// H�fundur: Snorri Agnarsson, snorri@hi.is

// Noti� Link.java, sem er � Canvas, sem hj�lparklasa.

// � eftirfarandi umfj�llun eru allar ke�jur endanlegar

// og l�glegar eins og l�st er � Link.java

// Visti� �essa skr� undir nafninu E12.java og geri�

// vi�eigandi vi�b�tur �ar sem �i� finni� ???

public class E12

{

    // Notkun: removeMisssnLink(chain,res);

    // Fyrir:  chain er ekki-t�m ke�ja.

    //         res er tveggja staka Link<T>[], �.e. res.length == 2.

    // Eftir:  res[0] v�sar � �ann hlekk innan gamla chain sem

    //         inniheldur minnsta gildi�.

    //         res[1] v�sar � ke�ju hinna hlekkjanna sem voru �

    //         gamla chain, � einhverri �skilgreindri r��.

    //         Allir hlekkir � gamla chain eru anna� hvort � ke�junni

    //         res[1] e�a eru hlekkurinn sem res[0] v�sar �.

    //         Ekki �arf a� taka fram (� Java) a� �ll gildi (head) �

    //         hlekkjunum eru �breytt, a�eins halarnir (tail) hafa

    //         hugsanlega breyst.

    //         Ekki m� �thluta neinum n�jum hlekkjum.

    // Ath.:   B�a m� til fylki res me� eftirfarandi Java skipun:

    //           Link<T>[] res = (Link<T>[])new Link<?>[2];

    //         �i� f�i� �� a�v�run fr� Java, en �a� er � lagi.

    public static<T extends Comparable<? super T>>

    void removeMinLink( Link<T> chain, Link<T>[] res )

    {

        // H�r vantar forritstexta.

        // �tf�ri� �etta me� lykkju �ar sem fastayr�ingin skal

        // vera keiml�k �eirri fastayr�ingu sem notu� var �

        // lausninni � MinOfMultiset sem vi� leystum ��ur �

        // Dafny.  A�alatri�i� h�r er a� fastayr�ingin

        // lykkjunnar s� g��. Ekki f�st m�rg stig fyrir lausn

        // sem ekki hefur g��a fastayr�ingu jafnvel ��tt

        // falli� virki samkv�mt l�singu.

        Link<T> min = chain;

        Link<T> discarded = new Link<T>();

        Link<T> rest = chain.tail;

        while(rest != null && rest.head != null)

            //  allir hlekkir í gamla chain eru í rest , discarded eða

            //  hausnum á min,

            //  fyrir sérhvern hlekk í discarded er hausinn á min

            //  minni eða jafn honum.

        {

            if(min.head.compareTo(rest.head) > 0) {

                min.tail = discarded;

                min = rest;

                rest = rest.tail;

            }

            else {

                Link<T> temp = rest.tail;

                rest.tail = discarded;

                discarded = rest;

                rest = temp;

            }

        }

        res[0] = min;

        res[1] = discarded;

    }

    // Notkun: Link<T> y = selectionSort(x);

    // Fyrir:  x er l�gleg ke�ja �ar sem hlekkirnir innihalda

    //         l�gleg gildi af tagi T.

    // Eftir:  y er ke�ja s�mu hlekkja �annig a� hlekkirnir

    //         � y eru � vaxandi hausar�� mi�a� vi� compareTo

    //         fyrir hluti af tagi T.

    public static<T extends Comparable<? super T>>

    Link<T> selectionSort( Link<T> x )

    {

        // H�r vantar forritstexta.

        // �tf�ri� �etta me� lykkju �ar sem fastayr�ingin skal

        // vera keiml�k �eirri fastayr�ingu sem notu� var �

        // lausninni � SelectionSort sem vi� leystum ��ur �

        // Dafny.  A�alatri�i� h�r er a� fastayr�ingin

        // lykkjunnar s� g��. Ekki f�st m�rg stig fyrir lausn

        // sem ekki hefur g��a fastayr�ingu jafnvel ��tt

        // falli� virki samkv�mt l�singu.

        if( x == null || x.tail == null) {

            return x;

        }

        Link<T>[] res = (Link<T>[])new Link<?>[2];

        removeMinLink(x, res);

        Link<T> rest = res[1];

        Link<T> z = res[0];

        Link<T> w = z;

        while( rest != null && rest.head != null)

            //  allir hlekkir í gamla x eru í rest eða z

            //  fyrir sérhvern hlekk í rest er hausinn á z

            //  minni eða jafn honum.

