



SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ

İşletim Sistemleri
Proje / Tasarım

GitHup: https://github.com/acar2188/OS_HW1

Grup-7

Fatih Acar-Y225012014
Ufuk Akkaya-Y225012005
Burak Gün-Y225012002
Tarık Şener-Y225012012
Ahmet Kardeşseven-Y225012006

2.Öğretim B Grubu

Proje

Proje Özeti

Projede dört seviyeli öncelikli görevlendirici yapısı oluşturulmuştur. 1 adet İlk Gelen İlk Çalışır (FCFS) algoritması ile çalışan kuyruk ve üç seviyeli geri beslemeli round robin kuyruk bulunmaktadır.

Proje 2 kısımdan oluşmaktadır: `main.java` ve `Process_SW.java`

Main kısımda `giris.txt` okunarak bütün prosesler okunarak `JobDispatchList`'e eklenir. Sonrasında program çalışmaya başlar ve `Thread.sleep((long)1000);` fonksiyonu ile her döngüde 1 saniye geçmesi sağlanır. Her döngüde `tick` değişkeni artırılır ve program zamanı `tick` değişkeni ile takip eder. Her döngünün başında `JobDispatchList` kontrol edilir varış zamanı gelen prosesler ilgili kuyruğa eklenir. **Main**'in geri kalan kısmında görevlendirici algoritmalar bulunur.

Process_SW Proses ile ilgili sınıf yapısı bulunmaktadır. Ayrıca ilgili proseslerin çalışma, askıya alınma ve sonlanması durumlarında ekrana basma fonksiyonları da bu kısımdadır.

Projede projeler `ProcessBuilder` ile oluşturulur.

```
ProcessBuilder PB;
```

`ProcessBuilder` sınıfının `command` fonksiyonu ile hangi programı ve hangi dosyayı açacağımızı belirttik.

```
PB.command("notepad.exe", "cikis.txt");
```

`ProcessBuilder` sınıfının `start` fonksiyonu ile proses başlar. Ve `start` fonksiyonu `java`'nın kendi proses sınıfının nesnesi olarak döner. Biz de kendimiz için tanımladığımız `HW_PB` prosesine atadık.

```
Process HW_PB;
```

```
HW_PB = PB.start();
```

Artık proses aşağıdaki fonksiyonlar ile bekleme, uyandırma ve sonlandırma işlemini yapabilmektedir.

```
HW_PB.wait(1000);
```

```
HW_PB.notify();
```

```
HW_PB.destroy();
```

Programın Çalıştırılması

Proje dizindeki `Out` klasöründe `START.bat` dosyası oluşturduk. İçerisinde çalıştırılabilir JAR dosyasını başlatan komut yazdık (`java -jar "os_hw1.jar"`). Bu dosyaya çift tıklayarak açtığımızda `cmd` ekranı açılır ve JAR dosyasını çalıştıran komut çağrılır ve yazdığımız proje çalışmaya başlar. `Video` klasöründe 4x hızlandırılmış video bulunmaktadır. Ayrıca raporun sonunda ekran görüntüleri bulunmaktadır.

Github Linki

https://github.com/acar2188/OS_HW1

Görevlendirici tarafından kullanılan yapıların tanımı ve tartışılması

Yapı olarak bilgiler giriş.txt dosyasından aşağıdaki formatta okunmaktadır ve ilk olarak job list kuyruğuna eklenir. Ulaşma zamanı gelen görevler job listesinden önceliğine göre ilgili kuyruğa aktarılır Kuyruk isimleri: (FCFS,RQ0,RQ1,RQ2)

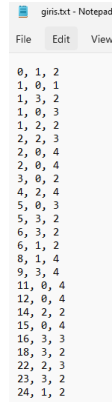


Figure 1:giris.txt içeriği

```
// TODO: DOSYADAN OKUNACAK

try
{
    File myObj = new File("giris.txt");
    Scanner myReader = new Scanner(myObj);
    while (myReader.hasNextLine()) {
        String data = myReader.nextLine();
        //System.out.println(data);
        String[] processData = data.split(", ", 3);
        int arriveTime = Integer.parseInt(processData[0].replaceAll(" ", ""));
        int priority = Integer.parseInt(processData[1].replaceAll(" ", ""));
        int brustTime = Integer.parseInt(processData[2].replaceAll(" ", ""));
        JobDispatchList.add(new Process_SW(pidCounter++, priority , arriveTime, brustTime));
    }
    myReader.close();
}
catch (FileNotFoundException e) {
    System.out.println("An error occurred.");
    e.printStackTrace();
}
```

Figure 2:Dosyadan veri okuma

Program yapısının ve bireysel modüllerin tanımı ve gerekçesi

Program yapısının açıklamasını proje özeti kısmında yazdık.

