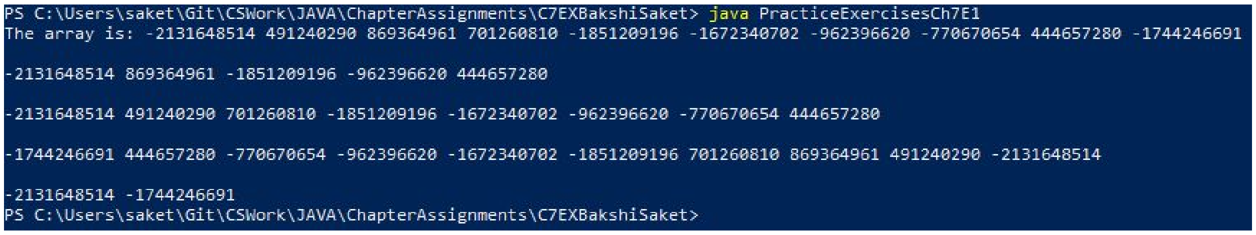
```java
/* Saket Bakshi 12/10/18. Period 6
This program, for #1 of Ch 7, initializes an array with 10 random integers and prints one line for
each of the following:
Every element at an even index
Every even element
All elements in reverse order
The first and last element
*/
import java.util.Random;

public class PracticeExercisesCh7E1
{
	public static void main(String[] args)
	{
		Random r = new Random(); //makes Random class object

		int[] array = new int[10]; //makes int type array
		for(int i = 0; i < 10; i++) //puts random int in each array element
		{
			array[i] = r.nextInt();
		}

		System.out.print("The array is: "); //prints the array
		for(int i = 0; i < 10; i++)
		{
			System.out.print(array[i] + " ");
		}

		System.out.println("");
		System.out.println("");
		for(int i = 0; i < 5; i++) //prints the even elements
		{
			System.out.print(array[2*i] + " ");
		}

		System.out.println("");
		System.out.println("");
		for(int i = 0; i < 10; i++)
		{
			if(array[i] % 2 == 0) //checks if element is even
			{
				System.out.print(array[i] + " "); //prints if even
			}
		}

		System.out.println("");
		System.out.println("");
		for(int i = 9; i >= 0; i--) //loop in array's reverse order
		{
			System.out.print(array[i] + " "); //prints element
		}

		System.out.println("");
		System.out.println("");
		System.out.print(array[0] + " " + array[9]); //prints first and last element
	}
}
```



```
PS C:\Users\saket\Git\CSWork\JAVA\ChapterAssignments\C7EXBakshiSaket> java PracticeExercisesCh7E1
The array is: -2131648514 491240290 869364961 701260810 -1851209196 -1672340702 -962396620 -770670654 444657280 -1744246691

-2131648514 869364961 -1851209196 -962396620 444657280

-2131648514 491240290 701260810 -1851209196 -1672340702 -962396620 -770670654 444657280

-1744246691 444657280 -770670654 -962396620 -1672340702 -1851209196 701260810 869364961 491240290 -2131648514

-2131648514 -1744246691
PS C:\Users\saket\Git\CSWork\JAVA\ChapterAssignments\C7EXBakshiSaket>
```

```
/* Saket Bakshi 12/10/18. Period 6
This program, for #2 of Ch 7, completes tasks for an array of integers
*/
import java.util.Arrays;

public class PracticeExercisesCh7E2
{
	private int[] values; //original array
	private int[] modifiedValues; //modified array
	private int currentSize; //length of original array

	/** This class completes tasks for integer arrays
	@param initialValues the initial array
	*/
	public PracticeExercisesCh7E2(int[] initialValues)
	{
		this.values = Arrays.copyOf(initialValues, initialValues.length); //copies array, not
the reference
		this.modifiedValues = Arrays.copyOf(initialValues, initialValues.length);
		this.currentSize = initialValues.length;
	}

	/** Prints the original array
	*/
	public void getOriginal()
	{
		for(int i = 0; i < this.currentSize; i++)
			System.out.print(this.values[i] + " ");