        {

            removeMinLink(rest, res);

            rest = res[1];

            w.tail = res[0];

            w = w.tail;

        }

        w.tail = null;

        return z;

    }

    // Notkun: Link<T> z = insert(x,y);

    // Fyrir:  x er ke�ja � vaxandi r�� (m� vera t�m).

    //         y v�sar � hlekk (m� ekki vera null).

    // Eftir:  z er ke�ja � vaxandi r�� sem inniheldur

    //         alla hlekkina �r x auk hlekksins y.

    //         Athugi� a� ekki m� �thluta neinum n�jum

    //         hlekkjum.

    public static<T extends Comparable<? super T>>

    Link<T> insert( Link<T> x, Link<T> y )

    {

        // H�r vantar forritstexta.

        if(x.head == null || y.head.compareTo(x.head) <= 0 ) {

            y.tail = x;

            return y;

        }

        Link<T> z = x;

        while( z.tail.head != null && z.tail.head.compareTo(y.head) < 0)

            // z er keðja í vaxandi röð sem inniheldur alla hlekki

            // úr x og hlekkinn y.

        {

            z = z.tail;

        }

        y.tail = z.tail;

        z.tail = y;

        return x;

    }

    // Notkun: Link<T> y = insertionSort(x);

    // Fyrir:  x er l�gleg ke�ja �ar sem hlekkirnir innihalda

    //         l�gleg gildi af tagi T.

    // Eftir:  y er ke�ja s�mu hlekkja �annig a� hlekkirnir

    //         � y eru � vaxandi hausar�� mi�a� vi� compareTo

    //         fyrir hluti af tagi T.

    public static<T extends Comparable<? super T>>

    Link<T> insertionSort( Link<T> x )

    {

        // H�r vantar forritstexta.

        Link<T> z = new Link<T>();

        Link<T> rest = x;

        while(rest.tail != null && rest.tail.head != null)

            //   sérhver hlekkur í upprunalega x er annaðhvort

            //   í keðjunum rest eða z sem innihalda lögleg gildi

            //   að tagi T,

            //   z er í vaxandi röð

        {

            Link<T> temp = rest.tail;

            z = insert(z,rest);

            rest = temp;

        }

        return z;

    }

    // Notkun: Link<T> x = makeChain(a,i,j);

    // Fyrir:  a er T[], ekki null.

    //         0 <= i <= j <= a.length.

    // Eftir:  x v�sar � ke�ju n�rra hlekkja sem innihalda

    //         gildin a[i..j), � �eirri r��, sem hausa.

    public static<T> Link<T> makeChain( T[] a, int i, int j )

    {

        if( i==j ) return null;

        Link<T> x = new Link<T>();

        x.head = a[i];

        x.tail = makeChain(a,i+1,j);

        return x;

    }

    // Keyri� skipanirnar

    //   javac E12.java

    //   java E12 1 2 3 4 3 2 1 10 30 20

    // og s�ni� �tkomuna � athugasemd h�r:

        // Selection Sort: 1 1 2 2 3 3 4

        // Insertion Sort: 1 1 10 2 2 30 3 3 30 4 null

    public static void main( String[] args )

    {

        Link<String> x = makeChain(args,0,args.length);

        x = selectionSort(x);

        while( x != null )

        {

            System.out.print(x.head+" ");

            x = x.tail;

        }

        System.out.println();

        x = makeChain(args,0,args.length);

        x = insertionSort(x);

        while( x != null )

        {

            System.out.print(x.head+" ");

            x = x.tail;

        }

    }

}