- Main
 - Dosyadan okuma ve JobDispatchList'i doldurma
 - Tick zamana bağlı prosesleri kuyruklara ekleme
 - Görevlendirici yapısı
- Proses_SW
 - Proses sınıfını barındırıyor
 - Proseslerin state bilgisine bağlı prosesleri başlatma, askıya alma ve sonlandırma yapıyor. Ve bu durumları ekrana basıyor.

Görevlendiricinin tartışılması eksiklikler ve olası iyileştirmeler

Burada bizim gördüğümüz en büyük eksik eğer FCFS'ye çok uzun sürecek bir proses gelirse bütün prosesler aksar. Bunun yerine FCFS kuyruğuna ortalama yanma süresini kapsayacak 5 saniye gibi bir değer belirleyip 5 saniyeyi aşan prosesler alttaki round robin kuyruğuna aktarılabilir. Böylece aşırı yüksek süreli prosesler sistemi kilitleme noktasına getirmez. Olası bir hata da tolere edilmiş olur.

FCFS yüksek öncelikli sıralayıcının çalışması

FCFS yüksek öncelikli bir proses varsa RQ0,RQ1 ve RQ2'den run durumunda olan proses varsa stop edilir, proses sonlandıysa prosesi listeden çıkartır, sonlanmadıysa öncelik düşürülür.

Detayı **main.java** içinde bulunmaktadır.

```
// FCFS Yüksek Öncelikli bir proses varsa
if(!Queue_FCFS.isEmpty())
{
    // Eğer FCFS öncesi RQ0,RQ1 ya RQ2'den run durumunda olan proses varsa stop edilir.
    switch(ActiveQueue)
    {
        case RQ0:
        {
            // Queue boş değilse
            if(!Queue_RQ0.isEmpty())
            {
                // Queue'nun ilk prosesi run durumundaysa
                runProcess = Queue_RQ0.peek();
                if(runProcess.ProcessState == Process_SW.State.Run)
                {
                    // Bu proses durdurulur
                    runProcess.Stop(tick);
                    // Proses sonlandıysa
                    if(runProcess.ProcessState == Process_SW.State.Terminated)
                    {
                        // Prosesi listeden sil.
                        Queue_RQ0.poll();
                    }
                    else // Proses bitmediyse
                    {
                        // Öncelik düşürülür
                        runProcess.Priority++;
                        // Durdurulan proses RQ1 kuyruğuna aktarılır.
                        Queue_RQ1.add( Queue_RQ0.poll());
                    }
                }
            }
        }
    }
}
```

Figure 3:FCFS Algoritması

```
runProcess = Queue_FCFS.peek();
if(runProcess.BurstTime <= 0)
{
    runProcess.Stop(tick); // durdurulmanın ardından sonlanır.
    Queue_FCFS.remove();
    if(!Queue_FCFS.isEmpty())
    {
        runProcess = Queue_FCFS.peek();
    }
    else
    {
        continue;
    }
}
// Proses Start ediliyor.
runProcess.Run(tick);
ActiveQueue = QueueType.FCFS;
```

Figure 4:FCFS Process Çalıştırma

Kullanıcı Geri Beslemeli sıralayıcının çalışması

Geri beslemeli sıralayıcıda her proses 1 saniye çalışarak daha az öncelikli kuyruğa aktarılır. Son kuyrukta bütün prosesler bitene kadar döner.

```
// Proses ilgili kuyruğa eklenir.
switch(process.Priority)
{
    case 0: Queue_FCFS.add(process);
            break;
    case 1: Queue_RQ0.add(process);
            break;
    case 2: Queue_RQ1.add(process);
            break;
    case 3: Queue_RQ2.add(process);
            break;
}

// Başlatılacak görev listesinden kaldırır.
JobDispatchList.remove(1);
add = true;
break;
```

Figure 5:Her durum için tanımlamalar oluşturulmuştur

Kullanıcı Geri Beslemeli sıralayıcının Round Robin modunda çalışması

Yüksek öncelikli proses olmadığında çalışan süreçlerdir.

