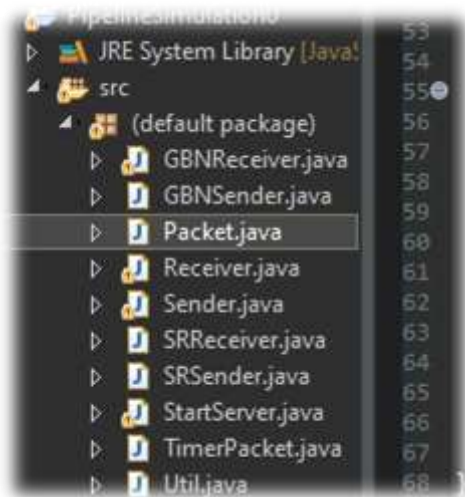


SELECTIVE REPEAT and GO BACK N SIMULATOR



The simulator have two clients that sender and receiver for both Go Back-N and Selective Repeat. Also there is a StartServer class, a TimePacket class, a Packet class and an Util class. The Util class receives and keeps packets. The array is created randomly by computer and it identify that the max size of array is 512. Also, Timerpacket class keeps time out.

When the program is running, it ask the user to choose that which simulator you want to use such as 0-GBN and 1-SR. After that, the user will enter the time out-time. This value can be anything such as 100 or 10000. The packets is determined randomly with maximum size of 512.

On the first testing, assume that the user choose the GO-BACK N protocol. The user enter the value of time out time that 10000. The simulator do not ask the user the expected losses in packages and acknowledge(losses is 0 now). The Simulator start to send the arrays.

