1+R[x[i]][i]/100);
return total;
}
```

Función imprimir_resultado :

```
void imprimir_tupla (int *x, int n) {
int i;
printf("\n<");
for (i=0;i<n-1;i++){
  printf("%d,",x[i]);
  }
printf("%d",x[n-1]);
printf(">");
}
```