Detayı **main.java** içinde bulunmaktadır.
RQ0,RQ1,RQ2

```
case RQ0:
{
    // Queue boş değilse
    if(!Queue_RQ0.isEmpty())
    {
        // Queue'nun ilk process'i run durumunda
        runProcess = Queue_RQ0.peek();
        if(runProcess.ProcessState == Process_SW.State.Run)
        {
            // Bu process durdurulur
            runProcess.Stop(tick);
            // Process sonlandığında
            if(runProcess.ProcessState == Process_SW.State.Terminated)
            {
                // Processi listeden sil.
                Queue_RQ0.poll();
            }
            else // Process bitmediyse
            {
                // Öncelik düşürülür
                runProcess.Priority++;
                // Durdurulan process RQ1 kuyruğuna aktarılır.
                Queue_RQ1.add(Queue_RQ0.poll());
            }
        }
    }
}
} break;
```

Figure 6:RQ0 Kuyruğu

```
case RQ1:
{
    // Queue boş değilse
    if(!Queue_RQ1.isEmpty())
    {
        // Queue'nun ilk process'i run durumunda
        runProcess = Queue_RQ1.peek();
        if(runProcess.ProcessState == Process_SW.State.Run)
        {
            // Bu process durdurulur
            runProcess.Stop(tick);
            // Process sonlandığında
            if(runProcess.ProcessState == Process_SW.State.Terminated)
            {
                // Processi listeden sil.
                Queue_RQ1.poll();
            }
            else // Process bitmediyse
            {
                // Öncelik düşürülür
                runProcess.Priority++;
                // Durdurulan process RQ2 kuyruğuna aktarılır.
                Queue_RQ2.add(Queue_RQ1.poll());
            }
        }
    }
}
} break;
```

Figure 7:RQ1 Kuyruğu

```
case RQ2:
{
    // Queue boş değilse
    if(!Queue_RQ2.isEmpty())
    {
        // Queue'nun ilk process'i run durumunda
        runProcess = Queue_RQ2.peek();
        if(runProcess.ProcessState == Process_SW.State.Run)
        {
            // Bu process durdurulur
            runProcess.Stop(tick);
            // Process sonlandığında
            if(runProcess.ProcessState == Process_SW.State.Terminated)
            {
                // Processi listeden sil.
                Queue_RQ2.poll();
            }
            else // Process bitmediyse
            {
                // Durdurulan process RQ2 kuyruğundan başından çıkarılarak sonuna aktarılır.
                Queue_RQ2.add(Queue_RQ2.poll());
            }
        }
    }
}
} break;
```

Figure 8:RQ2 Kuyruğu

Karışık sıralayıcı çalışması

Karışık sıralayıcı bütün görevlendiriciyi kapsamaktadır. Örneğin RQ0'da proses çalışırken FCFS'de proses gelmiş ise görevlendirici RQ0'daki process'i durdurur ve FCFS'deki proses çalıştırılır. Aynı durum RQ0 proses geldiğinde RQ1 ve RQ2 için de yapılır. Onun dışında bütün kuyruklar ayrı bir şekilde çalışır. İlk önce FCFS kuyruğu sonra RQ0 sonra RQ1 sonra RQ2 kuyruğunda işler biter. En joblistte ve kuyruklarda proses kalmadığında program sonlanır.

Kuyruk sınıfı

Aşağıda sınıf içerisinde FCFS ve Round Robin kuyrukları bulunmaktadır.

Detayı **main.java** içinde bulunmaktadır.

```
public class main {  
  
    public enum QueueType {  
        None,  
        FCFS,  
        RQ0,  
        RQ1,  
        RQ2;  
    };  
  
    public static void main(String[] args) {  
        // TODO Auto-generated method stub  
        int pidCounter = 0;  
        QueueType ActiveQueue = QueueType.None;  
  
        List<Process_SW> JobDispatchList = new LinkedList<Process_SW>();  
        Queue<Process_SW> Queue_FCFS = new LinkedList<Process_SW>();  
        Queue<Process_SW> Queue_RQ0 = new LinkedList<Process_SW>();  
        Queue<Process_SW> Queue_RQ1 = new LinkedList<Process_SW>();  
        Queue<Process_SW> Queue_RQ2 = new LinkedList<Process_SW>();  
    }  
}
```

Figure 9:Proses kuyrukları

Proses sınıfı

Aşağıda sınıf içerisinde process bilgileri bulunmaktadır.

