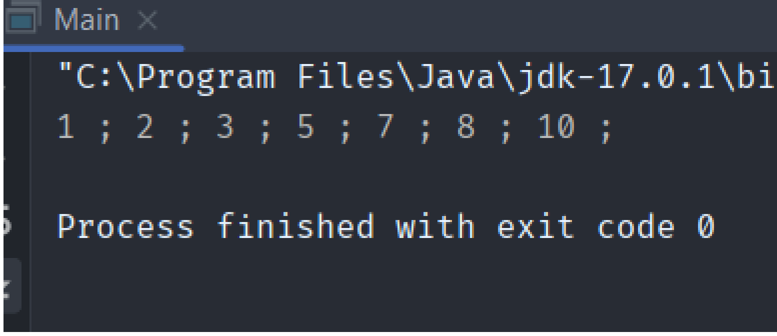
1- With countDown

```
import java.util.concurrent.CountDownLatch;
import java.util.concurrent.Semaphore;

public class Trieur extends Thread {
    private int[] t;
    private int debut, fin;
    private Trieur tr;
    private CountDownLatch a=new CountDownLatch(2);
    private CountDownLatch cp;
    public Trieur(int[] t,CountDownLatch a) {
        this(null, t, 0, t.length - 1,a);

    }
    private Trieur(Trieur tr, int[] t, int debut, int fin,CountDownLatch 
a) {
        this.tr = tr;
        this.t = t;
        this.debut = debut;
        this.fin = fin;
        this.cp=a;
    }
//    public synchronized void notifier() {
//        cp.countDown();
//    }

    public void run() {
        if (fin - debut < 2) {
            if (t[debut] > t[fin]) {
                echanger(debut, fin);
            }
        }else{
            int milieu = debut + (fin - debut) / 2;
            Trieur trieur1 = new Trieur(this, t, debut, milieu,this.a);
            Trieur trieur2 = new Trieur(this, t, milieu + 1, 
fin,this.a);
            trieur1.start();
            trieur2.start();
                try {
                    this.a.await();
                }
                catch(InterruptedException e) {}

            triFusion(debut,fin);
        }
        // if(this.tr != null)
```

```
              //this.notifier();
              //else
                 cp.countDown();
          }
          private void echanger (int a, int b){
              int c = t[a];
              t[a] = t[b];
              t[b] = c;
          }
          private void triFusion(int debut, int fin) {
              int[] tFusion = new int[fin - debut + 1];
              int milieu = (debut + fin) / 2;
              int i1 = debut,
                      i2 = milieu + 1;
              int iFusion = 0;
              while (i1 <= milieu && i2 <= fin)
              {
                  if (t[i1] < t[i2]) {
                      tFusion[iFusion++] = t[i1++];
                  }else {
                      tFusion[iFusion++] = t[i2++];
                  }

              }
              if (i1 > milieu) {
                  for (int i = i2; i <= fin; ) {
                      tFusion[iFusion++] = t[i++];
                  }
              } else {
                  for (int i = i1; i <= milieu; ) {
                      tFusion[iFusion++] = t[i++];
                  }
              }
              for (int i = 0, j = debut; i <= fin - debut; ) {
                  t[j++] = tFusion[i++];
              }
          }
      }
        import java.util.concurrent.CountDownLatch;

public class Main {
    public static void main(String[] args)
    {
        int[] t = {5, 8, 3, 2, 7, 10, 1};
        //int[] t = { 8,5,3,2};
        CountDownLatch a=new CountDownLatch(1);
        Trieur trieur = new Trieur(t,a);
        trieur.start();
        try
```



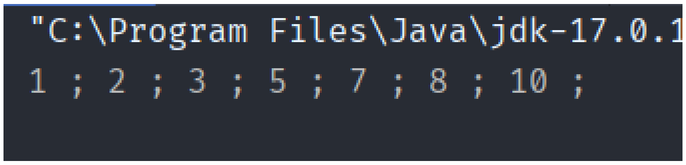
```
"C:\Program Files\Java\jdk-17.0.1\bi
1 ; 2 ; 3 ; 5 ; 7 ; 8 ; 10 ;

Process finished with exit code 0
```

```
        {
            a.await();
        }
        catch(InterruptedException e) {}
        for (int i = 0; i <t.length; i++)
        {
            System.out.print(t[i] + " ; ");
        }
        System.out.println();
    }}
```