To conclude, the simulator keep the time out-time. The data time is 73 ms. The simulator completed to run on 11 steps. There are secreenshots below.

```
Console x
<terminated> StartServer [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (16 Ara 2019 20:59:33)
0 - GBN
1 - SR
protocol> 0
timeout> 10000
windowSize> 35
GBN protocol
Start to receive data
GBN protocol
Start to send data
PKT RECV DAT 24 0
PKT SEND DAT 24 0
[B@3ad02edf
PKT SEND ACK 12 0
PKT RECV ACK 12 0
PKT RECV DAT 24 1
[B@7ca3c15b
PKT SEND DAT 24 1
PKT SEND ACK 12 1
PKT RECV ACK 12 1
PKT RECV DAT 24 2
PKT SEND DAT 24 2
[B@423dfe87
PKT SEND ACK 12 2
PKT RECV ACK 12 2
PKT RECV DAT 23 3
[B@2d8bb0f7
PKT SEND ACK 12 3
PKT RECV ACK 12 3
PKT SEND DAT 23 3
PKT SEND DAT 22 4
PKT RECV DAT 22 4
[B@1696d54f
PKT SEND ACK 12 4
PKT RECV ACK 12 4
PKT RECV DAT 23 5
[B@74dd67df
PKT SEND DAT 23 5
PKT RECV ACK 12 5

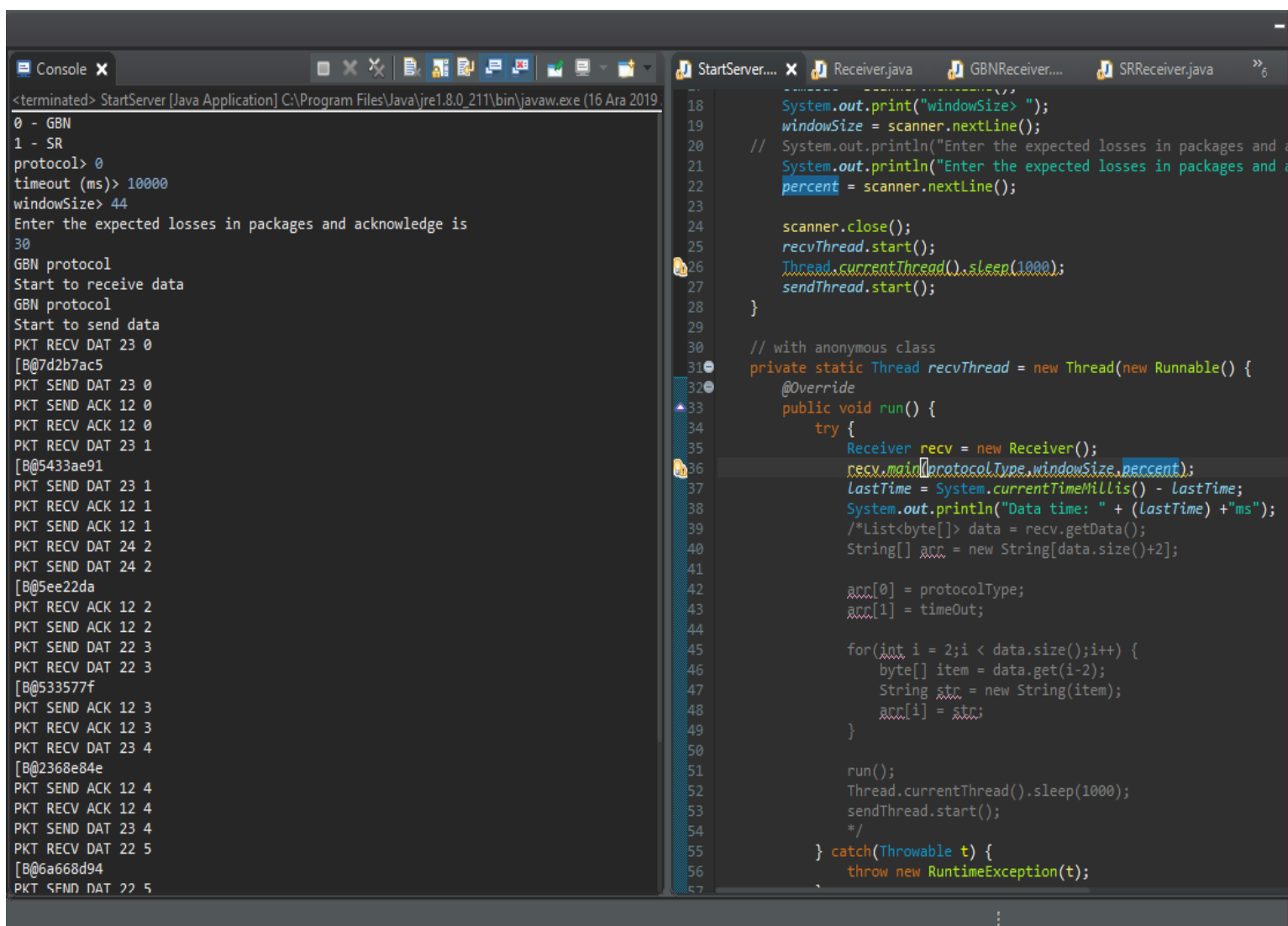
57 //
58
59 // without anonymous class
```

```
Console x
<terminated> StartServer [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (16 Ara 2019 20:59:33)
PKT SEND DAT 22 4
PKT RECV DAT 22 4
[B@1696d54f
PKT SEND ACK 12 4
PKT RECV ACK 12 4
PKT RECV DAT 23 5
[B@74dd67df
PKT SEND DAT 23 5
PKT RECV ACK 12 5
PKT SEND ACK 12 5
PKT RECV DAT 23 6
PKT SEND DAT 23 6
[B@59397685
PKT SEND ACK 12 6
PKT RECV ACK 12 6
PKT RECV DAT 22 7
[B@29c569dd
PKT SEND DAT 22 7
PKT SEND ACK 12 7
PKT RECV ACK 12 7
PKT RECV DAT 23 8
[B@959fd07
PKT SEND ACK 12 8
PKT RECV ACK 12 8
PKT SEND DAT 23 8
PKT RECV DAT 23 9
[B@16e5906c
PKT SEND DAT 23 9
PKT SEND ACK 12 9
PKT RECV ACK 12 9
PKT SEND EOT 12 10
PKT RECV EOT 12 10
PKT SEND EOT 12 10
Finish receiving data
PKT RECV EOT 12 10
Finish sending data
Data time: 73ms

57 //
58
59 // without anonymous class
```

Secondly, we created a code block that include expected losses in packages and acknowledge and the user enter a percentage for Go-Back N protocol. This is %30. Also, time-out time is entered 10000 by user.