Detayı **Process_SW.java** içinde bulunmaktadır.

```
public class Process_SW {  
  
    public static final String ANSI_RESET = "\u001B[0m";  
  
    // Declaring the color  
    // Custom declaration  
    public static final String ANSI_YELLOW = "\u001B[33m";  
  
    public enum State {  
        Create,  
        Ready,  
        Run,  
        Terminated;  
    };  
  
    int PID;  
    int Priority;  
    int ArriveTime;  
    int BrustTime;  
    State ProcessState;  
    boolean FirstRun;  
    ProcessBuilder PB;  
    Process HW_PB;  
  
    Process_SW(int pid, int priority, int arriveTime, int brustTime)  
    {  
        PID = pid;  
        Priority = priority;  
        ArriveTime = arriveTime;  
        BrustTime = brustTime;  
        ProcessState = State.Create;  
        FirstRun = true;  
    }  
}
```

Figure 10:Proses Sınıfı

Ana program - konsol uygulaması

```
C:\Windows\system32\cmd.exe
C:\Users\fatih.acar\eclipse-workspace\OS_HW1_LAST_NEW\OS_HW1_Grup7\out>java -jar "os_hw1.jar"
0 sn proses basladi. (id:0 oncelik:1 kalan sure:2 sn)
1 sn proses askiya alindi. (id:0 oncelik:1 kalan sure:1 sn)
1 sn proses basladi. (id:1 oncelik:0 kalan sure:1 sn)
2 sn proses sonlandi. (id:1 oncelik:0 kalan sure:0 sn)
2 sn proses basladi. (id:3 oncelik:0 kalan sure:3 sn)
3 sn proses yurutuluyor. (id:3 oncelik:0 kalan sure:2 sn)
4 sn proses yurutuluyor. (id:3 oncelik:0 kalan sure:1 sn)
5 sn proses sonlandi. (id:3 oncelik:0 kalan sure:0 sn)
5 sn proses basladi. (id:6 oncelik:0 kalan sure:4 sn)
6 sn proses yurutuluyor. (id:6 oncelik:0 kalan sure:3 sn)
7 sn proses yurutuluyor. (id:6 oncelik:0 kalan sure:2 sn)
8 sn proses yurutuluyor. (id:6 oncelik:0 kalan sure:1 sn)
9 sn proses sonlandi. (id:6 oncelik:0 kalan sure:0 sn)
9 sn proses basladi. (id:7 oncelik:0 kalan sure:4 sn)
10 sn proses yurutuluyor. (id:7 oncelik:0 kalan sure:3 sn)
11 sn proses yurutuluyor. (id:7 oncelik:0 kalan sure:2 sn)
12 sn proses yurutuluyor. (id:7 oncelik:0 kalan sure:1 sn)
13 sn proses sonlandi. (id:7 oncelik:0 kalan sure:0 sn)
13 sn proses basladi. (id:8 oncelik:0 kalan sure:2 sn)
14 sn proses yurutuluyor. (id:8 oncelik:0 kalan sure:1 sn)
15 sn proses sonlandi. (id:8 oncelik:0 kalan sure:0 sn)
15 sn proses basladi. (id:10 oncelik:0 kalan sure:3 sn)
16 sn proses yurutuluyor. (id:10 oncelik:0 kalan sure:2 sn)
17 sn proses yurutuluyor. (id:10 oncelik:0 kalan sure:1 sn)
18 sn proses sonlandi. (id:10 oncelik:0 kalan sure:0 sn)
18 sn proses basladi. (id:16 oncelik:0 kalan sure:4 sn)
19 sn proses yurutuluyor. (id:16 oncelik:0 kalan sure:3 sn)
20 sn proses yurutuluyor. (id:16 oncelik:0 kalan sure:2 sn)
21 sn proses yurutuluyor. (id:16 oncelik:0 kalan sure:1 sn)
