DEPARTAMENTO
DE COMPUTACION
Facultad de Ciencias Exactas y Naturales - UBA

TP Final

Introducción a la Computación
Primer Cuatrimestre de 2016

Grupo 9

Integrante	LU	Correo electrónico
Berríos Verboven, Nicolás	46/12	nbverboven@gmail.com
Salustri, Guido	487/12	guido_salustri91@hotmail.com

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires
Ciudad Universitaria - (Pabellón I/Planta Baja)
Intendente Güiraldes 2610 - C1428EGA
Ciudad Autónoma de Buenos Aires - Rep. Argentina
Tel/Fax: (++54 +11) 4576-3300
http://www.exactas.uba.ar

Índice

1. Ejercicio 1

1.1. Decisiones de diseño

El programa se dividió en cuatro archivos separados.

Uno que vendría a ser el cuerpo del programa donde se resuelve el problema planteado utilizando las dos técnicas, *Fuerza Bruta* y *Divide&Conquer*, el cual recibe como argumento de entrada un archivo de texto (en el formato especificado en el problema). Otro donde se encuentran los distintos algoritmos de *sorting* necesarios para resolver el problema con la técnica de *Divide&Conquer*. Se trabajo generando listas de tuplas de tres elementos, dos puntos y su distancia respectiva, ya que la distancia es el parámetro a comparar y los puntos son los que se deben devolver.

En cuanto a los otros dos archivos, estos se encargan de las mediciones de eficiencia, uno genera un archivo en formato csv donde se encuentran los datos requeridos para graficar (el nombre del archivo es uno de los argumentos de entrada). El otro toma estos datos (el archivo debe llamarse datos1.csv) y genera el gráfico correspondiente.

1.2. Analisis del grafico

Al analizar el grafico, se observa que la técnica de *Fuerza Bruta* resulta mucho menos eficiente que las de *Divide&Conquer* con los distintos algoritmos de sorting, ya que su curva de complejidad ajusta a algo de tipo $\mathcal{O}(n^2)$ mientras que las de *Divide&Conquer* ajustan mejor a algo del tipo $\mathcal{O}(n\ log(n))$.

Sin embargo, algo que resulta extraño, es que dentro de las técnicas de *Divide&Conquer* los más eficientes resultaron ser *merge* y *bubble sort*, mientras que *quick sort* es el menos eficiente. Este hecho creemos que puede atribuirse a la forma en que esta implementado *quick sort* ya que esta hecho de de forma recursiva y al ir generando copias de cada paso recursivo es posible que esto pueda afectar su eficiencia.

2. Ejercicio 2

2.1. Especificaciones

`problema subListaAsc` $([a : \mathbb{Z}],\ n : \mathbb{Z}) = res : \mathbb{B}$ {

`requiere` $n > 0$;

`asegura` $res = (\exists i, j : \mathbb{Z})(0 \leq i \leq j < |a| \land subSeqCreciente(a, i, j) \land |i - j| + 1 \geq n)$;

`aux subSeqCreciente`[1] $([xs : \mathbb{Z}],\ i,\ j : \mathbb{Z}) : \mathbb{B} = (\forall k : \mathbb{Z})(i \leq k < j - 1 \rightarrow xs[k] < xs[k + 1])$;

}

`problema sufijoSubAscN` $([a : \mathbb{Z}],\ i,\ n : \mathbb{Z}) = res : \mathbb{B}$ {

`requiere` $n > 0$;

`requiere` $|a| > 0$;

`requiere` $0 \leq i < |a|$;

`asegura` $res = (\exists j : \mathbb{Z})(i \leq j < |a| \land subSeqCreciente(a, i, j)^1 \land |i - j| + 1 \geq n)$;

}

2.2. Implementación y transformación de estados

```
1  def subListaAsc(a, n):
2      # E_0 ≡ {a = A_0 ∧ n = N_0 ∧ N_0 > 0}
3      i = 0
4      # E_1 ≡ P_c ≡ {a = A_0 ∧ n = N_0 ∧ N_0 > 0 ∧ i = 0}
5      while i < len(a) and not sufijoSubAscN(a, i, n):
6          # E_c0
7          i += 1
8          # E_c1
9      # E_2 ≡ Q_c
10     return i < len(a)
11     # E_final ≡ {Q_c ∧ res = i < |a|}
```

Debe notarse que, para que esta transformación de estados sea válida, entre otras cosas debe demostrarse la corectitud del ciclo presente en el programa con repecto a su especificación. Esto permite garantizar que, si antes del ciclo vale P_c, entonces vale Q_c al finalizar el mismo. Para esto se cuenta con los teoremas de terminación y del invariante.