2- With Semaphore and countdown

```
import java.util.concurrent.Semaphore;
import java.util.concurrent.CountDownLatch;

public class Trieur extends Thread {
    private int[] t;
    private int debut, fin;
    private Trieur parent;

    private Semaphore finThreadFils = new Semaphore(0);

    private CountDownLatch compteurFinTri;

    public Trieur(int[] t, CountDownLatch cpt) {
        this(null, t, 0, t.length - 1);
        compteurFinTri = cpt;
        this.start();
    }

    private Trieur(Trieur parent, int[] t, int debut, int
            fin) {
        this.parent = parent;
        this.t = t;
        this.debut = debut;
        this.fin = fin;

    }

    public void autoriseSemaphore() {
        this.finThreadFils.release();
    }

    public void run() {
        if (fin - debut < 2) {
            if (t[debut] > t[fin]) {
                echanger(debut, fin);
            }
        } else {

            int milieu = debut + (fin - debut) / 2;
            Trieur trieur1 = new Trieur(this, t, debut,
                    milieu);
            Trieur trieur2 = new Trieur(this, t, milieu + 1,
                    fin);
            trieur1.start();
            trieur2.start();
            try {
                finThreadFils.acquire(2);
            } catch (InterruptedException ie) {
            }
            triFusion(debut, fin);
        }
        if (parent != null) {
            parent.autoriseSemaphore();
        } else {
            compteurFinTri.countDown();
        }
    }

    private void echanger(int i, int j) {
        int valeur = t[i];
        t[i] = t[j];
        t[j] = valeur;
    }

    private void triFusion(int debut, int fin) {
        int[] tFusion = new int[fin - debut + 1];
        int milieu = (debut + fin) / 2;
        int i1 = debut,
                i2 = milieu + 1;
        int iFusion = 0;
        while (i1 <= milieu && i2 <= fin) {
            if (t[i1] < t[i2]) {
                tFusion[iFusion++] = t[i1++];
            } else {
                tFusion[iFusion++] = t[i2++];
            }
        }
        if (i1 > milieu) {

            for (int i = i2; i <= fin; ) {
                tFusion[iFusion++] = t[i++];
            }
        } else {
            for (int i = i1; i <= milieu; ) {
                tFusion[iFusion++] = t[i++];
```

```
                  }
              }

              for (int i = 0, j = debut; i <= fin - debut; ) {
                  t[j++] = tFusion[i++];
              }
          }
      }
import java.util.concurrent.CountDownLatch;

public class Main {
    public static void main(String[] args) {
        int[] t = {5, 8, 3, 2, 7, 10, 1};
        CountDownLatch finTri = new CountDownLatch(1);
        Trieur trieur = new Trieur(t,finTri);
       try {
            finTri.await();

        }
        catch(InterruptedException e) {}
        for (int i = 0; i <t.length; i++) {
            System.out.print(t[i] + " ; ");
        }
        System.out.println();
    }
}
```

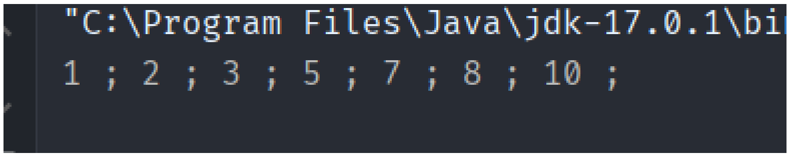


```
"C:\Program Files\Java\jdk-17.0.1
1 ; 2 ; 3 ; 5 ; 7 ; 8 ; 10 ;
```

3- With Semaphore

```
import java.util.concurrent.Semaphore;
import java.util.concurrent.CountDownLatch;

public class Trieur extends Thread {
    private int[] t;
    private int debut, fin;
    private Trieur parent;

    private Semaphore finThreadFils = new Semaphore(0);
    private Semaphore finTri;


    private CountDownLatch compteurFinTri;

    public Trieur(int[] t, Semaphore finTri) {
        this(null, t, 0, t.length - 1);
        this.finTri = finTri;
        this.start();
    }
```

```
private Trieur(Trieur parent, int[] t, int debut, int
        fin) {
    this.parent = parent;
    this.t = t;
    this.debut = debut;
    this.fin = fin;

}

public void autoriseSemaphore() {
    this.finThreadFils.release();
}

public void run() {
    if (fin - debut < 2) {
        if (t[debut] > t[fin]) {
            echanger(debut, fin);
        }
    } else {

        int milieu = debut + (fin - debut) / 2;
        Trieur trieur1 = new Trieur(this, t, debut,
                milieu);
        Trieur trieur2 = new Trieur(this, t, milieu + 1,
                fin);
        trieur1.start();
        trieur2.start();
        try {
            finThreadFils.acquire(2);
        } catch (InterruptedException ie) {
        }
        triFusion(debut, fin);
    }
    if (parent != null) {
        parent.autoriseSemaphore();
    } else {
        finTri.release();
    }
}

private void echanger(int i, int j) {
    int valeur = t[i];
    t[i] = t[j];
    t[j] = valeur;
}

private void triFusion(int debut, int fin) {
    int[] tFusion = new int[fin - debut + 1];
```



```
"C:\Program Files\Java\jdk-17.0.1\bi
1 ; 2 ; 3 ; 5 ; 7 ; 8 ; 10 ;
```

```
           int milieu = (debut + fin) / 2;
           int i1 = debut,
                   i2 = milieu + 1;
           int iFusion = 0;
           while (i1 <= milieu && i2 <= fin) {
               if (t[i1] < t[i2]) {
                   tFusion[iFusion++] = t[i1++];
               } else {
                   tFusion[iFusion++] = t[i2++];
               }
           }
           if (i1 > milieu) {

               for (int i = i2; i <= fin; ) {
                   tFusion[iFusion++] = t[i++];
               }
           } else {
               for (int i = i1; i <= milieu; ) {
                   tFusion[iFusion++] = t[i++];
               }
           }

           for (int i = 0, j = debut; i <= fin - debut; ) {
               t[j++] = tFusion[i++];
           }
       }
   }
1- import java.util.concurrent.CountDownLatch;
   import java.util.concurrent.Semaphore;

   public class Main {
       public static void main(String[] args) {
           int[] t = {5, 8, 3, 2, 7, 10, 1};
           Semaphore finThreadFils = new Semaphore(0);
           Trieur trieur = new Trieur(t,finThreadFils);
          try {
              finThreadFils.acquire(1);

           }
           catch(InterruptedException e) {}
           for (int i = 0; i <t.length; i++) {
               System.out.print(t[i] + " ; ");
           }
           System.out.println();
       }
   }
```