```
import java.util.*;
import java.io.*;
// import java.lang.*;
// import java.math.*;

public class Codeforces {
    static FastReader sc=new FastReader();
        static PrintWriter out=new PrintWriter(System.out);
        static long mod=1000000007;
        static long mod1=998244353;
        static int MAX=Integer.MAX_VALUE;
        static int MIN=Integer.MIN_VALUE;
        static long MAXL=Long.MAX_VALUE;
        static long MINL=Long.MIN_VALUE;
        public static void main (String[] args) throws java.lang.Exception
        {
                // your code goes here
                int t=I();
                while(t-->0)
                {
                        int W=I(),H=I();
                        int x1=I(),y1=I(),x2=I(),y2=I();
                        int w=I(),h=I();
                        int ans=MAX;
                        if(x2-x1+w<=W){
                                ans=Math.min(Math.max(0,w-x1),Math.max(0,w-(W-x2)));
                        }
                        if(y2-y1+h<=H){
                                ans=Math.min(ans,Math.min(Math.max(0,h-y1),Math.max(0,h-(H-y2))));
                        }
                        if(ans==MAX)out.println("-1");
                        else out.println(ans);
                }
                out.close();
        }
        public static class pair
    {
        int a;
        int b;
        public pair(int val,int index)
        {
            a=val;
            b=index;
        }
    }
        public static class myComp implements Comparator<pair>
        {
                //sort in ascending order.
                public int compare(pair p1,pair p2)
                {
                        if(p1.a==p2.a)
            return 0;
            else if(p1.a<p2.a)
            return -1;
            else
            return 1;
                }
                //sort in descending order.
                // public int compare(pair p1,pair p2)
        // {
        //     if(p1.a==p2.a)
        //     return 0;
        //     else if(p1.a<p2.a)
        //     return 1;
        //     else
        //     return -1;
        // }
        }
        public static long fact(long n)
        {
                long fact=1;
                for(long i=2;i<=n;i++){
```

```
                fact=((fact%mod)*(i%mod))%mod;
        }
        return fact;
    }
    public static long fact(int n)
    {
        long fact=1;
        for(int i=2;i<=n;i++){
            fact=((fact%mod)*(i%mod))%mod;
        }
        return fact;
    }
    public static long kadane(long a[],int n)
    {
        long max_sum=Long.MIN_VALUE,max_end=0;
        for(int i=0;i<n;i++){
            max_end+=a[i];
            if(max_sum<max_end){max_sum=max_end;}
            if(max_end<0){max_end=0;}
        }
        return max_sum;
    }
    public static void DFS(ArrayList<Integer> arr[],int s,boolean visited[])
    {
        visited[s]=true;
        for(int i:arr[s]){
            if(!visited[i]){
                DFS(arr,i,visited);
            }
        }
    }
    public static int BS(int a[],int x,int ii,int jj)
    {
        // int n=a.length;
        int mid=0;
        int i=ii,j=jj,in=0;
        while(i<=j)
        {
            mid=(i+j)/2;
            if(a[mid]<x){
                in=mid+1;
                i=mid+1;
            }
            else
            j=mid-1;
        }
        return in;
    }
    public static int lower_bound(int arr[], int N, int X)
    {
        int mid;
        int low = 0;
        int high = N;
        while(low<high) {
            mid=low+(high-low)/2;
            if(X<=arr[mid]){
                high=mid;
            }
            else{
                low=mid+1;
            }
        }
        if(low<N && arr[low]<X){
        low++;
        }
        out.println(arr[low]);
        return low;
    }
    public static ArrayList<Integer> primeSieve(int n)
    {
        ArrayList<Integer> arr=new ArrayList<>();
        boolean prime[] = new boolean[n + 1];
```

```
        for (int i = 0; i <= n; i++)
        prime[i] = true;
        for (int p = 2; p * p <= n; p++)
        {
            if (prime[p] == true)
            {
                for (int i = p * p; i <= n; i += p)
                prime[i] = false;
            }
        }
        for (int i = 2; i <= n; i++)
        {
            if (prime[i] == true)
            arr.add(i);
        }
        return arr;
        }

        // Fenwick / BinaryIndexed  Tree  USE IT - FenwickTree ft1=new FenwickTree(n);
        public static class FenwickTree
        {
                long farr[];
                int n;
                public FenwickTree(int c)
                {
                        n=c+1;
                        farr=new long[n];
                }
                // public void update_range(int l,int r,long p)
                // {
                //      update(l,p);
                //      update(r+1,(-1)*p);
                // }
                public void update(int x)
                {
                        for(;x<=n;x+=x&(-x))
                        {
                                farr[x]++;
                        }
