

/\* This program creates a linked list that loads numbers 0-9 and prints it to the console. It then prompts the user to enter the amount of integers they want in their list. Once they enter an amount, the program prompts them to enter a value for each list item. Once they've entered each value, the program prints the list. I used the Scanner class for the user input, and created classes for node and list and used the methods from those classes to create the lists and show them on the screen. \*/

### **Main Method**

```
import java.util.Scanner;

public class babyList{
    public static void main(String[] args){
        node nodeA = new node();
        list listA = new list();
        listA.init();
        for(int i = 0; i < 10; i++){
            listA.addLast(i);
        }
        System.out.println("Contents of linked list A:");
        listA.showList();
        System.out.println();

        Scanner scanner = new Scanner(System.in);

        System.out.println("Enter the number of integers in your list. ");
        int size = scanner.nextInt();

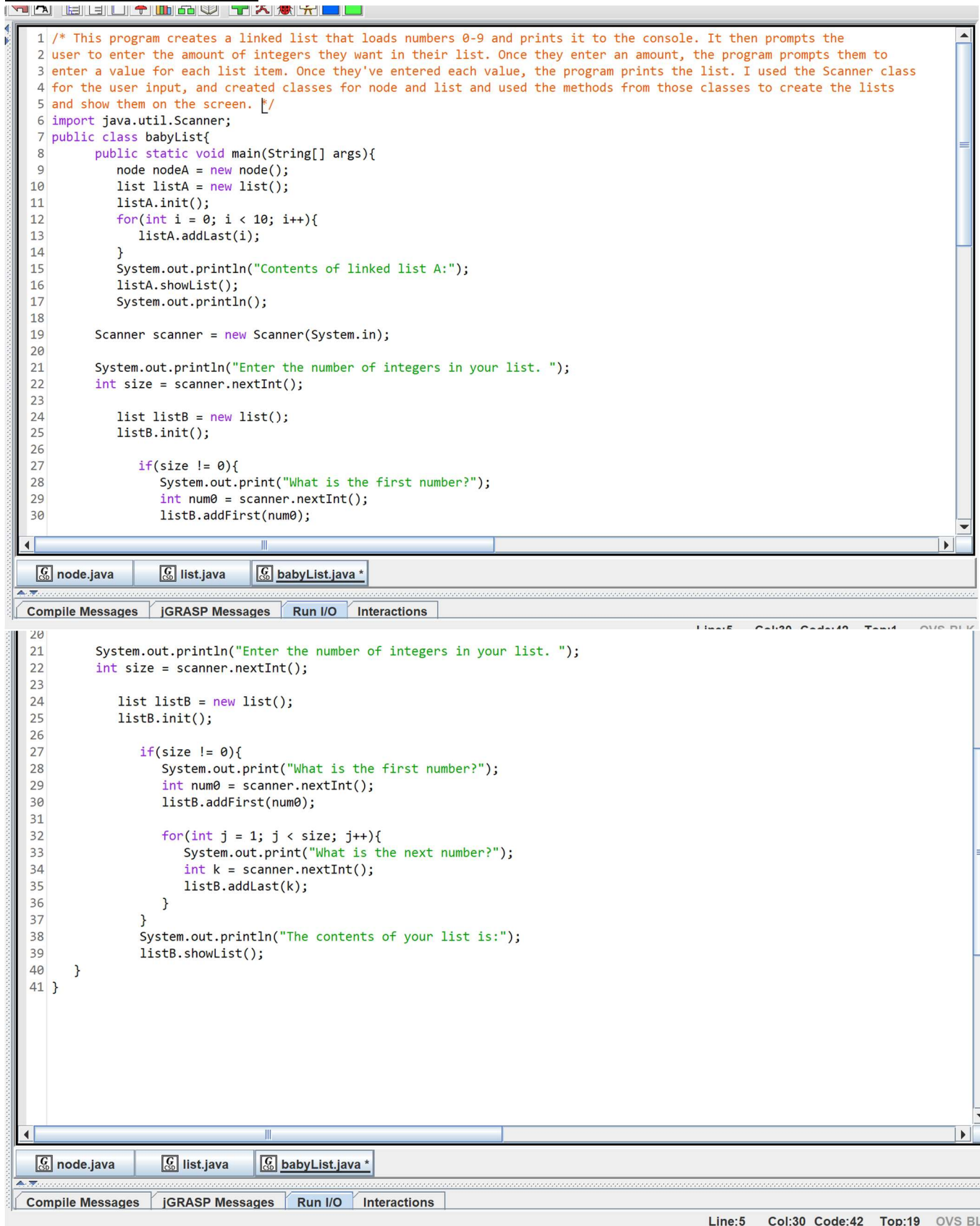
        list listB = new list();
        listB.init();

        if(size != 0){
            System.out.print("What is the first number?");
            int num0 = scanner.nextInt();
            listB.addFirst(num0);

            for(int j = 1; j < size; j++){
                System.out.print("What is the next number?");
                int k = scanner.nextInt();
                listB.addLast(k);
            }
        }
        System.out.println("The contents of your list is:");
        listB.showList();
    }
}
```

```
}  
}
```

