ISIS 1105 Diseño de Algoritmos
Semestre 2012-1
Prof. Rodrigo Cardoso
Tarea 2 Solución

Para entregar por Sicua+, antes de Marzo 28, 10:00
Grupos de hasta 2 personas, de una misma sección.

1 (25/100) Exponenciación para naturales

Considere el problema de desarrollar un algoritmo que calcule la exponenciación a^b, para a, b: **nat**, en tiempo logarítmico. En la solución solo pueden usarse operaciones aritméticas (suma resta, multiplicación, división entera, residuo módulo), aunque solo las multiplicaciones se consideran operaciones básicas.

```
[Ctx C: a,b: nat ∧ b≥0
{Q: true }
 …
{Inv P: y≥0 ∧ z*x^y = a^b }
{cota t: y }

 do … od

{R: z = a^b }
]
```

1a Explique qué técnica pudo haberse utilizado para proponer el invariante P.

Es un balance de información:
x^y representa lo que no se ha calculado.
z representa lo ya calculado.

[5/25]

1b Desarrolle un algoritmo que satisfaga la especificación indicada.

```
[Ctx C: a,b: nat ∧ b≥0

   {Q: true }

   x,y,z:= a,b,1;

   {Inv P: y≥0 ∧ z*x^y = a^b }
   {cota t: y }

   do y≠0    →     if    y mod 2 = 0 →     y,x:= y div 2,x*x
                   []    y mod 2 ≠ 0 →     y,z:= y-1,z*x
                   fi
   od

   {R: z = a^b }
]
```

[15/25]

1c Argumente por qué la complejidad temporal de su algoritmo es logarítmica en b, si se cuenta como operación básica el número de multiplicaciones.

En cada iteración se efectúa una multiplicación, ya sea `x:= x*x` o `z:= z*x`.

La cota `y` baja a `y/2` cada una o dos iteraciones, dependiendo de la paridad de `y`. El peor caso se da si la rebaja a `y/2` sucede exactamente cada dos iteraciones y esto pasa si $b=2^r-1$, para algún $r \geq 0$. En este caso se dan $\theta(2*r)$ multiplicaciones, i.e. , $\theta(\log b)$.

[5/25]

2 (25/100) Compresión de texto

Un texto está compuesto por caracteres alfabéticos y blancos. El texto mide `n`, $n \geq 0$ y está guardado en un arreglo de caracteres `b[0..n-1]`. Se quiere comprimir el texto eliminando los blancos entre palabras (una palabra es una secuencia no vacía de caracteres no blancos) y separando éstas con un carácter '/'. La variable `m` denota, al final, el tamaño del texto comprimido y el carácter final debe ser '/' seguido de blancos, a excepción del caso en que `m=n`. El siguiente ejemplo ilustra el comportamiento esperado:

Entrada:

```
      0                                   n
b:  Este   es el  mensaje  de ejemplo
```

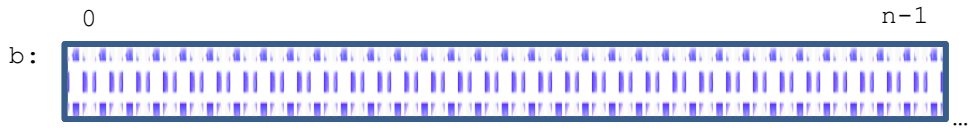
Salida:

```
      0                             m     n
b:  Este/es/el/mensaje/de/ejemplo/
```

2a Especifique el problema (Contexto, Pre-, Poscondición) como un desarrollo de un ciclo. Use gráficas para explicar los estados del proceso.

Ctx C: b:[0..n-1] **of char** ∧ n:**nat**

Pre Q:



Pos R:

[5/25]

2b Proponga un invariante `P` y una cota `t` para desarrollar el programa solución.

Inv P:

∧ sep ≡ true

cota t: n-q

[5/25]

2c Escriba código que satisfaga lo anotado en **1a** y **1b**.

```
[Ctx C
{Pre Q}
```

```
p,q,sep:= 0,0,true;

{Inv P}
{Cota t}
do q≠n →
      if    sep    →    if    b[q]=' '    → q:= q+1
                        []    b[q]≠' '    → b[p],b[q],sep:= b[q],' ',false;
                                            q:= q+1
                        fi
      []    ¬sep   →    if    b[q]=' '    → b[p]:= '/';
                                            p,q,sep:= p+1,q+1,true
                        []    b[q]≠' '    → b[p],b[q]:= b[q],' ';
                                            q:= q+1
                        fi
od
{Pos R }
]
```

[10/25]

2d Estime la complejidad temporal de su solución (operación básica: asignación).

La cota cuenta asignaciones de q ($\theta(1)$ cada iteración). En total: $\theta(n)$.

[3/25]

2e Estime la complejidad espacial de su solución.

Solo 3 variables: $\theta(1)$.

[2/25]

3 (25/100) Mínimo número de monedas

Sea $A = \{a_1,a_2,...,a_n\}$ un conjunto de diferentes denominaciones de monedas, donde cada $a_k \in \mathbf{nat}$, para $0<k\leq n$. Se supone, además, que $1=a_1<a_2<...<a_n$.