22 sn proses sonlandi. (id:16 oncelik:0 kalan sure:0 sn)
22 sn proses basladi. (id:17 oncelik:0 kalan sure:4 sn)
23 sn proses yurutuluyor. (id:17 oncelik:0 kalan sure:3 sn)
24 sn proses yurutuluyor. (id:17 oncelik:0 kalan sure:2 sn)
25 sn proses yurutuluyor. (id:17 oncelik:0 kalan sure:1 sn)
26 sn proses sonlandi. (id:17 oncelik:0 kalan sure:0 sn)
26 sn proses basladi. (id:19 oncelik:0 kalan sure:4 sn)
27 sn proses yurutuluyor. (id:19 oncelik:0 kalan sure:3 sn)
28 sn proses yurutuluyor. (id:19 oncelik:0 kalan sure:2 sn)
29 sn proses yurutuluyor. (id:19 oncelik:0 kalan sure:1 sn)
30 sn proses sonlandi. (id:19 oncelik:0 kalan sure:0 sn)
30 sn proses basladi. (id:13 oncelik:1 kalan sure:2 sn)
31 sn proses askiya alindi. (id:13 oncelik:1 kalan sure:1 sn)
31 sn proses basladi. (id:14 oncelik:1 kalan sure:4 sn)
32 sn proses askiya alindi. (id:14 oncelik:1 kalan sure:3 sn)
32 sn proses basladi. (id:24 oncelik:1 kalan sure:2 sn)
33 sn proses askiya alindi. (id:24 oncelik:1 kalan sure:1 sn)
34 sn proses basladi. (id:4 oncelik:2 kalan sure:2 sn)
35 sn proses askiya alindi. (id:4 oncelik:2 kalan sure:1 sn)
35 sn proses yurutuluyor. (id:0 oncelik:2 kalan sure:1 sn)
36 sn proses sonlandi. (id:0 oncelik:2 kalan sure:0 sn)
37 sn proses basladi. (id:5 oncelik:2 kalan sure:3 sn)
38 sn proses askiya alindi. (id:5 oncelik:2 kalan sure:2 sn)
38 sn proses basladi. (id:9 oncelik:2 kalan sure:4 sn)
39 sn proses askiya alindi. (id:9 oncelik:2 kalan sure:3 sn)
39 sn proses basladi. (id:18 oncelik:2 kalan sure:2 sn)
40 sn proses askiya alindi. (id:18 oncelik:2 kalan sure:1 sn)
40 sn proses basladi. (id:22 oncelik:2 kalan sure:3 sn)
41 sn proses askiya alindi. (id:22 oncelik:2 kalan sure:2 sn)
```

```
C:\Windows\system32\cmd.exe
41 sn proses yurutuluyor. (id:13 oncelik:2 kalan sure:1 sn)
42 sn proses sonlandi. (id:13 oncelik:2 kalan sure:0 sn)
43 sn proses yurutuluyor. (id:14 oncelik:2 kalan sure:3 sn)
44 sn proses askiya alindi. (id:14 oncelik:2 kalan sure:2 sn)
44 sn proses yurutuluyor. (id:24 oncelik:2 kalan sure:1 sn)
45 sn proses sonlandi. (id:24 oncelik:2 kalan sure:0 sn)
46 sn proses basladi. (id:2 oncelik:3 kalan sure:2 sn)
47 sn proses askiya alindi. (id:2 oncelik:3 kalan sure:1 sn)
47 sn proses basladi. (id:11 oncelik:3 kalan sure:2 sn)
48 sn proses askiya alindi. (id:11 oncelik:3 kalan sure:1 sn)
48 sn proses basladi. (id:12 oncelik:3 kalan sure:2 sn)
49 sn proses askiya alindi. (id:12 oncelik:3 kalan sure:1 sn)
49 sn proses basladi. (id:15 oncelik:3 kalan sure:4 sn)
50 sn proses askiya alindi. (id:15 oncelik:3 kalan sure:3 sn)
50 sn proses basladi. (id:20 oncelik:3 kalan sure:3 sn)
51 sn proses askiya alindi. (id:20 oncelik:3 kalan sure:2 sn)
51 sn proses basladi. (id:21 oncelik:3 kalan sure:2 sn)