## Main Method Screenshots



```
1  /* This program creates a linked list that loads numbers 0-9 and prints it to the console. It then prompts the  
2  user to enter the amount of integers they want in their list. Once they enter an amount, the program prompts them to  
3  enter a value for each list item. Once they've entered each value, the program prints the list. I used the Scanner class  
4  for the user input, and created classes for node and list and used the methods from those classes to create the lists  
5  and show them on the screen. */  
6  import java.util.Scanner;  
7  public class babyList{  
8      public static void main(String[] args){  
9          node nodeA = new node();  
10         list listA = new list();  
11         listA.init();  
12         for(int i = 0; i < 10; i++){  
13             listA.addLast(i);  
14         }  
15         System.out.println("Contents of linked list A:");  
16         listA.showList();  
17         System.out.println();  
18  
19         Scanner scanner = new Scanner(System.in);  
20  
21         System.out.println("Enter the number of integers in your list. ");  
22         int size = scanner.nextInt();  
23  
24         list listB = new list();  
25         listB.init();  
26  
27         if(size != 0){  
28             System.out.print("What is the first number?");  
29             int num0 = scanner.nextInt();  
30             listB.addFirst(num0);  
31  
32             for(int j = 1; j < size; j++){  
33                 System.out.print("What is the next number?");  
34                 int k = scanner.nextInt();  
35                 listB.addLast(k);  
36             }  
37         }  
38         System.out.println("The contents of your list is:");  
39         listB.showList();  
40     }  
41 }
```

node.java list.java babyList.java \*

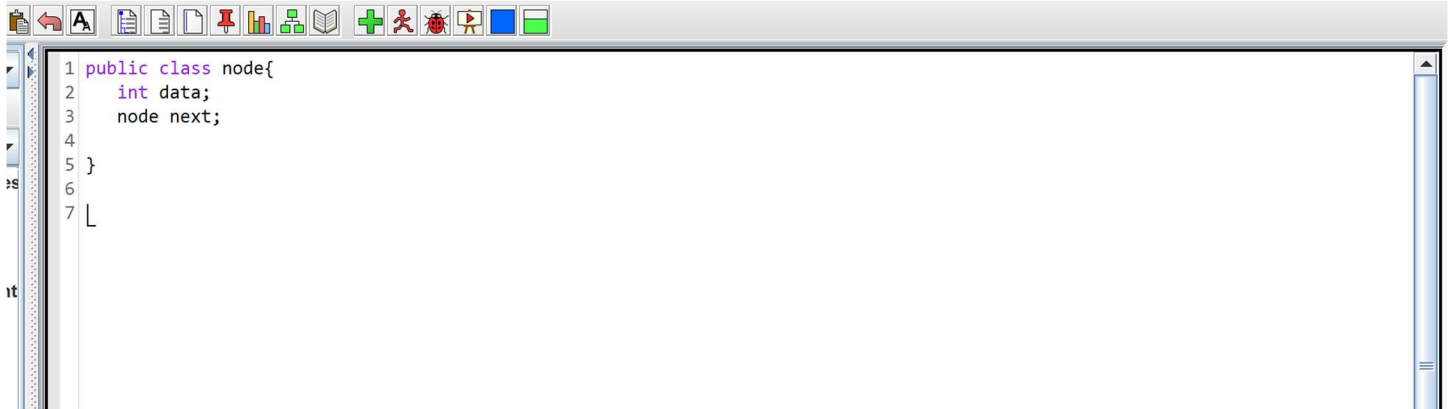
Compile Messages JGRASP Messages Run I/O Interactions