	}

	/** Prints the modified array
	*/
	public void getModded()
	{
		for(int i = 0; i < this.currentSize; i++)
			System.out.print(this.modifiedValues[i] + " ");
	}

	/** swaps first and last element in array
	*/
	public void partA()
	{
		int temp = this.values[0];
		this.modifiedValues[0] = this.modifiedValues[this.currentSize - 1];
		this.modifiedValues[this.currentSize - 1] = temp;
	}

	/** shifts elements to the right, last element becomes first
	*/
	public void partB()
	{
		int temp = this.values[this.currentSize - 1]; //creates temp value for last element
		for(int i = this.currentSize - 1; i > 0; i--) //shifts array
		{
			this.modifiedValues[i] = this.modifiedValues[i-1];
		}
		this.modifiedValues[0] = temp;
	}

	/** replaces even elements with 0
	*/
	public void partC()
	{
		for(int i = 0; i < this.modifiedValues.length-1; i = i + 2) //goes through even
indexes of the array
		{
```

```
            this.modifiedValues[i] = 0;
        }
    }

    /** replaces each element except first and last with the larger of its two neighbors
    */
    public void partD()
    {
        for(int i = 1; i < this.currentSize - 1; i++)
        {
            if(this.values[i-1] < this.values[i+1]) //checks which neighbor is larger
                this.modifiedValues[i] = this.values[i+1]; //assigns the larger
neighbor to the current index
            else if(this.values[i-1] > this.values[i+1])
                this.modifiedValues[i] = this.values[i-1];
            else if(this.values[i-1] == this.values[i+1])
                this.modifiedValues[i] = this.values[i];
        }
    }

    /** Removes the middle element if the array length is odd, or the middle two if length is
even
    */
    public void partE()
    {
        if(this.currentSize % 2 == 0) //if array is even in length
        {
            for(int i = (this.currentSize/2) - 1; i < this.currentSize - 2; i++) //shifts the
array down two starting from the middle
            {
                this.modifiedValues[i] = this.modifiedValues[i+2];
            }

            for(int i = 0; i < this.currentSize - 2; i++)
            {
                System.out.print(this.modifiedValues[i] + " "); //prints all but last 2
elements
            }
        }
        else //if array length is odd
        {
            for(int i = (this.currentSize - 1) / 2; i < this.currentSize - 2; i++) //shift array
down one starting from middle
```

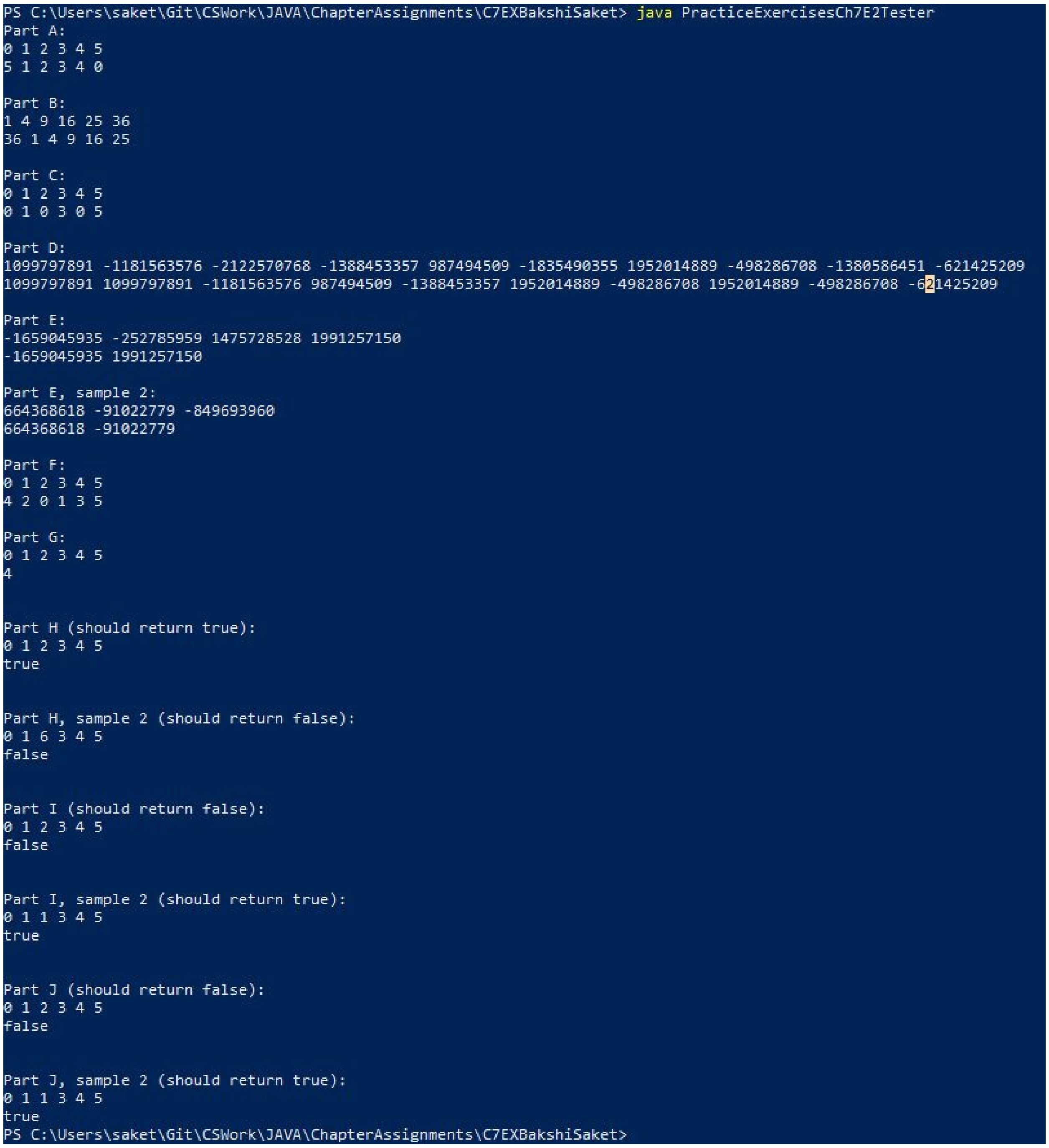
```
				{
					this.modifiedValues[i] = this.modifiedValues[i+1];
				}

				for(int i = 0; i < this.currentSize - 1; i++)
				{
					System.out.print(this.modifiedValues[i] + " "); //prints all but last
element
				}
			}
		}

		/** moves even elements to front, preserves rest of order of the array
		*/
		public void partF()
		{
			for(int i = 0; i < this.currentSize; i++) //goes through each element of the array
