Problema 2

Pregunta 1: ¿Cuál es el algoritmo más simple para construir la función Z de una cadena de caracteres?

Calculemos todos los prefijos de la cadena s, en el caso de bababa sus prefijos son

bababa
babab
baba
bab
ba
b

notemos que ningún prefijo comienza en a por lo tanto $\forall i$ tal que $s[i]=a$ $Z[i]=0$ luego para i tal que $s[i]=b$ e $i \neq 0$ tenemos la siguiente solución, llamemos P al arreglo de prefijos calculados entonces

$Z[i]$ = longitud mayor prefijo compartido entre s.substr(i, end) y $P[j]$ tal que longitud$(P[j]) \leq$ longitud(s.substr(i, end))

En el caso del ejemplo

$$Z[2] = \text{longitud del mayor prefijo común}\left(baba;\begin{bmatrix} baba \\ bab \\ ba \\ b \end{bmatrix}\right) = 4$$

El orden de complejidad del algoritmo es

Caso peor a lo más n comparaciones, luego m sufijos posibles de longitud k la cota del algoritmo es $O(n^3)$

Pregunta 2

r y l inicio y final de la subcadena s que termine lo más a la derecha posible y que sea prefijo de s.

Demostrar que para cualquier i

$$i \leq l \Rightarrow Z[i] \geq Z_{min}[i] \triangleq \min(Z[i-r], l-i+1)$$

Dem.

Consideremos primero el caso tal que $s[i]$ no es la primera letra de un prefijo entonces $Z[i]=0$ para ese caso $i=l$ entonces $Z[i]=0$ luego $Z_{min}[i] = \min([Z[i-r], l-i+1)$ $=0$

por lo tanto se cumple que $Z[i] \geq Z_{min}[i]$ en ese caso

Supongamos ahora que $s[i]$ es la primera letra de un prefijo entonces

$Z[i]$ = long del mayor prefijo común entre subs(i, end) y $P[j]$ con $P[\;]$ todos los prefijos con con $p[j]$.length $\leq$ subs(i,end).length.

luego $Z_{min}[i] = \min(Z[i-r], l-i+1)$, si $Z[i-r] = 0$ se cumple, si no $Z[i-r] \geq l-i+1$ ya que el prefijo común será mas grande, en todo caso se cumple que

$$Z[i] \geq \min(Z[i-r], l-i+1) = Z_{min}[i].$$

Pregunta 3

Notemos que si vamos a recorrer toda la cadena ya que si actualizamos l podriemos llegar hasta el final, ademas ya solo tendriamos que movernos desde el Z min

Pregunta 4

Primero notemos que Z y Z_{min} se actualizan a la par por lo que no es un ciclo anidado, luego acceder a $Z_{min}[i]$ es de orden 1. Buscar extender el matching también es de orden lineal por lo que la complejidad final es $O(n)$