2.3. Especificación del ciclo

- $P_c \equiv \{a = A_0 \ \land \ n = N_0 \ \land N_0 > 0 \ \land \ i = 0\}$
- $Q_c \equiv \{a = A_0 \ \land \ n = N_0 \ \land N_0 > 0 \ \land$
  $(i = |a| \lor (0 \leq i < |a| \land (\exists j : \mathbb{Z})(i \leq j < |a| \land subSeqCreciente(a, i, j) \land |i - j| + 1 \geq n))) \land$
  $(\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \land \ subSeqCreciente(a, r, j) \ \land \ |r - j| + 1 \geq n))\}$
- $I \equiv \{a = A_0 \ \land \ n = N_0 \ \land N_0 > 0 \ \land \ 0 \leq i \leq |a| \ \land \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \land \ subSeqCreciente(a, r, j) \ \land \ |r - j| + 1 \geq n))\}$
- $B \equiv \{i < |a| \ \land \ \neg(\exists j : \mathbb{Z})(i \leq j < |a| \ \land \ subSeqCreciente(a, i, j) \ \land \ |i - j| + 1 \geq n))\}$
- $f_v = |a| - i$
- $cota = 0$

[1] La función auxiliar `subSeqCreciente` es la misma en ambos problemas.

2.4. Demostración de correctitud y terminación del ciclo

2.4.1. $\{P_c\} \rightarrow \{I\}$

Sabiendo que

$$P_c \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ i = 0\}$$

debemos implicar cada uno de los términos del invariante.

- $a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0$: Están directamente mencionados en P_c.
- $0 \leq i \leq |a|$: Como $i = 0$ entonces, trivialmente, vale $0 \leq i \leq |a|$. Si $|a| = 0$, sigue siendo válido debido al valor de i.
- $(\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n))$: Al reemplazar el valor de i en el antecedente se obtiene $0 \leq i < 0$. Este predicado es falso, ya que no hay elementos en ese rango. Entonces, la implicación es verdadera y también lo es el $\forall$.

Luego, $P_c \rightarrow I$.

2.4.2. $Vale\{I \wedge B\}\langle cuerpo\ del\ ciclo\rangle\ Vale\{I\}$

Para esta parte, debe realizarse la transformación de estados dentro del ciclo, pero previamente es necesario verificar una cosa. En la guarda del ciclo se realiza un llamado a la función `sufijoSubAscN`, cuya especificación se detalla en la sección 2.1 y, para que esto sea válido, debe cumplirse la precondición de dicho problema al llegar a esa parte del código en cada iteración.

Por P_c se sabe que $n = N_0$ y que $N_0 > 0$. Mediante un simple reemplazo podemos ver que $n > 0$, que es una de los requiere de `sufijoSubAscN`. Como el valor de n no se modifica, puede afirmarse que esta precondición se cumple en cada invocación.

Ahora bien, para llegar a la evaluación del término de la guarda en donde se llama a la función de interés, primero debe ser verdadero que $i < |a|$. En particular, esto debe valer la primera vez que se entre al ciclo. Además, se sabe que, por P_c, $i = 0$. Entonces puede deducirse que, la primera vez que se evalúa el segundo término de la guarda, $|a| > 0$. Por lo tanto, el llamado a `sufijoSubAscN` se realiza si la lista que se pasa como parámetro tiene al menos un elemento.

Para verificar la condición restante, puede utilizarse nuevamente el hecho de que en la primera iteración $i = 0$ y que la lista a posee una longitud mayor a 0 cuando se llama a `sufijoSubAscN`. Como ya se estableció anteriormente, el llamado a dicha función se realiza luego de que $i < |a|$ sea verdadero. Como i incrementa su valor en 1 con cada iteración, llegará un punto en que su valor sea exactamente $|a|$, la guarda evalúe a falso y se salga del ciclo (ver sección 2.4.4). En el paso anterior, $i = |a| - 1 < |a|$. Luego, puede afirmarse que `sufijoSubAscN` se invoca con un i que pertenece al rango $[0..|a| - 1]$, es decir $0 \leq i \leq |a| - 1$ o, de otra forma, $0 \leq i < |a|$.