When the program send all packets without expected losses packages, the data time is 73 ms. However, when the user enter the expected losses (%30) in both packages and acknowledge, some packets lost. For this reason, the data time is 77 milliseconds and the delay of sending packets is 4 milliseconds. Also, the simulator completed to send the packets on 6 steps.



```
<terminated> StartServer [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (16 Ara 2019)
0 - GBN
1 - SR
protocol> 0
timeout (ms)> 10000
windowSize> 44
Enter the expected losses in packages and acknowledge is
30
GBN protocol
Start to receive data
GBN protocol
Start to send data
PKT RECV DAT 23 0
[B@7d2b7ac5
PKT SEND DAT 23 0
PKT SEND ACK 12 0
PKT RECV ACK 12 0
PKT RECV DAT 23 1
[B@5433ae91
PKT SEND DAT 23 1
PKT RECV ACK 12 1
PKT SEND ACK 12 1
PKT RECV DAT 24 2
PKT SEND DAT 24 2
[B@5ee22da
PKT RECV ACK 12 2
PKT SEND ACK 12 2
PKT SEND DAT 22 3
PKT RECV DAT 22 3
[B@533577f
PKT SEND ACK 12 3
PKT RECV ACK 12 3
PKT RECV DAT 23 4
[B@2368e84e
PKT SEND ACK 12 4
PKT RECV ACK 12 4
PKT SEND DAT 23 4
PKT RECV DAT 22 5
[B@6a668d94
PKT SEND DAT 22 5
```

```
18 System.out.print("windowSize> ");
19 windowSize = scanner.nextLine();
20 // System.out.println("Enter the expected losses in packages and a
21 System.out.println("Enter the expected losses in packages and a
22 percent = scanner.nextLine();
23
24 scanner.close();
25 recvThread.start();
26 Thread.currentThread().sleep(1000);
27 sendThread.start();
28 }
29
30 // with anonymous class
31 private static Thread recvThread = new Thread(new Runnable() {
32     @Override
33     public void run() {
34         try {
35             Receiver rcv = new Receiver();
36             rcv.main(protocolType, windowSize, percent);
37             lastTime = System.currentTimeMillis() - lastTime;
38             System.out.println("Data time: " + (lastTime) + "ms");
39             /*List<byte[]> data = rcv.getData();
40             String[] acc = new String[data.size()+2];
41
42             acc[0] = protocolType;
43             acc[1] = timeOut;
44
45             for(int i = 2; i < data.size(); i++) {
46                 byte[] item = data.get(i-2);
47                 String str = new String(item);
48                 acc[i] = str;
49             }
50
51             run();
52             Thread.currentThread().sleep(1000);
53             sendThread.start();
54             */
55         } catch (Throwable t) {
56             throw new RuntimeException(t);
57         }
```

```
<terminated> StartServer [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (16 Ara 2019)
GBN protocol
Start to send data
PKT RECV DAT 23 0
[B@7d2b7ac5
PKT SEND DAT 23 0
PKT SEND ACK 12 0
PKT RECV ACK 12 0
PKT RECV DAT 23 1
[B@5433ae91
PKT SEND DAT 23 1
PKT RECV ACK 12 1
PKT SEND ACK 12 1
PKT RECV DAT 24 2
PKT SEND DAT 24 2
[B@5ee22da
PKT RECV ACK 12 2
PKT SEND ACK 12 2
PKT SEND DAT 22 3
PKT RECV DAT 22 3
[B@533577f
PKT SEND ACK 12 3
PKT RECV ACK 12 3
PKT RECV DAT 23 4
[B@2368e84e
PKT SEND ACK 12 4
PKT RECV ACK 12 4
PKT SEND DAT 23 4
PKT RECV DAT 22 5
[B@6a668d94
PKT SEND DAT 22 5
PKT RECV ACK 12 5
PKT SEND ACK 12 5
PKT SEND EOT 12 6
PKT RECV EOT 12 6
PKT SEND EOT 12 6
Finish receiving data
PKT RECV EOT 12 6
Data time: 77ms

import java.util.List;
import java.util.Random;
import java.util.Scanner;

public class StartServer {

    private static String protocolType, timeOut, windowSize, percent;
    private static long lastTime;

    public static void main(String[] args) throws Exception {
        System.out.println("0 - GBN");
        System.out.println("1 - SR");
        System.out.print("protocol> ");
        Scanner scanner = new Scanner(System.in);
        protocolType = scanner.nextLine();
        System.out.print("timeout (ms)> ");
        timeOut = scanner.nextLine();
        System.out.print("windowSize> ");
        windowSize = scanner.nextLine();
        // System.out.println("Enter the expected losses in packages and ack
        System.out.println("Enter the expected losses in packages and ack
        percent = scanner.nextLine();

        scanner.close();
        recvThread.start();
        Thread.currentThread().sleep(1000);
        sendThread.start();
    }

    // with anonymous class
    private static Thread recvThread = new Thread(new Runnable() {
        @Override
        public void run() {
            try {
                Receiver recv = new Receiver();
                recv.main(protocolType, windowSize, percent);
                lastTime = System.currentTimeMillis() - lastTime;
                System.out.println("Data time: " + (lastTime) + "ms");
                /*List<byte[]> data = recv.getData();
                String[] arr = new String[data.size() / 10];
```

