Problema de la Mochila	
***Descripción**: Encontrar s tal que $p \leq c_0$, para cada objeto od_j en om su número de unidades es menor o igual que m_j y v tenga el mayor valor posible*	
***Técnica:** Programación Dinámica Con Filtro*	
Tipos • S - *SoluciónMochila* • A - *Integer*	
Propiedades Compartidas	*od, List <OM >, ordenada por vu de mayor a menor* *n, Número de elementos, derivada* *c_0, Capacidad inicial*
Propiedades Individuales	*c, entero no negativo* *j, entero en [0,N)* *va, entero, derivada*
Solución: Se tipo SolucionMochila	
***Tamaño**: n-j*	
***Solucion Parcial**: (a,v)* *Siendo v el valor de los objetos en la mochila*	
***Alternativas**: $A = [0,k], k = \min\left(\frac{c}{p_j}, m_j\right)$, ordenadas de mayor a menor* *La alternativa a representa el número de unidades del objeto j que introducimos en la mochila*	
Instanciación $Inicial = (c_0, 0{,}0)$	
Casos base $c = 0 \,\|\| j = n - 1$ ***Solución Caso Base*** $\begin{cases} (0{,}0), & c = 0 \\ (k, kv_j), & j = n - 1 \end{cases}$ $k = \min\left(\frac{c}{p_j}, m_j\right)$ ***Número de subproblemas:*** 1 ***Subproblema*** $p = (c, j, va) \xrightarrow{a} p_a = (c - ap_j, j + 1, va + av_j)$ **Esquema Recursivo** $sp(p) = \begin{cases} (0{,}0), & c = 0 \\ (k, kv_j), & j = N - 1 \\ \max\limits_{a \in A}\{(a, av_j + sp(p_a).v)\}, & en\ otro\ caso \end{cases}$	

$\boldsymbol{cS}(a,(a',v)) = (a, v + av_j)$)	
$\boldsymbol{sA}$: Elige la solución parcial con mayor valor de v	
***Objetivo**: v: Valor de la Mochila*	
***ObjetivoEstimado(a)** = va+ct(a)*	
Función de Cota $ct(a) = av_j + ct'(c - ap_j, i+1)$ $ct'(c,i) = \begin{cases} 0, & c = 0 \\ kv_i, & i = n-1 \\ kv_i + ct'(c - kp_i, i+1), & i < n \end{cases}$ $k = \min\left(\frac{(double)c}{p_j}, m_j\right)$	
Solución reconstruida $sr(a,v) = od_j \times a$*, es un caso base* $sr(a,s) = s + od_j \times a$*, es un caso recursivo*	
Complejidad	

Solucion Parcial

- *(a,v).a :* La alternativa de la solución parcial
- *(a,v).v :* La propiedad de la solución parcial

ObjetoMochila *(v,p,m,vu)*

- *Valor, Básica*
- *Peso, Básica*
- *Número Máximo de Unidades, Básica*
- *Valor Unitario, Derivada, v/p*

SolucionMochila *(om,v,p)*

- *ObjetosE nMochila: Multiset<ObjetoMochila>, básica*
- *Valor, Derivada, Valor Total de los objetos en la mochila*
- *Peso, Derivada, Peso Total de los objetos en la Mochila*

Notación Solución Mochila

- *$\emptyset$, Multiset Vacío*
- $od_j \times a$*, Multiset con a unidades del objeto en la posición j*
- $s + od_j \times a$*, Añadir a unidades a la solución s del objeto en la posición j*

ProblemaMochila *od, n, c_0, (c,j,va)*

- *Objetos disponibles*
- *Número de objetos disponibles*
- *Capacidad Inicial*
- *Capacidad, Básica*
- *Índice en objetos disponibles*
- *Valor Acumulado*

Notación

- od_j, *Objeto disponible en la posición j*
- v_j, *Valor del objeto en la posición j*
- p_j, *Peso del objeto en la posición j*
- m_j, *Número máximo de unidades del objeto en la posición j*

Operadores y Funciones

- *sA: Selecciona alternativa. Es una función de tipo Sp<A,T> sA(List<Sp<A,T>> sp)*
- *cS: Combina soluciones de subproblemas . Es una función de tipo Sp<A,T> cS(A a, Sp<A,T> sp)*
- *ct: Función de cota*
- *sr: Solución Reconstruida*