Habiendo comprobado la validez del llamado a la función, se proseguirá con la transformación de estados del ciclo.

```
5       # E_1 ≡ P_c ≡ {a = A_0 ∧ n = N_0 ∧ N_0 > 0 ∧ i = 0}
6       while i < len(a) and not sufijoSubAscN(a, i, n):
7           # E_c0 ≡ {I ∧ B ∧ i = i_0}
8           i += 1
9           # E_c1 ≡ {E_c0 ∧ i = i_0 + 1}
```

$$E_{c0} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ 0 \leq i \leq |a| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n)) \ \wedge \ i < |a| \ \wedge \ \neg(\exists j : \mathbb{Z})(i \leq j < |a| \ \wedge \ subSeqCreciente(a, i, j)^{2} \ \wedge \ |i - j| + 1 \geq n) \ \wedge \ i = i_0\}$$

[2]Se refiere a la función auxiliar del mismo nombre utilizada en la especificación del problema. No se transcribe el contenido de la misma por razones de espacio.

$$E_{c0} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ 0 \leq i < |a| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg((\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n))) \ \wedge \ \neg(\exists j : \mathbb{Z})(i \leq j < |a| \ \wedge \ subSeqCreciente(a, i, j) \ \wedge \ |i - j| + 1 \geq n) \ \wedge \ i = i_0\}$$
(Juntando $0 \leq i \leq |a|$ y $i < |a|$)

$$E_{c0} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ 0 \leq i < |a| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r \leq i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n)) \ \wedge \ i = i_0\}$$
(Usando que, para i, vale que $\neg(\exists j : \mathbb{Z})(i \leq j < |a| \wedge subSeqCreciente(a, i, j) \wedge |i - j| + 1 \geq n)$)

$$E_{c0} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ 0 \leq i_0 < |a| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r \leq i_0 \rightarrow \neg((\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n)))\}$$
(Remplazando i por i_0)

$$E_{c0} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ 0 \leq i_0 < |a| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i_0 + 1 \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n))\}$$
(Usando que $r \leq i_0$ es equivalente a $r < i_0 + 1$)

$$E_{c1} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ 0 \leq i_0 < |a| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i_0 + 1 \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n)) \ \wedge \ i = i_0 + 1\}$$

$$E_{c1} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ 0 \leq i - 1 < |a| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n))\}$$
(Reemplazando por i en $0 \leq r < i_0 + 1$ y por $i + 1$ en $0 \leq i_0 < |a|$)

$$E_{c1} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ 1 \leq i < |a| + 1 \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n))\}$$
(Sumando 1 en cada término de $0 \leq i - 1 < |a|$)

$$E_{c1} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ 0 \leq i \leq |a| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n))\} \equiv I$$
(Usando que si $1 \leq i$ entonces $0 \leq i$ y que $i < |a| + 1$ es equivalente a decir que $i \leq |a|$)

Queda así demostrado que el cuerpo del ciclo preserva el invariante.

2.4.3. $\{B \wedge I \wedge f_v(v_1, ..., v_n) = F'\}\langle cuerpo\ del\ ciclo\rangle\{f_v(v_1, ..., v_n) < F'\}$

Evalúo la función variante al comienzo del ciclo (E_{c0}) y al final (E_{c1}):

- $f_v|_{E_{c0}} = |a| - i = |a| - i_0$ ya que, en E_{c0}, $i = i_0$.
- $f_v|_{E_{c1}} = |a| - i = |a| - (i_0 + 1)$ dado que $i = i_0 + 1$ al llegar a E_{c1}.

Como puede apreciarse, $f_v|_{E_{c1}} < f_v|_{E_{c0}}$, por lo que f_v es monótona decreciente.

2.4.4. $\{I \wedge f_v(v_1, ..., v_n) \leq cota\} \rightarrow \{\neg B\}$

En primer lugar, conociendo B, se busca llegar a una expresión para $\neg B$.

$$B \equiv \{i < |a| \ \wedge \ \neg(\exists j : \mathbb{Z})(i \leq j < |a| \ \wedge \ subSeqCreciente(a, i, j) \ \wedge \ |i - j| + 1 \geq n))\}$$
$$\neg B \equiv \{i \geq |a| \ \vee \ (0 \leq i < |a| \wedge (\exists j : \mathbb{Z})(i \leq j < |a| \wedge subSeqCreciente(a, i, j) \wedge |i - j| + 1 \geq n)))\}$$

Con la expresión de la función variante y sabiendo que la cota es 0, se tiene que $f_v \leq 0 \Leftrightarrow |a| - i \leq 0 \Leftrightarrow |a| \leq i$, que resulta ser el primer término de la negación de la guarda. Como esta última es una sucesión de predicados unidos mediante $\vee$, el hecho de que uno sea verdadero implica que $\neg B$ también lo es [3].

Luego, $\{I \wedge f_v(v_1, ..., v_n) \leq cota\} \rightarrow \{\neg B\}$.