On the other testing for Go-Back N protocol, the user enter the expected losses (%50) in both packages and acknowledge, the data time is 84 milliseconds and the delay of sending packets is 11 milliseconds. Also, there are 8 steps.

```
Console X
<terminated> StartServer [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (17 Ara 2019 19:00)
0 - GBN
1 - SR
protocol> 0
timeout (ms)> 10000
windowSize> 44
Enter the expected losses in packages and acknowledge is
50
GBN protocol
Start to receive data
GBN protocol
Start to send data
PKT RECV DAT 23 0
PKT SEND DAT 23 0
[B@38647d7b
PKT SEND ACK 12 0
PKT RECV ACK 12 0
PKT RECV DAT 22 1
[B@69261251
PKT SEND ACK 12 1
PKT RECV ACK 12 1
PKT SEND DAT 22 1
PKT RECV DAT 23 2
PKT SEND DAT 23 2
[B@409c38e5
PKT SEND ACK 12 2
PKT RECV ACK 12 2
PKT RECV DAT 23 3
[B@381bf639
PKT SEND DAT 23 3
PKT SEND ACK 12 3
PKT RECV ACK 12 3
PKT RECV DAT 23 4
[B@230fcbe4
PKT SEND ACK 12 4
PKT SEND DAT 23 4
PKT RECV ACK 12 4
PKT RECV DAT 21 5
[B@7a62e14e

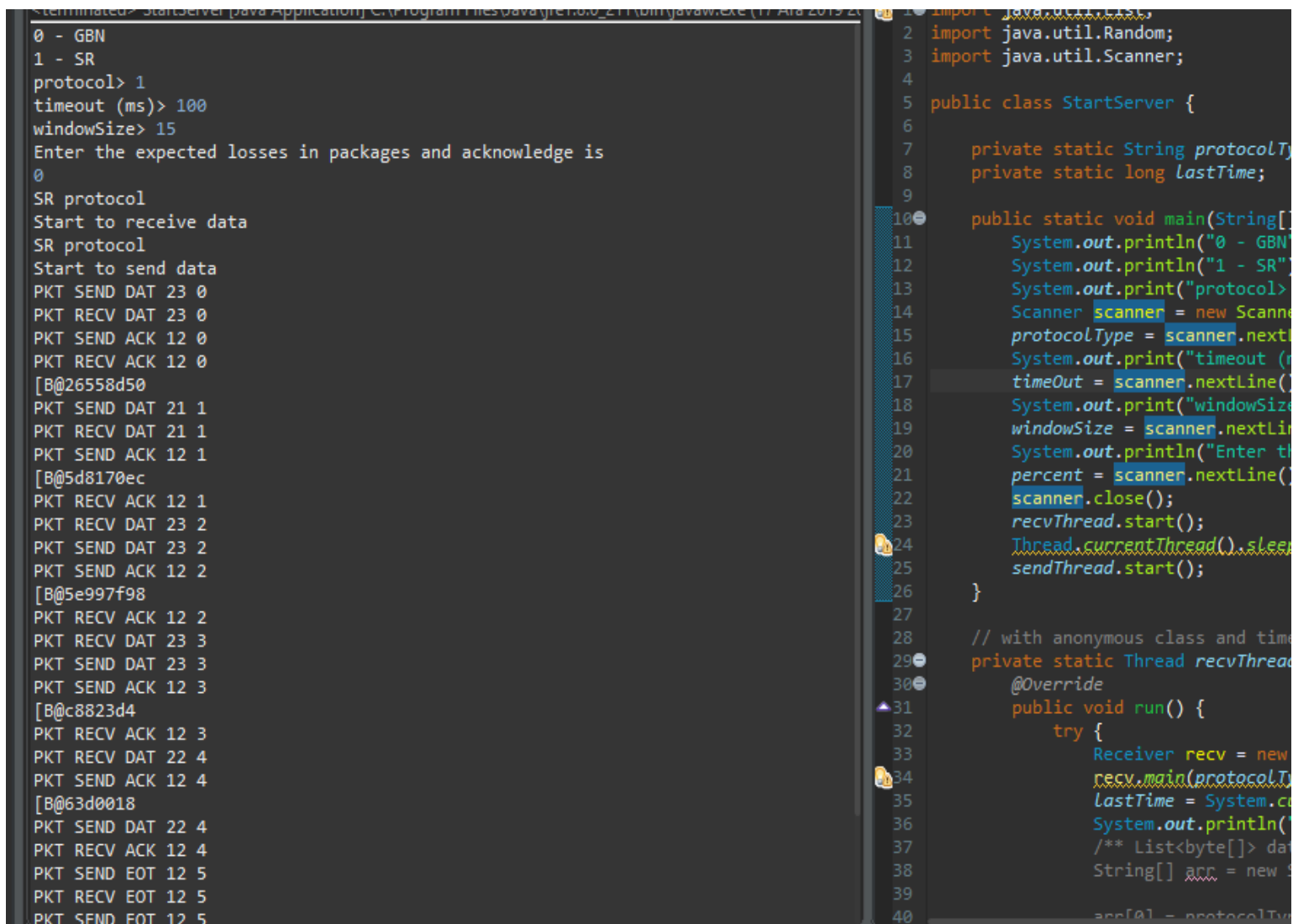
StartServer.java
1 import java.util.List;
2 import java.util.Random;
3 import java.util.Scanner;
4
5 public class StartServer {
6
7     private static String protocolType;
8     private static long lastTime;
9
10    public static void main(String[] args) {
11        System.out.println("0 - GBN");
12        System.out.println("1 - SR");
13        System.out.print("protocol> ");
14        Scanner scanner = new Scanner(System.in);
15        protocolType = scanner.nextLine();
16        System.out.print("timeout (ms)> ");
17        timeout = scanner.nextInt();
18        System.out.print("windowSize> ");
19        windowSize = scanner.nextInt();
20        System.out.println("Enter the expected losses in packages and acknowledge is");
21        percent = scanner.nextInt();
22        scanner.close();
23        recvThread.start();
24        Thread.currentThread().sleep(10000);
25        sendThread.start();
26    }
27
28    // with anonymous class and time
29    private static Thread recvThread = new Thread() {
30        @Override
31        public void run() {
32            try {
33                Receiver rcv = new Receiver(protocolType, lastTime);
34                rcv.main(protocolType, lastTime);
35                lastTime = System.currentTimeMillis();
36                System.out.println("Data time: " + lastTime);
37                /** List<byte[]> data = new ArrayList<>();
38                String[] acks = new String[100];
39                for (int i = 0; i < acks.length; i++) {
40                    acks[i] = protocolType + " " + i + " " + lastTime;
41                }
42                data.add(acks);
