// Tilvik af H11<T> eru forgangsbiðraðir hluta af tagi T.

// Tilvik af H11<T> er aðeins hægt að búa til fyrir klasa T

// sem eru Comparable, þ.e. tilvik af T eru samanburðarhæf hvort

// við annað, með þeim samningi sem því fylgir.

// Skilgreiningar (þetta má skilgreina formlega í Dafny):

//

//   Í fylki a sem inniheldur sæti i segjum við að sæti 2\*i+1 og

//   2\*i+2 séu börn sætis i, ef þau eru til staðar í fylkinu.

//   Á svipaðan hátt segjum við að fyrir sæti i > 0 sé sæti

//   floor((i-1)/2) foreldri sætis i.

//

//   Við skilgreinum að sæti 2\*i+1 og 2\*i+2 séu afkomendur sætis

//   i og við skilgreinum einnig að ef sæti k er afkomandi j og

//   j er afkomandi i þá er k afkomandi i (gegnvirkni, transitivity).

//

//   Fyrir svæði a[i..j) í fylki samanburðarhæfra gilda segjum við

//   að svæðið uppfylli hrúguskilyrði þá og því aðeins að fyrir

//   sérhver tvö sæti n og m innan svæðisins, þar sem m er afkomandi

//   n gildi að bæði a[m] og a[n] eru lögleg gildi (ekki null, til

//   dæmis) og a[m] <= a[n], þ.e. a[m].compareTo(a[n]) <= 0.

//

//   Við segjum einnig að svæði a[0..j) sé hrúga ef svæðið

//   uppfyllir hrúguskilyrði.

//

//   Setning: Ef a[i..j) er svæði í fylki og j < 2\*i+1 þá uppfyllir

//   svæðið hrúguskilyrði því ekkert sæti innan svæðisins er

//   afkomandi annars sætis innan svæðisins. Jafngilt skilyrði er

//   þegar i > floor((j-1)/2) eða þegar i >= floor((j-1)/2)+1.

//

//   Setning (sannanleg í Dafny): Ef a[0..j) er hrúga þá er a[0]

//   stærsta gildið í svæðinu (ef j != 0, að sjálfsögðu).

//

//   Takið eftir að hrúga a[0..j) er tvíundartré með j hnúta í

//   eins miklu jafnvægi og hægt er að öðlast með j hnúta tré.

public class H11< T extends Comparable<? super T>>

{

    private T[] a;

    private int n;

    // draugabreyta multiset<T> m;

    // Fastayrðing gagna:

    //    ...Skrifið stöðulýsingu hér sem lýsir því hvernig

    //    ...gildin í forgangsbiðröðinni, sem einnig eru gildin

    //    ...í draugabreytunni m, eru geymd fremst í fylkinu a,

    //    ...þannig að þau mynda hrúgu n gilda.  Munið að setja

    //    ...skilyrði á útkomur samanburða milli viðeigandi sæta

    //    ...í svæðinu og setjið skilyrði sem tengja saman n og

    //    ...a.length.  Reynið að sjá til þess að ekki þurfi mjög

    //    ...oft að endurúthluta a, en sjáið einnig til þess að

    //    ...minnissóun sé ekki óhófleg.

    //

    //    Munið að fastayrðingin er (í Java, ekki í Dafny) sjálfgefinn

    //    hluti af eftirskilyrði allra opinberra aðgerða, þar með

    //    talið allra smiða. Einnig er fastayrðingin sjálfgefinn

    //    hluti forskilyrðis allra opinberra boða annarra en

    //    smiða.

    // Notkun: H11<T> pq = new H11<T>();

    // Fyrir:  Ekkert (annað en að T verður að vera löglegt).

    // Eftir:  pq er ný tóm forgangsbiðröð gilda af tagi T

    //         með pláss fyrir ótakmarkaðan fjölda gilda.

    public H11()

    {

        a = (T[]) new Comparable<?>[100];

        n = 0;

    }

    // Notkun: rollDown(a,i,j);

    // Fyrir:  a[i..j) og a[i+1..j) eru svæði í a.

    //         a[i+1..j) uppfyllir hrúguskilyrði.