[3] También puede pensarse como que $p \rightarrow (p \vee q)$ es una tautología.

2.4.5. $\{I \wedge \neg B\} \rightarrow \{Q_c\}$

Conociendo I y habiendo dado una expresión para $\neg B$, pueden combinarse para obtener

$$\{I \wedge \neg B\} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ 0 \leq i \leq |a| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n)) \ \wedge \ (i \geq |a| \ \vee \ (0 \leq i < |a| \wedge (\exists j : \mathbb{Z})(i \leq j < |a| \wedge subSeqCreciente(a, i, j) \wedge |i - j| + 1 \geq n)))\}$$

$$\{I \wedge \neg B\} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ ((0 \leq i \leq |a| \ \wedge \ i \geq |a|) \ \vee \ (0 \leq i \leq |a| \ \wedge \ 0 \leq i < |a| \wedge (\exists j : \mathbb{Z})(i \leq j < |a| \wedge subSeqCreciente(a, i, j) \wedge |i - j| + 1 \geq n))) \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n))\}$$

(`Distribuyendo` $0 \leq i \leq |a|$ `con el` $\vee$)

$$\{I \wedge \neg B\} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ (i = |a| \ \vee \ (0 \leq i < |a| \ \wedge \ (\exists j : \mathbb{Z})(i \leq j < |a| \ \wedge \ subSeqCreciente(a, i, j) \ \wedge \ |i - j| + 1 \geq n))) \ \wedge (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n))\}$$

(`Combinando` $0 \leq i \leq |a|$ `y` $i \geq |a|$`,` `y` $0 \leq i \leq |a|$ `y` $0 \leq i < |a|$)

Esto último es exactamente Q_c, por lo que puede afirmarse que $\{I \wedge \neg B\} \rightarrow \{Q_c\}$. Luego, el ciclo es correcto con respecto a la especificación propuesta y termina en un número finito de iteraciones.

2.5. Demostración de correctitud de `subListaAsc`

Al llegar a este punto ya fue demostrada la correctitud del ciclo presente en el código propuesto para resolver el problema. Con esto, puede asegurarse que, luego de la ejecución de dicho ciclo, vale Q_c. Lo que resta para poder afirmar que el programa es correcto con respecto a la especificación del problema es ver que el estado final implica la poscondición. Para esto, se parte de la base de que

$$E_{final} \equiv \{Q_c \ \wedge \ res = i < |a|\}.$$

Reemplazando Q_c se obtiene

$$E_{final} \equiv \{a = A_0 \ \wedge \ n = N_0 \ \wedge N_0 > 0 \ \wedge \ (i = |a| \ \vee \ (0 \leq i < |a| \ \wedge \ (\exists j : \mathbb{Z})(i \leq j < |a| \ \wedge \ subSeqCreciente(a, i, j) \ \wedge \ |i - j| + 1 \geq n))) \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |a| \ \wedge \ subSeqCreciente(a, r, j) \ \wedge \ |r - j| + 1 \geq n)) \ \wedge \ res = i < |a|\}.$$

Lo primero que puede hacerse es reemplazar a y n por su valor inicial, dado que no se modifican durante la ejecución del programa. Queda, entonces,

$$E_{final} \equiv \{N_0 > 0 \ \wedge \ (i = |A_0| \ \vee \ (0 \leq i < |A_0| \ \wedge \ (\exists j : \mathbb{Z})(i \leq j < |A_0| \ \wedge \ subSeqCreciente(A_0, i, j) \ \wedge \ |i - j| + 1 \geq N_0))) \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |A_0| \ \wedge \ subSeqCreciente(A_0, r, j) \ \wedge \ |r - j| + 1 \geq N_0)) \ \wedge \ res = i < |A_0|\}.$$

Además, puede distribuirse el término con el $\forall$ con el $\vee$ para obtener

$$E_{final} \equiv \{N_0 > 0 \ \wedge \ ((i = |A_0| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |A_0| \ \wedge \ subSeqCreciente(A_0, r, j) \ \wedge \ |r - j| + 1 \geq N_0))) \ \vee \ (0 \leq i < |A_0| \ \wedge \ (\exists j : \mathbb{Z})(i \leq j < |A_0| \ \wedge \ subSeqCreciente(A_0, i, j) \ \wedge \ |i - j| + 1 \geq N_0) \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |A_0| \ \wedge \ subSeqCreciente(A_0, r, j) \ \wedge \ |r - j| + 1 \geq N_0)))) \ \wedge \ |r - j| + 1 \geq N_0)) \ \wedge \ res = i < |A_0|\}.$$

