#include<stdio.h>

int tpe = -1,tps=-1,stks[10],stke[10];

void pushs(int);

void pushe(int);

int pope();

int pops();

int tope();

int tops();

int emptye(int \*);

int emptys(int \*);

int main()

{

int arrs[4] = {1,2,5,6};

int arre[4] = {3,4,7,8};

int n = sizeof(arrs)/sizeof(arrs[0]);

mergeintervals(arrs,arre, n);

while(emptye(stke)&&emptys(stks))

{

printf("%d%d",pops(),pope());

}

return 0;

}

void mergeintervals(int arrs[],int arre[], int n)

{

insertionsort(arrs,arre,n);

int start,end,index;

pushs(arrs[0]);

pushe(arre[0]);

for (int i = 1 ; i < n; i++)

{

start = tops();

end=tope();

if (end < arrs[i])

{

pushs(arrs[i]);

pushe(arre[i]);

}

else if (end< arre[i])

{

end= arre[i];

pops();

pope();

pushs(start);

pushe(end);

}

}

}

void pushs(int value)

{

stks[++tps] = value;

}

void pushe(int value)

{

stke[++tpe] = value;

}

int tops()

{

return stks[tps];

}

int tope()

{

return stke[tpe];

}

int pope()

{

int a=stke[tpe];

tpe--;

return a;

}

int pops()

{

int b=stks[tps];

tps--;

return b;

}

int emptys(int stks[])

{

if(tps == -1)

return 0;

else

return 1;

}

int emptye(int stke[])

{

if(tpe == -1)

return 0;

else

return 1;

}

void insertionsort(int arrs[],int arre[], int n)

{

int i, key, j,k;

for (i = 1; i < n; i++)

{

key = arrs[i];

j = i-1;

k=arre[i];

while (j >= 0 && arrs[j]> key)

{

arrs[j+1] = arrs[j];

arre[j+1] = arre[j];

j = j-1;

}

arrs[j+1] = key;

arre[j+1]=k;

}

for (int i = 0; i <n; i++)

printf("%d",arrs[i]);

for (int i = 0; i <n; i++)

printf("%d",arre[i]);

}