    // Eftir:  a[i..j) inniheldur sömu gildi og áður,

    //         en þeim hefur verið umraðað þannig að

    //         a[i..j) uppfyllir nú hrúguskilyrði.

    public static<E extends Comparable<? super E>> void rollDown( E[] a, int i, int j )

    {

        // Hér vantar forritstexta

        // Hér ætti að vera lykkja með fastayrðingu sem getur verið

        // eitthvað a þessa leið:

        int k = i;

        while(true)

            //   i <= k < j

            //   Allir samanburðir milli sæta p < q í svæðinu a[i..j)

            //   eru í samræmi við hrúguskilyrði nema e.t.v. í þeim

            //   tilvikum þegar i=j.

        {

            int c = 2\*k+1;

            if(c >= j) {

                return;

            }

            if(c+1 < j && (int)a[c+1] < (int)a[c]) {

                c = c+1;

            }

            E temp = a[k];

            a[k] = a[c];

            a[c] = temp;

            k = a[c];

        }

    }

    // Notkun: rollUp(a,i,j);

    // Fyrir:  a[i..j) og a[i..j+1) eru svæði í a.

    //         a[i..j) uppfyllir hrúguskilyrði.

    // Eftir:  a[i..j+1) inniheldur sömu gildi og áður,

    //         en þeim hefur verið umraðað þannig að

    //         a[i..j+1) uppfyllir nú hrúguskilyrði.

    public static<E extends Comparable<? super E>> void rollUp( E[] a, int i, int j )

    {

        // Hér vantar forritstexta.

        // Hér ætti að vera lykkja með fastayrðingu sem getur verið

        // eitthvað a þessa leið:

        int k = j;

        while(true)

            //   i <= k <= j

            //   Allir samanburðir milli sæta p < q í svæðinu a[i..j+1)

            //   eru í samræmi við hrúguskilyrði nema e.t.v. í þeim

            //   tilvikum þegar i==j.

        {

            if(k==0) {

                return;

            }

            int c = (k-1)/2;

            if((int)a[k] >= (int)a[c]) {

                return;

            }

            E temp = a[k];

            a[k] = a[c];

            a[c] = temp;

            k = c;

        }

    }

    // Notkun: sort(a);

    // Fyrir:  a er fylki gilda af tagi E (og E er löglegt).

    // Eftir:  a hefur verið raðað í vaxandi röð.

    public static<E extends Comparable<? super E>> void sort( E[] a )

    {

        // Hér vantar forritstexta.

        // Þetta skal útfæra á hraðvirkan hátt, þ.e. O(n log(n)),

        // annaðhvort með því að nota einungis rollDown í tveimur

        // lykkjum, eða með því að nota rollUp í einni lykkju og

        // rollDown í annarri lykkju.

    }

    // Notkun: int n = pq.count();

    // Fyrir:  pq er forgangsbiðröð.

    // Eftir:  n er fjöldi gilda í pq.

    public int count()

    {

        return this.n;

    }

    // Skrifið lýsingu hér

    // Notkun: ???

    // Fyrir:  ???

    // Eftir:  ???

    public T deleteMax()

    {

        // Hér vantar forritstexta.

        // Notið rollDown til að útfæra aðgerðina.

        // Munið að uppfæra einnig draugabreytuna m.

    }

    // Skrifið lýsingu hér

    // Notkun: ???

    // Fyrir:  ???

    // Eftir:  ???

    public void put( T x )

    {

        // Hér vantar forritstexta.

        // Notið rollUp til að útfæra aðgerðina.

        // Munið að uppfæra einnig draugabreytuna m.

        // Athugið að undir einhverjum kringumstæðum þurfið þið að

        // stækka fylkið a. Eðlilegt er þá að tvöfalda stærðina.

        // Notið viðeigandi fastayrðingu í lykkjunni þegar þið

        // afritið frá gamla fylkinu yfir í nýja.

    }

    // Prófið að keyra

    //   java H11 1 2 3 4 10 20 30 40

    // Það ætti að skrifa

    //   1 10 2 20 3 30 4 40

    //   40 4 30 3 20 2 10 1

    public static void main( String[] args )

    {

        sort(args);

        for( int i=0 ; i!=args.length ; i++ ) System.out.print(args[i]+" ");

        System.out.println();

        H11<String> pq = new H11<String>();

        for( int i=0 ; i!=args.length ; i++ ) pq.put(args[i]);

        while( pq.count() != 0 ) System.out.print(pq.deleteMax()+" ");

    }

}