Veamos cada término del $\vee$ por separado:

- $i = |A_0| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |A_0| \ \wedge \ subSeqCreciente(A_0, r, j) \ \wedge \ |r - j| + 1 \geq N_0))$:

  Si se reemplaza i en el rango del $\forall$ por el valor que posee, el predicado se transforma en

$$i = |A_0| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < |A_0| \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |A_0| \ \wedge \ subSeqCreciente(A_0, r, j) \ \wedge \ |r - j| + 1 \geq N_0)).$$

Agregando el hecho de que $res = i < |A_0|$ queda

$$i = |A_0| \ \wedge \ (\forall r : \mathbb{Z})(0 \leq r < |A_0| \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |A_0| \ \wedge \ subSeqCreciente(A_0, r, j) \ \wedge \ |r - j| + 1 \geq N_0)) \ \wedge \ res = i < |A_0|.$$

Pero $i = 0$, por lo que $res = False$. El resto del predicado puede reescribirse como

$$\neg(\exists i, j : \mathbb{Z})(0 \leq i \leq j < |A_0| \wedge subSeqCreciente(A_0, i, j) \wedge |i - j| + 1 \geq N_0),$$

así que, reemplazando en el predicado inicial, se obtiene

$$res = False \wedge \neg(\exists i, j : \mathbb{Z})(0 \leq i \leq j < |A_0| \wedge subSeqCreciente(A_0, i, j) \wedge |i - j| + 1 \geq N_0).$$

- $0 \leq i < |A_0| \wedge (\exists j : \mathbb{Z})(i \leq j < |A_0| \wedge subSeqCreciente(A_0, i, j) \wedge |i - j| + 1 \geq N_0) \wedge (\forall r : \mathbb{Z})(0 \leq r < i \rightarrow \neg(\exists j : \mathbb{Z})(r \leq j < |A_0| \wedge subSeqCreciente(A_0, r, j) \wedge |r - j| + 1 \geq N_0))$:

  Usando que $(p \wedge q) \rightarrow p$ es una tautología, puede reemplazarse el predicado por uno más débil, quedando

  $$0 \leq i < |A_0| \wedge (\exists j : \mathbb{Z})(i \leq j < |A_0| \wedge subSeqCreciente(A_0, i, j) \wedge |i - j| + 1 \geq N_0).$$

  Si se agrega $res = i < |A_0|$, y mirando el rango de i, puede concluirse que $res = True$. Además, puedo agregar el i al $\exists$ y, combinando todo, obtener

  $$res = True \wedge (\exists i, j : \mathbb{Z})(0 \leq i \leq j < |A_0| \wedge subSeqCreciente(A_0, i, j) \wedge |i - j| + 1 \geq N_0).$$

Juntando las dos ramas del $\vee$, puede llegarse a la conclusión de que

$$res = (\exists i, j : \mathbb{Z})(0 \leq i \leq j < |A_0| \wedge subSeqCreciente(A_0, i, j) \wedge |i - j| + 1 \geq N_0),$$

que es exactamente la poscondición del problema.

Luego, el programa es correcto con respecto a la especificación de `subListaAsc`.

3. Ejercicio 3

3.1. Decisiones de diseño

Se definieron 4 atributos para facilitar el uso de los métodos de la clase.

Al generar una instancia de `Rompecabezas`, se guarda en las variables `_ancho` y `_alto` la información referente a las dimensiones del objeto.

Cuando se llama al método `cargar` se recorre cada linea del archivo que recibe como parámetro, convirtiendo cada elemento en un número entero y guardándolo en `_rompecabezas`. En cuanto se encuentra al caracter `' '`, se escribe su posición en `_espacio_vacio` y, antes de agregarlo a `_rompecabezas`, se interpreta como el producto del ancho y el alto del rompecabezas. Esto responde al siguiente razonamiento:

Supongamos un rompecabezas de n filas y m columnas contenido en un archivo *.txt en el formato descripto en el enunciado. Si se piensa en su forma resuelta, esta puede representarse como

$$\begin{bmatrix} 1 & 2 & \cdots & m \\ m+1 & m+2 & \cdots & 2m \\ \vdots & \vdots & \ddots & \vdots \\ (n-1)*m+1 & (n-1)*m+2 & \cdots & n*m \end{bmatrix}$$

en dónde el caracter `' '` ocuparía el lugar de $n * m$.

Se definió el método privado `_unirListas`, que devuelve el resultado de concatenar los elementos de `_rompecabezas`.