Line:5 Col:30 Code:42 Top:19 OVS BL

### Node Class

```
public class node{  
    int data;  
    node next;  
  
}
```

### Node Class Screenshot



### List Class Code

```
public class list{  
    int data;  
    node next;  
    node curr;  
    node front;  
    node tail;  
    node newNode;  
  
    public void init(){  
        front = null;  
    }  
  
    public node makeNode(int data){  
        newNode = new node();  
        newNode.data = data;  
        newNode.next = null;  
  
        return newNode;  
    }  
  
    public node addFirst(int data){  
        front = makeNode(data);  
        tail = front;  
        return front;  
    }
```

```
}
```

```
public node addLast(int data){  
    if(front == null){  
        front = makeNode(data);  
    }  
    else{  
        tail = findTail();  
        tail.next = makeNode(data);  
        tail = tail.next;  
        tail.next = null;  
    }  
    return tail;  
}
```

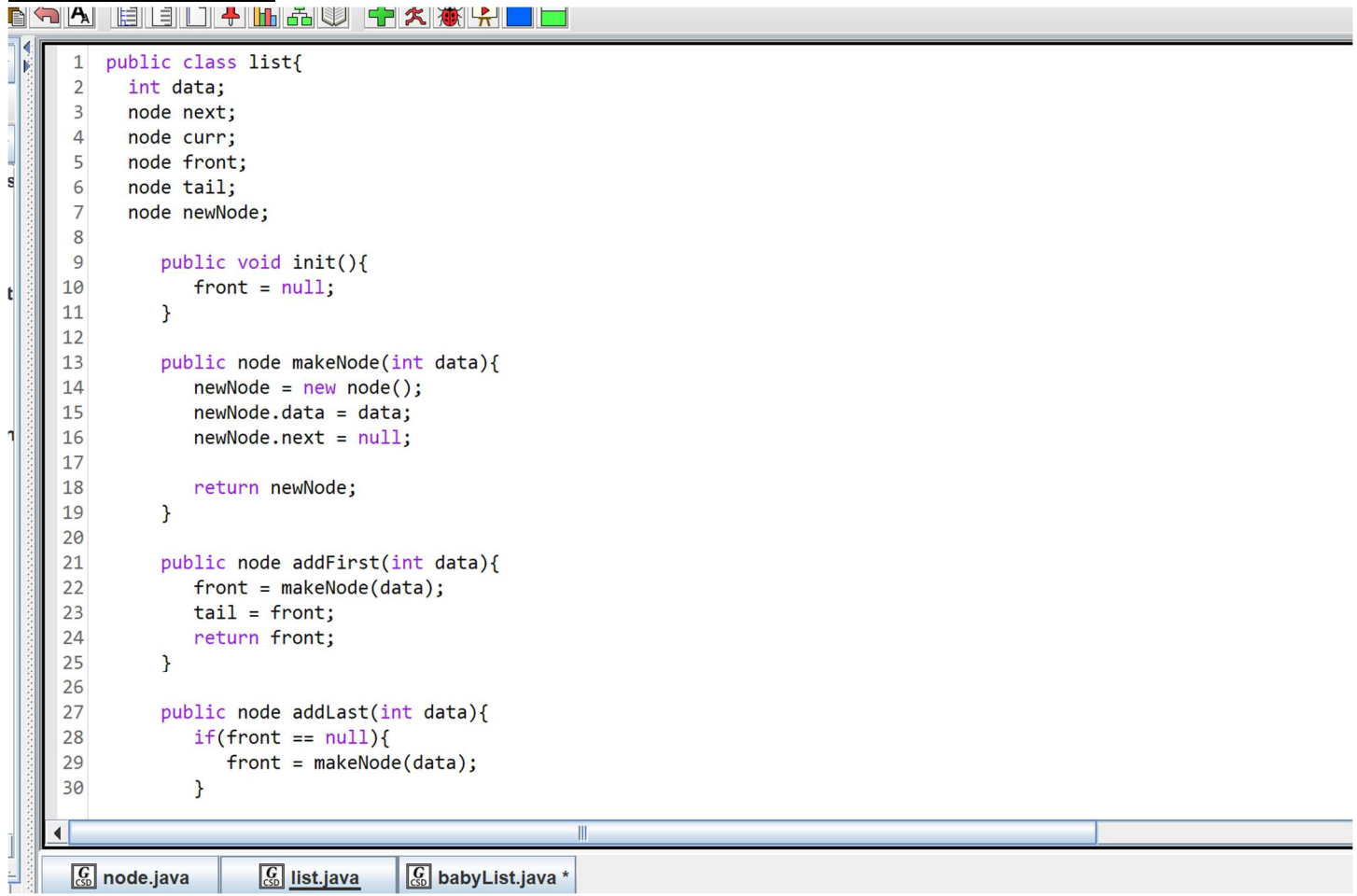
```
public node findTail(){  
    node curr;  
    curr = front;  
    while(curr.next != null){  
        curr = curr.next;  
    }  
    return curr;  
}
```

```
public void showList(){  
    node curr;  
    curr = front;  
    while(curr != null){  
        System.out.println(curr.data);  
        curr = curr.next;  
    }  
}
```

```
public void buildList(int length){  
    int j;  
    init();  
  
    for(j = 0; j < length; j++){  
        if(j == 0){  
            node front = makeNode(j);  
        }  
        else{  
            tail = findTail();  
            tail.next = makeNode(j);  
        }  
    }
```

```
}  
}  
}
```