52 sn proses askiya alindi. (id:21 oncelik:3 kalan sure:1 sn)
52 sn proses basladi. (id:23 oncelik:3 kalan sure:2 sn)
53 sn proses askiya alindi. (id:23 oncelik:3 kalan sure:1 sn)
53 sn proses yurutuluyor. (id:4 oncelik:3 kalan sure:1 sn)
54 sn proses sonlandi. (id:4 oncelik:3 kalan sure:0 sn)
55 sn proses yurutuluyor. (id:5 oncelik:3 kalan sure:2 sn)
56 sn proses askiya alindi. (id:5 oncelik:3 kalan sure:1 sn)
56 sn proses yurutuluyor. (id:9 oncelik:3 kalan sure:3 sn)
57 sn proses askiya alindi. (id:9 oncelik:3 kalan sure:2 sn)
57 sn proses yurutuluyor. (id:18 oncelik:3 kalan sure:1 sn)
58 sn proses sonlandi. (id:18 oncelik:3 kalan sure:0 sn)
59 sn proses yurutuluyor. (id:22 oncelik:3 kalan sure:2 sn)
60 sn proses askiya alindi. (id:22 oncelik:3 kalan sure:1 sn)
60 sn proses yurutuluyor. (id:14 oncelik:3 kalan sure:2 sn)
61 sn proses askiya alindi. (id:14 oncelik:3 kalan sure:1 sn)
61 sn proses yurutuluyor. (id:2 oncelik:3 kalan sure:1 sn)
62 sn proses sonlandi. (id:2 oncelik:3 kalan sure:0 sn)
63 sn proses yurutuluyor. (id:11 oncelik:3 kalan sure:1 sn)
64 sn proses sonlandi. (id:11 oncelik:3 kalan sure:0 sn)
65 sn proses yurutuluyor. (id:12 oncelik:3 kalan sure:1 sn)
66 sn proses sonlandi. (id:12 oncelik:3 kalan sure:0 sn)
67 sn proses yurutuluyor. (id:15 oncelik:3 kalan sure:3 sn)
68 sn proses askiya alindi. (id:15 oncelik:3 kalan sure:2 sn)
68 sn proses yurutuluyor. (id:20 oncelik:3 kalan sure:2 sn)
69 sn proses askiya alindi. (id:20 oncelik:3 kalan sure:1 sn)
69 sn proses yurutuluyor. (id:21 oncelik:3 kalan sure:1 sn)
70 sn proses sonlandi. (id:21 oncelik:3 kalan sure:0 sn)
71 sn proses yurutuluyor. (id:23 oncelik:3 kalan sure:1 sn)
72 sn proses sonlandi. (id:23 oncelik:3 kalan sure:0 sn)
73 sn proses yurutuluyor. (id:5 oncelik:3 kalan sure:1 sn)
74 sn proses sonlandi. (id:5 oncelik:3 kalan sure:0 sn)
75 sn proses yurutuluyor. (id:9 oncelik:3 kalan sure:2 sn)
76 sn proses askiya alindi. (id:9 oncelik:3 kalan sure:1 sn)
76 sn proses yurutuluyor. (id:22 oncelik:3 kalan sure:1 sn)
77 sn proses sonlandi. (id:22 oncelik:3 kalan sure:0 sn)
78 sn proses yurutuluyor. (id:14 oncelik:3 kalan sure:1 sn)
79 sn proses sonlandi. (id:14 oncelik:3 kalan sure:0 sn)
80 sn proses yurutuluyor. (id:15 oncelik:3 kalan sure:2 sn)
81 sn proses askiya alindi. (id:15 oncelik:3 kalan sure:1 sn)
81 sn proses yurutuluyor. (id:20 oncelik:3 kalan sure:1 sn)
82 sn proses sonlandi. (id:20 oncelik:3 kalan sure:0 sn)
83 sn proses yurutuluyor. (id:9 oncelik:3 kalan sure:1 sn)
84 sn proses sonlandi. (id:9 oncelik:3 kalan sure:0 sn)
85 sn proses yurutuluyor. (id:15 oncelik:3 kalan sure:1 sn)
86 sn proses sonlandi. (id:15 oncelik:3 kalan sure:0 sn)
Program Sonlandi.
```