3.2. Estrategia del backtracking

En primer lugar se definieron los casos base de la recursión. Si la llamada a `resuelto` devuelve `True`[4] y esto se logró en, a lo sumo, n pasos, entonces `resolver` devuelve `True`. En cambio, si no se pudo llegar a una solución en esa cantidad de movimientos, el resultado es `False`.

A continuación se describe la solución propuesta. Dada la posición del espacio vacío, se verifica que exista alguna posición adyacente que sea válida. De ser así, se realiza el movimiento en la dirección de dicha posición y se llama recursivamente a `resolver` con $n - 1$ pasos. Si el desplazamiento conduce a la resolución del rompecabezas, la función devuelve `True`, caso contrario mueve al espacio vacío en la dirección opuesta y continúa el resto de las posibilidades. Si luego de revisarlas todas no se llegó a una solución válida, el resultado es `False`.

El orden en que se chequean las direcciones se definió de forma arbitraria como: `UP`, `LEFT`, `DOWN`, `RIGHT`.

[4]Esto es si en `_rompecabezas` todos los valores están ordenados en forma creciente de izquierda a derecha y de arriba hacia abajo

4. Ejercicio 4

4.1. Descripción de los casos de test

Se definió un caso de prueba para el constructor de la clase de forma de verificar que arrojara una excepción del tipo `TypeError` cuando se intentaba instanciar un árbol inválido.

Se implementaron 2 casos de test para las funciones identificadas del 4 al 11 en el enunciado (3 en el caso de `find`) según la estrategia de *caja negra*:

- `izquierda`, `derecha`: Se corroboró que ocurriera una ecxepción del tipo `AttributeError` cuando alguna de las dos fuera vacía y que, para una dada instancia de `Arbol`, las dos funciones devolvieran el resultado correcto.
- `__eq__`: Se comprobó que, efectivamente, la igualdad estaba devolvía `True` cuando dos árboles eran iguales y `False` en el caso contrario.
- `find`: Se verificó que devolviera `False` si se buscaba un elemento que no se encontraba en el árbol vacío o si este último era vacío y que el resultado fuera `True` si dicho elemento estaba presente en el árbol.
- `espejo`: Se verificó que el espejo de un árbol vacío también lo fuera y que la función no modificara el objeto original.
- `preorder, posorder, inorder`: En los tres casos se testeó que si los métodos eran llamados por un árbol vacío, el resultado fuera la lista vacía y, además, que el resultado que devolvieran fuera correcto, de forma análoga a las pruebas realizadas para `izquierda` y `derecha`.

Para ejecutar los casos de test debe ingresarse el comando `python3 test.py` en una terminal posicionada adecuadamente.