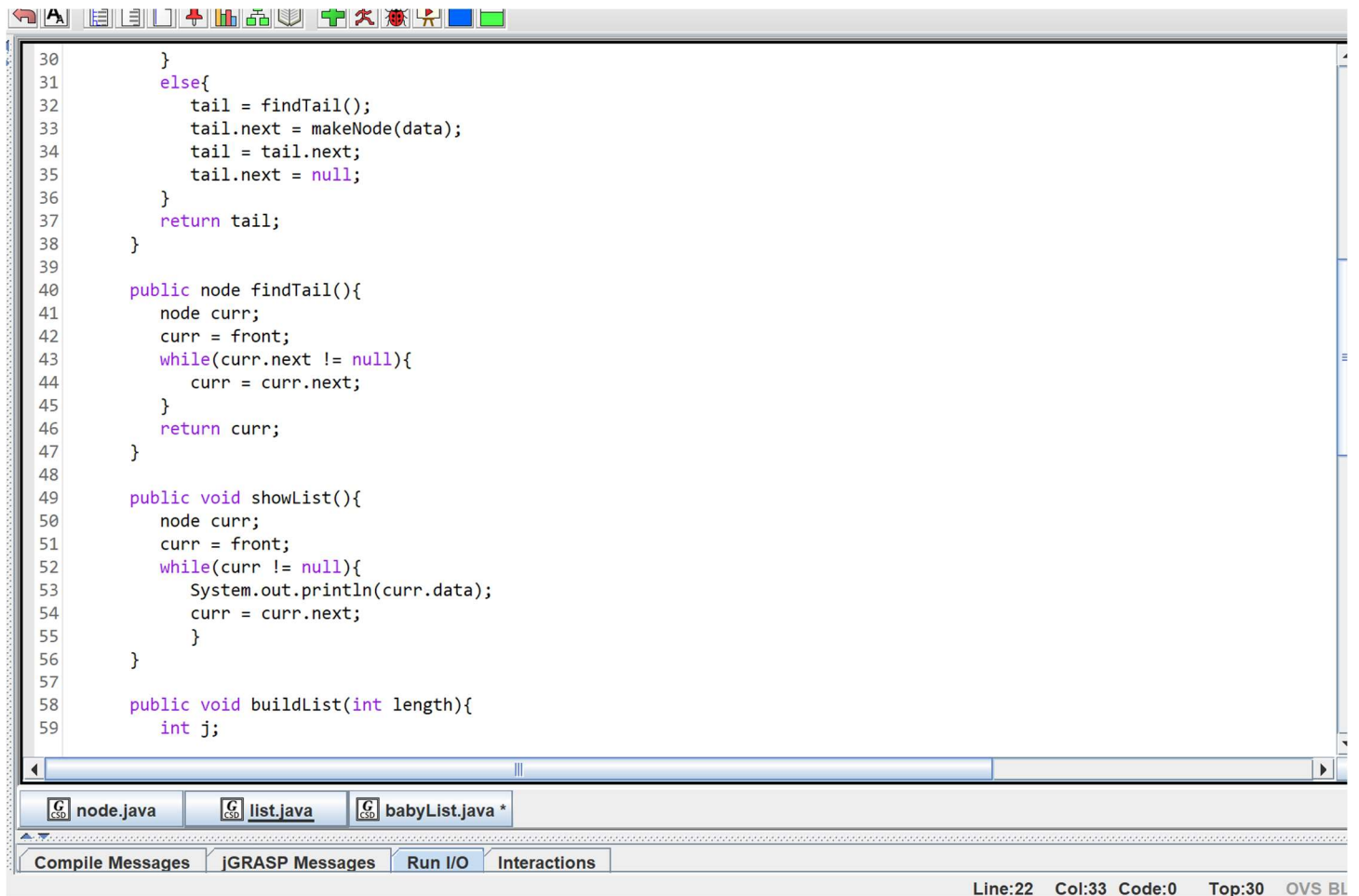
## List Class Screenshots



The screenshot shows a Java IDE with a toolbar at the top and a code editor. The code editor displays the implementation of a `List` class. The code is as follows:

```
1 public class list{  
2     int data;  
3     node next;  
4     node curr;  
5     node front;  
6     node tail;  
7     node newNode;  
8  
9     public void init(){  
10         front = null;  
11     }  
12  
13     public node makeNode(int data){  
14         newNode = new node();  
15         newNode.data = data;  
16         newNode.next = null;  
17  
18         return newNode;  
19     }  
20  
21     public node addFirst(int data){  
22         front = makeNode(data);  
23         tail = front;  
24         return front;  
25     }  
26  
27     public node addLast(int data){  
28         if(front == null){  