Dada una cantidad de dinero $C \in \mathbf{nat}$, se quiere encontrar el mínimo número de monedas de las denominaciones de A con las que se puede expresar la cantidad C.

Se quiere definir un algoritmo de programación dinámica para solucionar el problema. Para empezar, considere el siguiente lenguaje que establece notación para diseñar el algoritmo:

$mm(i,x)$: mínimo número de monedas de denominaciones $a_1,a_2,...,a_i$ para expresar la cantidad x, para $0\leq i\leq n$.

3a ¿Cómo se puede usar mm para expresar la pregunta del problema?

$mm(n,C) = ?$

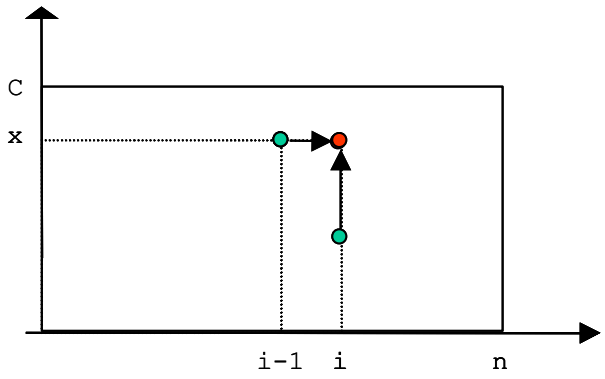
[2/25]

3b Defina una recurrencia para mm en un dominio adecuado para la solución del problema.

$$mm(i,x) = \begin{cases} x & \text{, si } 1=i \\ mm(i-1,x) & \text{, si } 1<i\leq n,\ a_i>x \\ \min\{mm(i-1,x),1+mm(i,x-a_i)\} & \text{, si } 1<i\leq n,\ a_i\leq x \end{cases}$$

[10/25]

3c Dibuje un diagrama de necesidades correspondiente a la definición de 2.



[8/25]

3d Defina estructuras de datos y plantee un invariante para un algoritmo que resuelva el problema según lo establecido en 3c.

Se usará un vector:

```
MM[0..C] of nat
```

Y el invariante

$P: 1 \le i \le n \wedge 0 \le x \le C \wedge (\forall k \mid 0 \le k < x : MM[k] = mm(i,k)) \wedge (\forall k \mid x \le k < C : MM[k] = mm(i-1,k))$

3e Estime el tiempo y el espacio que consume su algoritmo.

Como estructura de datos basta el vector MM, de tamaño $O(C)$.

$S(n) = O(C)$.

[2/20]

El algoritmo debe calcular una matriz de nC elementos.

Para calcular un elemento se requiere un tiempo $\theta(1)$.

Entonces:

$T(n) = \theta(nC)$

[3/25]

4 (25/100) Mínimo desperdicio copiando pistas de audio

Un disco duro tiene N pistas, $N>0$, cada una con duración en segundos $d_1, \ldots, d_N$. Un reproductor MP3 tiene una capacidad de almacenar hasta M segundos de grabación.

Se quiere encontrar un subconjunto de pistas del disco duro que pueda copiarse en el reproductor (sin repetir pistas) que minimice el tiempo de grabación que se desperdicie.

Desarrolle un algoritmo de programación dinámica que resuelva el problema y estime el orden de complejidad temporal y espacial de su algoritmo.

Sea:

`minD(i,x)` ≈ "mínimo desperdicio para grabar las pistas `1,2,…,i` si el reproductor MP3 tiene una capacidad de `x` segundos"

El valor del mínimo desperdicio se resuelve al calcular `minD(n,M)`.

[5/25]

Recurrencia para `minD`:

$$
\begin{aligned}
\text{minD}(i,x) &= x && , 1=i,\ x<d_1 \\
&= x-d_1 && , 1=i,\ d_1 \le x \\
&= \text{minD}(i-1,x) && , 1<i\le n,\ x<d_i \\
&= \min\{\text{minD}(i-1,x),\text{minD}(i-1,x-d_i)\} && , 1<i\le n,\ d_i \le x
\end{aligned}
$$

[10/25]

Para saber cuáles pistas grabar hay que recordar las decisiones en el cálculo del mínimo desperdicio. Sea:
`p(i,x)` ≈ "la pista `i` lleva para obtener el mínimo desperdicio mínimo desperdicio `minD(i,x)`"

Entonces:

$$
\begin{aligned}
\text{p}(i,x) &= \text{false} && , 1=i,\ x<d_1 \\
&= \text{true} && , 1=i,\ d_1 \le x \\
&= \text{false} && , 1<i\le n,\ x<d_i \\
&= \text{false} && , 1<i\le n,\ d_i \le x,\ \text{minD}(i-1,x) \le \text{minD}(i-1,x-d_i) \\
&= \text{true} && , 1<i\le n,\ d_i \le x,\ \text{minD}(i-1,x) > \text{minD}(i-1,x-d_i)
\end{aligned}
$$

[5/25]