5. Historial de commits

```
commit f3bcc3fb07f4f7d4f32028ec1aa0cafcc344ef98
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Sat Jun 18 13:00:35 2016 -0300

    Corregido typo -ej3

commit d501ac20f0719883753d9cf8ca7c46153a8797ab
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Sat Jun 18 12:57:22 2016 -0300

    Agregado resolver -ej3

commit c8eae17febb4b06c7c58ac26d7edec61bfd7e41c
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Fri Jun 17 14:26:27 2016 -0300

    Cambié la forma de verificar si el rompecabezas está resuelto.

commit 933d4aea6a6c7c1820a62732e4e3f0c6a52c7ca2
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Thu Jun 16 17:01:08 2016 -0300

    Agregué otra implementación de minimoposta y otro caso de test para el ej4

commit 86373b9f5343d86d83045cb9a7319ff7d733dc5c
Author: guidosalustri <gsalustri@ws6.labo4>
Date:   Thu Jun 16 16:52:08 2016 -0300

    datos para armar los graficos

commit 7d46c74a46943115cbda5ab373f19fc7272783bf
Author: guidosalustri <gsalustri@ws6.labo4>
Date:   Thu Jun 16 16:00:50 2016 -0300

    trando de graficar

commit 2806028c37fdd046d260d5d2649fdf667c5cdc69
Merge: 9b1544f ae91bd5
Author: guidosalustri <gsalustri@ws14.labo2>
Date:   Wed Jun 15 17:51:35 2016 -0300

    Merge branch 'master' of https://bitbucket.org/nbverboven/tp-final

commit 9b1544f97f11aa95da7180afe366c62d15bfc3e9
Author: guidosalustri <gsalustri@ws14.labo2>
Date:   Wed Jun 15 17:51:04 2016 -0300

    ej1 andando bien, falta retocarlo un poco

commit ae91bd51297098ddd10bdd7a0fedeabe88126fe6
Author: berrios nicolas <nberrios@ws7.labo1>
Date:   Wed Jun 15 16:15:53 2016 -0300
```

```
    Listo test.py

commit c87e44ab09f46c4c46e8c55a7b11056aee7ae138
Author: berrios nicolas <nberrios@ws6.labo4>
Date:   Wed Jun 15 14:05:27 2016 -0300

    Faltan test para rama derecha e izquierda

commit de02c81aee5976a26a7fc3c758a2111f9d202fb0
Author: berrios nicolas <nberrios@ws6.labo4>
Date:   Wed Jun 15 14:04:56 2016 -0300

    Corregido __eq__ en la clase Arbol

commit 2ac3eb2f55cd7300e72f35f8b4b9a452e93892a1
Author: berrios nicolas <nberrios@ws6.labo4>
Date:   Wed Jun 15 12:59:45 2016 -0300

    Cambié el método de fuerza bruta

commit fe912fb7e63d620600cadf42212bb7b3348598fc
Merge: 6d20d95 ab7bb33
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 18:14:24 2016 -0300

    asd

commit 6d20d95d24cdd0999689f2d2011eb3f1ba122fa4
Merge: b123221 5ca2cb8
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 18:10:32 2016 -0300

    Resueltos conflictos del merge

commit 5ca2cb8ba6c55cf46814933eb7584169248029f1
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 18:04:06 2016 -0300

    Cambié algo que o me acuerdo

commit 597381c71a3f9f4cf16006551d3ae2cd6b7dc185
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 18:03:12 2016 -0300

    Agregado el cálculo de distancias por D&C

commit d9eee29820000a7c2954a672a76cfb4125f261ca
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 18:02:22 2016 -0300

    asd
```

```
commit 7bef6e13ffa9967a6ee5d17747bab9dd76f75737
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 18:00:47 2016 -0300

    Corregido typo

commit ab7bb33a35f7740f3d57cdbe7d9dd5de0932195c
Author: guidosalustri <gsalustri@ws7.labo1>
Date:   Tue Jun 14 15:42:35 2016 -0300

    div&conquer

commit ef1b6b0c0524207c408127d4e3d9b6fddc1b58ea
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 15:28:13 2016 -0300

    asd

commit f2a9cc9e9dc9b7619edca7115a4d72ed46c78475
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 15:25:50 2016 -0300

    Actualizados archivos de ej1 con la rama master

commit b1232214d6b988e193035f1693059b76caf6b8e5
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 15:21:17 2016 -0300

    Creado ej1_Divide\&Conquer.py

commit aee03c51b7dd5f83d0f50e90deb8c17a0de5c2d3
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 15:19:47 2016 -0300

    Borré los algoritmos de sorting y renombré ej1.py como ej1_Fuerza_Bruta.py

commit f488c884230be8420dabc8734fd2c450c3490dfd
Merge: c7727a1 42f1387
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 14:23:54 2016 -0300

    Merge branch 'master' of https://bitbucket.org/nbverboven/tp-final

commit c7727a107d769314726e8b749d1ccea20cac3b82
Merge: a652a3f 28db1a0
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 14:22:09 2016 -0300

    Merge branch 'Nico'

commit 28db1a07fd2b9204b4d31bab5d7eb317811832be
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 14:16:24 2016 -0300
```

```
    Agregué varios casos de test para la clase Arbol(). Faltaría agregar el del

commit 5f5372e2a23d94dfb17ca5e68400a4b354a49569
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 14 14:15:03 2016 -0300

    Cambié la forma en que se levantan excepciones

commit 42f13876e4ea17cb5f733af72ba29b141c9fb364