29             front = makeNode(data);  
30         }  
31     }  
32 }
```

The IDE's tab bar at the bottom shows three open files: `node.java`, `list.java` (which is the active file), and `babyList.java *`.

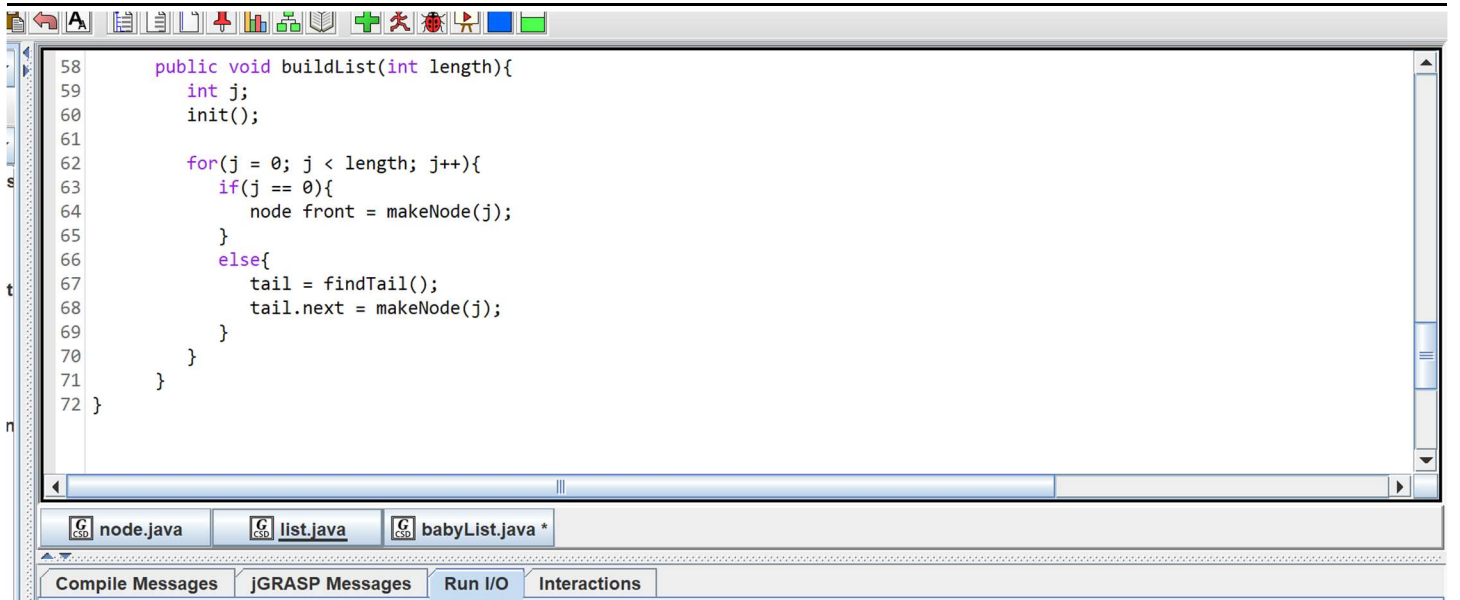


```
30     }
31     else{
32         tail = findTail();
33         tail.next = makeNode(data);
34         tail = tail.next;
35         tail.next = null;
36     }
37     return tail;
38 }
39
40 public node findTail(){
41     node curr;
42     curr = front;
43     while(curr.next != null){
44         curr = curr.next;
45     }
46     return curr;
47 }
48
49 public void showList(){
50     node curr;
51     curr = front;
52     while(curr != null){
53         System.out.println(curr.data);
54         curr = curr.next;
55     }
56 }
57
58 public void buildList(int length){
59     int j;
```