                }
                public long get(int x)
                {
                        long ans=0;
                        for(;x>0;x-=x&(-x))
                        {
                                ans=ans+farr[x];
                        }
                        return ans;
                }
        }

        //Disjoint Set Union
        public static class DSU
        {
            static int par[],rank[];
            public DSU(int c)
            {
                par=new int[c+1];
                rank=new int[c+1];
                for(int i=0;i<=c;i++)
                {
                    par[i]=i;
                    rank[i]=0;
                }
            }
            public static int find(int a)
            {
                if(a==par[a])
                return a;
                return par[a]=find(par[a]);
            }
            public static void union(int a,int b)
```

```
            {
                int a_rep=find(a),b_rep=find(b);
                if(a_rep==b_rep)
                return;
                if(rank[a_rep]<rank[b_rep])
                par[a_rep]=b_rep;
                else if(rank[a_rep]>rank[b_rep])
                par[b_rep]=a_rep;
                else
                {
                    par[b_rep]=a_rep;
                    rank[a_rep]++;
                }
            }
        }

        //SEGMENT TREE CODE

        // public static void segmentUpdate(int si,int ss,int se,int qs,int qe,long x)
        // {
        //      if(ss>qe || se<qs)return;
        //      if(qs<=ss && qe>=se)
        //      {
        //              seg[si][0]+=1L;
        //              seg[si][1]+=x*x;
        //              seg[si][2]+=2*x;
        //              return;
        //      }
        //      int mid=(ss+se)/2;
        //      segmentUpdate(2*si+1,ss,mid,qs,qe,x);
        //      segmentUpdate(2*si+2,mid+1,se,qs,qe,x);
        // }
        // public static long segmentGet(int si,int ss,int se,int x,long f,long s,long t,long a[])
        // {
        //      if(ss==se && ss==x)
        //      {
        //              f+=seg[si][0];
        //              s+=seg[si][1];
        //              t+=seg[si][2];
        //              long ans=a[x]+(f*((long)x+1L)*((long)x+1L))+s+(t*((long)x+1L));
        //              return ans;
        //      }
        //      int mid=(ss+se)/2;
        //      if(x>mid){
        //              return segmentGet(2*si+2,mid+1,se,x,f+seg[si][0],s+seg[si][1],t+seg[si][2],a);
        //      }else{
        //              return segmentGet(2*si+1,ss,mid,x,f+seg[si][0],s+seg[si][1],t+seg[si][2],a);
        //      }
        // }


    public static class myComp1 implements Comparator<pair1>
    {
                //sort in ascending order.
        public int compare(pair1 p1,pair1 p2)
        {
            if(p1.a==p2.a)
            return 0;
            else if(p1.a<p2.a)
            return -1;
            else
            return 1;
        }
                //sort in descending order.
                // public int compare(pair p1,pair p2)
        // {
        //     if(p1.a==p2.a)
        //     return 0;
        //     else if(p1.a<p2.a)
        //     return 1;
        //     else
        //     return -1;
```

```
        // }
    }
        public static class pair1
    {
        long a;
        long b;
        public pair1(long val,long index)
        {
            a=val;
            b=index;
        }
    }
    public static ArrayList<pair1> mergeIntervals(ArrayList<pair1> arr)
        {
            //****************use this in main function-Collections.sort(arr,new myComp1());
            ArrayList<pair1> a1=new ArrayList<>();
            if(arr.size()<=1)
            return arr;
            a1.add(arr.get(0));
            int i=1,j=0;
            while(i<arr.size())
            {
                if(a1.get(j).b<arr.get(i).a)
                {
                   a1.add(arr.get(i));
                   i++;
                   j++;
                }
                else if(a1.get(j).b>arr.get(i).a && a1.get(j).b>=arr.get(i).b)
                {
                    i++;
                }
                else if(a1.get(j).b>=arr.get(i).a)
                {
                    long a=a1.get(j).a;
                    long b=arr.get(i).b;
                    a1.remove(j);
                    a1.add(new pair1(a,b));
                    i++;
                }
            }
            return a1;
        }
        public static boolean palindrome(String s,int n)
        {
                for(int i=0;i<=n/2;i++){
                        if(s.charAt(i)!=s.charAt(n-i-1)){
                                return false;
                        }
                }
                return true;
        }
        public static long gcd(long a,long b)
        {
            if(b==0)
            return a;
            else
            return gcd(b,a%b);
        }
        public static boolean prime(int n)
        {
            if (n <= 1)
                return false;
            if (n <= 3)
                return true;
            if (n % 2 == 0 || n % 3 == 0)
                return false;