43            } catch (Exception e) {
44                e.printStackTrace();
45            }
46        }
47    };
48}
```

```
Console X
<terminated> StartServer [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (17 Ara 2019 19:00)
PKT RECV ACK 12 0
PKT RECV DAT 22 1
[B@69261251
PKT SEND ACK 12 1
PKT RECV ACK 12 1
PKT SEND DAT 22 1
PKT RECV DAT 23 2
PKT SEND DAT 23 2
[B@409c38e5
PKT SEND ACK 12 2
PKT RECV ACK 12 2
PKT RECV DAT 23 3
[B@381bf639
PKT SEND DAT 23 3
PKT SEND ACK 12 3
PKT RECV ACK 12 3
PKT RECV DAT 23 4
[B@230fcbe4
PKT SEND ACK 12 4
PKT SEND DAT 23 4
PKT RECV ACK 12 4
PKT RECV DAT 21 5
[B@7a62e14e
PKT SEND ACK 12 5
PKT RECV ACK 12 5
PKT SEND DAT 21 5
PKT SEND DAT 23 6
PKT RECV DAT 23 6
[B@5b98b5f7
PKT SEND ACK 12 6
PKT RECV ACK 12 6
PKT SEND EOT 12 7
PKT RECV EOT 12 7
PKT SEND EOT 12 7
Finish receiving data
PKT RECV EOT 12 7
Data time: 84ms
Finish sending data

StartServer.java
1 import java.util.List;
2 import java.util.Random;
3 import java.util.Scanner;
4
5 public class StartServer {
6
7     private static String protocolType;
8     private static long lastTime;
9
10    public static void main(String[] args) {
11        System.out.println("0 - GBN");
12        System.out.println("1 - SR");
13        System.out.print("protocol> ");
14        Scanner scanner = new Scanner(System.in);
15        protocolType = scanner.nextLine();
16        System.out.print("timeout (ms)> ");
17        timeout = scanner.nextInt();
18        System.out.print("windowSize> ");
19        windowSize = scanner.nextInt();
20        System.out.println("Enter the expected losses in packages and acknowledge is");
21        percent = scanner.nextInt();
22        scanner.close();
23        recvThread.start();
24        Thread.currentThread().sleep(10000);
25        sendThread.start();
26    }
27
28    // with anonymous class and time
29    private static Thread recvThread = new Thread() {
30        @Override
31        public void run() {
32            try {
33                Receiver rcv = new Receiver(protocolType, lastTime);
34                rcv.main(protocolType, lastTime);
35                lastTime = System.currentTimeMillis();
36                System.out.println("Data time: " + lastTime);
37                /** List<byte[]> data = new ArrayList<>();
38                String[] acks = new String[100];
39                for (int i = 0; i < acks.length; i++) {
40                    acks[i] = protocolType + " " + i + " " + lastTime;
41                }
42                data.add(acks);
43            } catch (Exception e) {
44                e.printStackTrace();
45            }
46        }
47    };
48}
```