node.java list.java babyList.java \*

Compile Messages JGRASP Messages Run I/O Interactions

Line:22 Col:33 Code:0 Top:30 OVS BL



```
58     public void buildList(int length){
59         int j;
60         init();
61
62         for(j = 0; j < length; j++){
63             if(j == 0){
64                 node front = makeNode(j);
65             }
66             else{
67                 tail = findTail();
68                 tail.next = makeNode(j);
69             }
70         }
71     }
72 }
```

node.java list.java babyList.java \*

Compile Messages JGRASP Messages Run I/O Interactions

## Code Running

The screenshot shows a Java IDE with a code editor at the top and a console at the bottom. The code editor contains the following Java code:

```
19 Scanner scanner = new Scanner(System.in);
20
21 System.out.println("Enter the number of integers in your list. ");
22 int size = scanner.nextInt();
23
24 list listB = new list();
25 listB.init();
26
```

Below the code editor, there are tabs for `node.java`, `list.java`, and `babyList.java`. The `Run I/O` tab is selected, showing the execution output:

```
----jGRASP exec: java babyList
Contents of linked list A:
0
1
2
3
4
5
6
7
8
9
Enter the number of integers in your list.
>> L
```

On the left side of the console, there are buttons for `End`, `Clear`, and `Help`, along with a small icon of a document with a magnifying glass.

At the bottom of the IDE, the status bar shows: `Line:5 Col:30 Code:42 Top:19`.

```
19 Scanner scanner = new Scanner(System.in);
20
21 System.out.println("Enter the number of integers in your list. ");
22 int size = scanner.nextInt();
23
24 list listB = new list();
25 listB.init();
26
```

node.java list.java babyList.java

Compile Messages JGRASP Messages Run I/O Interactions

End  
Clear  
Help  
A

```
4
5
6
7
8
9
Enter the number of integers in your list.
>> 5
>> What is the first number?17
```

user program Line:5 Col:30 Code:42 Top:19 OVS RI K

```
19 Scanner scanner = new Scanner(System.in);
20
21 System.out.println("Enter the number of integers in your list. ");
22 int size = scanner.nextInt();
23
24 list listB = new list();
25 listB.init();
26
```

node.java list.java babyList.java

Compile Messages JGRASP Messages Run I/O Interactions

End  
Clear  
Help  
A

```
4
5
6
7
8
9
Enter the number of integers in your list.
>> 5
>> What is the first number?17
>> What is the next number?27
>> What is the next number?14
>> What is the next number?78
>> What is the next number?507
```

ing user program Line:5 Col:30 Code:42 Top:19 OVS I



```
33         System.out.print("What is the next number?");
34         int k = scanner.nextInt();
35         listB.addLast(k);
36     }
37 }
38 System.out.println("The contents of your list is:");
39 listB.showList();
40 }
```

node.java

list.java

babyList.java \*

Compile Messages

jGRASP Messages

Run I/O

Interactions

End

Clear

Help

A

8  
9

Enter the number of integers in your list.

5  
What is the first number?17  
What is the next number?27  
What is the next number?14  
What is the next number?78  
What is the next number?507

The contents of your list is:

17  
27  
14  
78  
507

----jGRASP: operation complete.

Line:38 Col:33 Code:84 Top:33 OVS BLK