            double sq=Math.sqrt(n);

            for (int i = 5; i <= sq; i = i + 6)
                if (n % i == 0 || n % (i + 2) == 0)
                    return false;
```

```
            return true;
        }
    public static boolean prime(long n)
        {
            if (n <= 1)
                return false;
            if (n <= 3)
                return true;
            if (n % 2 == 0 || n % 3 == 0)
                return false;
            double sq=Math.sqrt(n);

            for (int i = 5; i <= sq; i = i + 6)
                if (n % i == 0 || n % (i + 2) == 0)
                    return false;
            return true;
        }
        public static int gcd(int a,int b)
        {
            if(b==0)
            return a;
            else
            return gcd(b,a%b);
        }
        public static void printArray(long a[])
        {
                for(int i=0;i<a.length;i++){
                        out.print(a[i]+" ");
                }
                out.println();
        }
        public static void printArray(int a[])
        {
                for(int i=0;i<a.length;i++){
                        out.print(a[i]+" ");
                }
                out.println();
        }
        public static void printArray(char a[])
        {
                for(int i=0;i<a.length;i++){
                        out.print(a[i]+" ");
                }
                out.println();
        }
        public static void printArray(boolean a[])
        {
                for(int i=0;i<a.length;i++){
                        out.print(a[i]+" ");
                }
                out.println();
        }
        public static void printArray(int a[][])
        {
                for(int i=0;i<a.length;i++){
                        for(int j=0;j<a[i].length;j++){
                                out.print(a[i][j]+" ");
                        }out.println();
                }
        }
        public static void printArray(char a[][])
        {
                for(int i=0;i<a.length;i++){
                        for(int j=0;j<a[i].length;j++){
                                out.print(a[i][j]+" ");
                        }out.println();
                }
        }
        public static void printArray(ArrayList<Long> arr)
        {
                for(int i=0;i<arr.size();i++){
                        out.print(arr.get(i)+" ");
```

```
                }
                out.println();
        }
        public static void printMapInt(HashMap<Integer,Integer> hm){
                for(Map.Entry<Integer,Integer> e:hm.entrySet()){
                        out.println(e.getKey()+"->"+e.getValue());
                }out.println();
        }
        public static void printMapLong(HashMap<Long,Long> hm){
                for(Map.Entry<Long,Long> e:hm.entrySet()){
                        out.println(e.getKey()+"->"+e.getValue());
                }out.println();
        }
        public static long pwr(long m,long n)
        {
                long res=1;
                m=m%mod;
                if(m==0)
                return 0;
                while(n>0)
                {
                        if((n&1)!=0)
                        {
                                res=(res*m)%mod;
                        }
                        n=n>>1;
                        m=(m*m)%mod;
                }
                return res;
        }
        public static void sort(int[] A)
        {
        int n = A.length;
        Random rnd = new Random();
        for(int i=0; i<n; ++i)
                {
            int tmp = A[i];
            int randomPos = i + rnd.nextInt(n-i);
            A[i] = A[randomPos];
            A[randomPos] = tmp;
        }
        Arrays.sort(A);
    }
    public static void sort(long[] A)
        {
            int n = A.length;
        Random rnd = new Random();
        for(int i=0; i<n; ++i)
                {
            long tmp = A[i];
            int randomPos = i + rnd.nextInt(n-i);
            A[i] = A[randomPos];
            A[randomPos] = tmp;
        }
            Arrays.sort(A);
        }
        public static int I(){return sc.I();}
    public static long L(){return sc.L();}
    public static String S(){return sc.S();}
    public static double D(){return sc.D();}
}
class FastReader {
    BufferedReader br;
    StringTokenizer st;
    public FastReader(){
        br = new BufferedReader(new InputStreamReader(System.in));
    }
    String next(){
        while (st == null || !st.hasMoreElements()){
            try {
                st = new StringTokenizer(br.readLine());
            }
```

```
            catch (IOException e){
                e.printStackTrace();
            }
        }
        return st.nextToken();
    }
    int I(){
        return Integer.parseInt(next());
    }
    long L(){
        return Long.parseLong(next());
    }
    double D(){
         return Double.parseDouble(next());
    }
    String S(){
        String str = "";
        try
        {
            str = br.readLine();
        }
        catch (IOException e)
        {
            e.printStackTrace();
        }
        return str;
    }
}
```