On the last testing, assume that the user choose the Selective Repeat protocol. At the beginning, the user enter the value of time out time and do not enter any expected losses in packages and knowledge (losses is 0 now). The Simulator start to send the arrays.

At the end, the simulator keep the time out time. The data time is 59 ms. The simulator completed to send the packets on 6 steps. There are screenshots below.



```
0 - GBN
1 - SR
protocol> 1
timeout (ms)> 100
windowSize> 15
Enter the expected losses in packages and acknowledge is
0
SR protocol
Start to receive data
SR protocol
Start to send data
PKT SEND DAT 23 0
PKT RECV DAT 23 0
PKT SEND ACK 12 0
PKT RECV ACK 12 0
[B@26558d50
PKT SEND DAT 21 1
PKT RECV DAT 21 1
PKT SEND ACK 12 1
[B@5d8170ec
PKT RECV ACK 12 1
PKT RECV DAT 23 2
PKT SEND DAT 23 2
PKT SEND ACK 12 2
[B@5e997f98
PKT RECV ACK 12 2
PKT RECV DAT 23 3
PKT SEND DAT 23 3
PKT SEND ACK 12 3
[B@c8823d4
PKT RECV ACK 12 3
PKT RECV DAT 22 4
PKT SEND ACK 12 4
[B@63d0018
PKT SEND DAT 22 4
PKT RECV ACK 12 4
PKT SEND EOT 12 5
PKT RECV EOT 12 5
PKT SEND EOT 12 5
```

```
1 import java.util.List;
2 import java.util.Random;
3 import java.util.Scanner;
4
5 public class StartServer {
6
7     private static String protocolType;
8     private static long lastTime;
9
10    public static void main(String[] args) {
11        System.out.println("0 - GBN");
12        System.out.println("1 - SR");
13        System.out.print("protocol> ");
14        Scanner scanner = new Scanner(System.in);
15        protocolType = scanner.nextLine();
16        System.out.print("timeout (ms)> ");
17        timeOut = scanner.nextInt();
18        System.out.print("windowSize> ");
19        windowSize = scanner.nextInt();
20        System.out.println("Enter the expected losses in packages and acknowledge is");
21        percent = scanner.nextInt();
22        scanner.close();
23        recvThread.start();
24        Thread.currentThread().sleep(timeOut);
25        sendThread.start();
26    }
27
28    // with anonymous class and time
29    private static Thread recvThread;
30    @Override
31    public void run() {
32        try {
33            Receiver recv = new Receiver(protocolType, lastTime, System.currentTimeMillis());
34            recv.main(protocolType, lastTime, System.currentTimeMillis());
35            System.out.println("Data received:");
36            /** List<byte[]> data */
37            String[] acc = new String[windowSize];
38            for (int i = 0; i < acc.length; i++) {
39                acc[i] = protocolType + " " + i + " " + lastTime + " " + System.currentTimeMillis();
40            }
41        } catch (Exception e) {
42            e.printStackTrace();
43        }
44    }
45}
```