Author: guidosalustri <gsalustri@ws23.labo7>
Date:   Tue Jun 14 12:47:28 2016 -0300

    ej 1 sin los algoritmos de sorting

commit ea8cba6888b8c88eb195d3c323c563f832fab659
Author: guidosalustri <gsalustri@ws23.labo7>
Date:   Tue Jun 14 12:46:07 2016 -0300

    algoritmos de sorting en un archivo separado

commit df48efef3707dd5512350e3831e146115fbaa08f
Author: guidosalustri <gsalustri@ws23.labo7>
Date:   Tue Jun 14 12:19:38 2016 -0300

    merge

commit cfbe841c7daf6025f4c3e1eaadfa5574d446f81d
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Mon Jun 13 01:50:30 2016 -0300

    Agregada implementación del ej2

commit a652a3f6d2fad645711253d87d9b7fbe4df8186c
Author: guidosalustri <gsalustri@ws6.labo4>
Date:   Fri Jun 10 13:47:59 2016 -0300

    megre sin terminar y test

commit bbe9b5386537d53bf237d6d584bfa1da8d0625e1
Author: guidosalustri <gsalustri@ws6.labo4>
Date:   Fri Jun 10 11:54:17 2016 -0300

    quicksort

commit c9cebd1203925b05a0662977d89db5c17cedb546
Author: guidosalustri <gsalustri@ws6.labo4>
Date:   Fri Jun 10 10:22:22 2016 -0300

    distancias de listas arreglado

commit 22af75e3323d39af03884df4b1488ae02ef1b87b
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
```

```
Date:   Tue Jun 7 15:47:32 2016 -0300

    Modificado find - ej4

commit 365d369cd80469a14cf309794e47e4b0b9f2e888
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 7 15:41:18 2016 -0300

    Agregado espejo - ej4

commit d6dee8695e77d261cb966be22dd5570819c867cf
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 7 15:33:20 2016 -0300

    Agregados posorder y preorder - ej4

commit 23740cbc04dc1a8f890a5bd4f1cb2a575741493c
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 7 15:28:40 2016 -0300

    Agregado inorder() - ej4

commit 2542991b14b9ecb0e58e155052adc1e12d9954c7
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 7 15:10:47 2016 -0300

    Agregados algunos métodos del ej4

commit 80d390c6632c005d0ce16f827853b8c7277f1bbd
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 7 13:45:03 2016 -0300

    Ya están casi todos los métodos de Rompecabezas. Hay que revisarlos y agrega

commit b4ec5b00e5ebc7faf12d528a17579d25469fb537
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Tue Jun 7 12:24:00 2016 -0300

    Ya anda Rompecabezas.resuelto()

commit 2a656c77e6ce3cbe798e84ce20f5d571297b44d8
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Sun Jun 5 00:13:31 2016 -0300

    Faltaba agregar el parámetro implícito en la llamada al atributo _rompecabez

commit e026c911c1f4d99d796919f116407b85335b7ced
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Sun Jun 5 00:06:18 2016 -0300

    Agregado método mover a la clase Rompecabezas. Falta probarlo.

commit 6423bc7d1781e750cb6ebd0b0c6f461969330cfc
```

```
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Sat Jun 4 23:41:41 2016 -0300

    Agregué un atributo para conocer la posición del espacio vacío en la clase R

commit a4f542e44ab78228271b8f5ce252fd4e50ba6ba8
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Sat Jun 4 23:20:29 2016 -0300

    No funciona el método resuelto en ej3

commit f66932720e99e903f1e1afa2d30ccae35f2ab21d
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Sat Jun 4 21:57:07 2016 -0300

    Arreglado cargar.

commit 3f8cbf2bfa3cf1eb31c9e5a1bc84d543c77ee72c
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Sat Jun 4 21:44:17 2016 -0300

    Al cargar el puzzle no está eliminando los tabs o el salto de línea. Arregla

commit 6122944ff6a0bee746454e95a5c6466769abed98
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Sat Jun 4 21:01:47 2016 -0300

    Agregué los métodos cargar y resuelto del ejercicio 3

commit 6b301dfed90df38242c3fe743db8a39b3b49fa92
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Fri Jun 3 18:21:54 2016 -0300

    La función listaDeDist se rompía cuando el archivo de entrada tenía un solo

commit eaf4e75a4b352d3f502238fe0e8f2cf71ceccb6d
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Fri Jun 3 17:57:12 2016 -0300

    Agregado bubbleSort

commit d29b87e1f9db1e6a360e4431f23547cbec7c60d3
Author: Nicolás Berríos Verboven <nbverboven@gmail.com>
Date:   Fri Jun 3 17:07:28 2016 -0300

    Cambié el nombre de algunas variables e implementé otra versión de maxPos.

commit d28d2b7fb6a23b1a08858dbd0dcccf135e0484b6
Author: guidosalustri <gsalustri@ws7.labo4>
Date:   Fri Jun 3 13:58:37 2016 -0300

    cambie parametro de entrada de listadepuntos
```

```
commit a2ac3b8a63cb8ab0a116f63646a4f3689f6dfd87
Author: guidosalustri <gsalustri@ws7.labo4>
Date:   Fri Jun 3 13:48:14 2016 -0300

    ejercicio uno hasta upsort

commit 210579017ae39389b0a76eb45b538eda616d2e39
Author: berrios nicolas <nberrios@ws6.labo4>
Date:   Fri Jun 3 11:45:01 2016 -0300

    Subidos archivos necesarios para los ejercicios 3 y 4
```