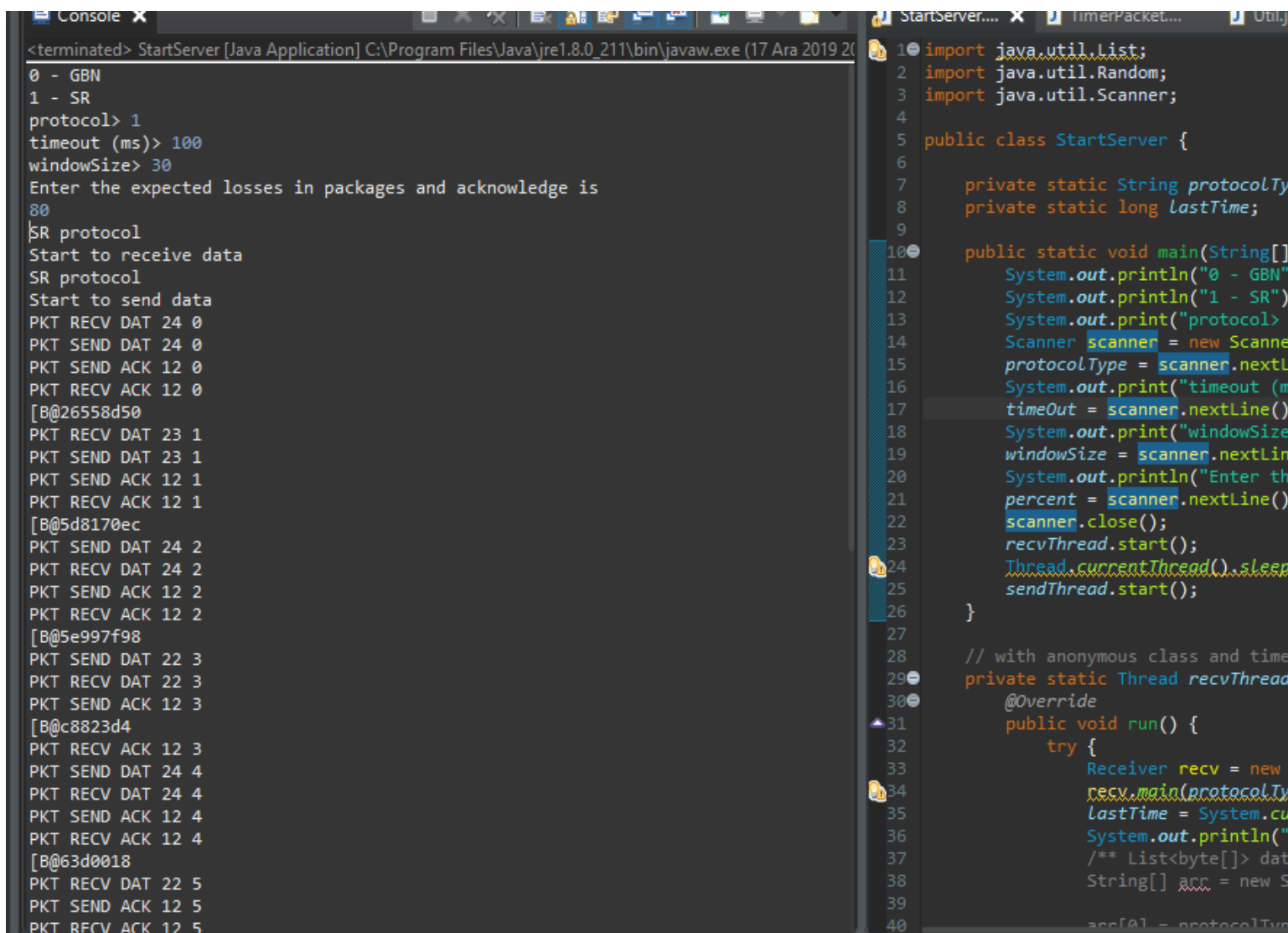


```
Console X
<terminated> StartServer [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (17 Ara 2019 20
Enter the expected losses in packages and acknowledge is
0
SR protocol
Start to receive data
SR protocol
Start to send data
PKT SEND DAT 23 0
PKT RECV DAT 23 0
PKT SEND ACK 12 0
PKT RECV ACK 12 0
[B@26558d50
PKT SEND DAT 21 1
PKT RECV DAT 21 1
PKT SEND ACK 12 1
[B@5d8170ec
PKT RECV ACK 12 1
PKT RECV DAT 23 2
PKT SEND DAT 23 2
PKT SEND ACK 12 2
[B@5e997f98
PKT RECV ACK 12 2
PKT RECV DAT 23 3
PKT SEND DAT 23 3
PKT SEND ACK 12 3
[B@c8823d4
PKT RECV ACK 12 3
PKT RECV DAT 22 4
PKT SEND ACK 12 4
[B@63d0018
PKT SEND DAT 22 4
PKT RECV ACK 12 4
PKT SEND EOT 12 5
PKT RECV EOT 12 5
PKT SEND EOT 12 5
Finish receiving data
Data time: 59ms
PKT RECV EOT 12 5
Finish sending data

StartServer.... X TimerPa
1 import java.util.List
2 import java.util.Rand
3 import java.util.Scan
4
5 public class StartSer
6
7     private static St
8     private static lo
9
10
11     public static voi
12         System.out.pr
13         System.out.pr
14         System.out.pr
15         Scanner scanr
16         protocolType
17         System.out.pr
18         timeout = sca
19         System.out.pr
20         windowSize =
21         System.out.pr
22         percent = sca
23         scanner.close
24         recvThread.st
25         Thread.curren
26         sendThread.st
27     }
28
29     // with anonymous
30     private static Th
31     @Override
32     public void r
33     try {
34         Recei
35         recv.
36         last7
37         System
38         /** L
39         Strin
```

When the program send all packets without expected losses packages with Selective Repeat protocol, the data time is 59 milliseconds. If the user enter the expected losses (%50) in both packages and acknowledge, some packets lost. Therefore, the data time is 73 milliseconds and the delay of sending packets is 14 milliseconds. Also, the simulator completed to send the packets on 11 steps.

The last example for Selective Repeat protocol, the user enter the expected losses (%80) in both packages and acknowledge, the data time is 85 ms and the delay of sending packets is 26 milliseconds. Also, there are 11 steps to finish sending data.



```
<terminated> StartServer [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (17 Ara 2019 20:00)
0 - GBN
1 - SR
protocol> 1
timeout (ms)> 100
windowSize> 30
Enter the expected losses in packages and acknowledge is
80
SR protocol
Start to receive data
SR protocol
Start to send data
PKT RECV DAT 24 0
PKT SEND DAT 24 0
PKT SEND ACK 12 0
PKT RECV ACK 12 0
[B@26558d50
PKT RECV DAT 23 1
PKT SEND DAT 23 1
PKT SEND ACK 12 1
PKT RECV ACK 12 1
[B@5d8170ec
PKT SEND DAT 24 2
PKT RECV DAT 24 2
PKT SEND ACK 12 2
PKT RECV ACK 12 2
[B@5e997f98
PKT SEND DAT 22 3
PKT RECV DAT 22 3
PKT SEND ACK 12 3
[B@c8823d4
PKT RECV ACK 12 3
PKT SEND DAT 24 4
PKT RECV DAT 24 4
PKT SEND ACK 12 4
PKT RECV ACK 12 4
[B@63d0018
PKT RECV DAT 22 5
PKT SEND ACK 12 5
PKT RECV ACK 12 5

1 import java.util.List;
2 import java.util.Random;
3 import java.util.Scanner;
4
5 public class StartServer {
6
7     private static String protocolType;
8     private static long lastTime;
9
10    public static void main(String[] args) {
11        System.out.println("0 - GBN");
12        System.out.println("1 - SR");
13        System.out.print("protocol> ");
14        Scanner scanner = new Scanner(System.in);
15        protocolType = scanner.nextLine();
16        System.out.print("timeout (ms)> ");
17        timeout = scanner.nextLine();
18        System.out.print("windowSize> ");
19        windowSize = scanner.nextLine();
20        System.out.println("Enter the expected losses in packages and acknowledge is");
21        percent = scanner.nextLine();
22        scanner.close();
23        recvThread.start();
24        Thread.currentThread().sleep(26);
25        sendThread.start();
26    }
27
28    // with anonymous class and timer
29    private static Thread recvThread;
30    @Override
31    public void run() {
32        try {
33            Receiver recv = new Receiver(protocolType, lastTime, System.currentTimeMillis());
34            recv.main(protocolType, lastTime, System.currentTimeMillis());
35            System.out.println("Data received:");
36            /** List<byte[]> data */
37            String[] arr = new String[windowSize];
38            for (int i = 0; i < arr.length; i++) {
39                arr[i] = protocolType + " " + i + " ";
40            }
41        } catch (Exception e) {
42            e.printStackTrace();
43        }
44    }
45}
```

Console X

```
<terminated> StartServer [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (17 Ara 2019 20:10:18)
PKT RECV ACK 12 3
PKT SEND DAT 24 4
PKT RECV DAT 24 4
PKT SEND ACK 12 4
PKT RECV ACK 12 4
[B@63d0018
PKT RECV DAT 22 5
PKT SEND ACK 12 5
PKT RECV ACK 12 5
[B@50ed1623
PKT SEND DAT 22 5
PKT SEND DAT 23 6
PKT RECV DAT 23 6
PKT SEND ACK 12 6
[B@459dd619
PKT RECV ACK 12 6
PKT RECV DAT 22 7
PKT SEND DAT 22 7
PKT SEND ACK 12 7
PKT RECV ACK 12 7
[B@8c6dbfd
PKT SEND DAT 23 8
PKT RECV DAT 23 8
PKT SEND ACK 12 8
[B@6615273d
PKT RECV ACK 12 8
PKT RECV DAT 22 9
PKT SEND DAT 22 9
PKT SEND ACK 12 9
PKT RECV ACK 12 9
[B@5ef169d2
PKT SEND EOT 12 10
PKT RECV EOT 12 10
PKT SEND EOT 12 10
Finish receiving data
PKT RECV EOT 12 10
Finish sending data
Data time: 85ms
```

StartServer.... X TimerPacket... Util.java

```
1 import java.util.List;
2 import java.util.Random;
3 import java.util.Scanner;
4
5 public class StartServer {
6
7     private static String protocolType;
8     private static long lastTime;
9
10    public static void main(String[] args) {
11        System.out.println("0 - GBN");
12        System.out.println("1 - SR");
13        System.out.print("protocol> ");
14        Scanner scanner = new Scanner(System.in);
15        protocolType = scanner.nextLine();
16        System.out.print("timeout (ms)> ");
17        timeOut = scanner.nextLine();
18        System.out.print("windowSize> ");
19        windowSize = scanner.nextLine();
20        System.out.println("Enter the percent");
21        percent = scanner.nextLine();
22        scanner.close();
23        recvThread.start();
24        Thread.currentThread().sleep(1000);
25        sendThread.start();
26    }
27
28    // with anonymous class and time ca
29    private static Thread recvThread =
30        @Override
31        public void run() {
32            try {
33                Receiver recv = new Receiver();
34                recv.main(protocolType, lastTime);
35                lastTime = System.currentTimeMillis();
36                System.out.println("Data received");
37                /** List<byte[]> data = new ArrayList<>();
38                String[] arr = new String[windowSize];
39                for (int i = 0; i < windowSize; i++) {
40                    arr[i] = protocolType;
```