			{
				if(this.values[i] % 2 == 0) //if element is even
				{
					int temp = this.values[i]; //a temp variable takes the element's
value
					for(int j = i; j >= 1; j--) //the array is shifted down by 1 from right to
left
					{
						this.modifiedValues[j] = this.modifiedValues[j-1];
					}
					this.modifiedValues[0] = temp; //first element is set to the temp
integer
				}
			}
		}

		/** prints the second largest element
		*/
		public void partG()
		{
			int largest = this.values[0]; //sets largest and second largest to 0
			int secondLargest = this.values[0];
			for(int i = 0; i < this.currentSize; i++)
			{
				if(this.values[i] > largest) //if current value is larger than anything before
				{
```

```
				secondLargest = largest; //the previous largest is set to second
largest
				largest = this.values[i]; //the new largest is set
			}
		}
		System.out.println(secondLargest);
	}

	/** returns true if array is in increasing order
	*/
	public void partH()
	{
		boolean order = false;
		for(int i = 0; i < this.currentSize - 2; i++) //checks each element until
second-to-last
		{
			if(this.values[i] < this.values[i+1]) //checks if next element is greater
				order = true; //maintains boolean as true
			else //or else...
			{
				order = false; //boolean is set to false
				i = currentSize; //loop ends
			}
		}
		if(order)
			System.out.println("true");
		else
			System.out.println("false");
	}

	/** checks if two adjacent elements are duplicates
	*/
	public void partI()
	{
		boolean order = false;
		for(int i = 0; i < this.currentSize - 2; i++)
		{
			if(this.values[i] == this.values[i+1]) //checks is elements are identical
			{
				order = true; //sets boolean to true
				i = currentSize; //ends loop
			}
		}
		if(order)
			System.out.println("true");
		else
			System.out.println("false");
	}

	/** checks if there are duplicate elements
	*/
	public void partJ()
	{
		boolean duplicate = false;
		for(int i = 0; i < this.currentSize - 1; i++) //goes through each number
		{
			for(int j = i + 1; j < this.currentSize - 1; j++) //now goes through rest of
numbers
			{
				if(this.values[j]==this.values[i]) //if first number is equal to second...
					duplicate = true; //boolean set to true
					j = this.currentSize; //inner loop ends
			}
			if(duplicate)
				i = this.currentSize; //outer loop ends
		}
		if(duplicate)
			System.out.println("true");
		else
			System.out.println("false");
	}
}
/* Saket Bakshi 12/10/18. Period 6
This program, for #2 of Ch 7, tests a class that completes tasks for an array of integers
*/
import java.util.Random;

public class PracticeExercisesCh7E2Tester
{
	public static void main(String[] args)
	{
		System.out.println("Part A:");
		int[] arrayA = { 0, 1, 2, 3, 4, 5 };
		PracticeExercisesCh7E2 a = new PracticeExercisesCh7E2(arrayA);
		a.getOriginal();
		System.out.println("");
```

```
a.partA();
a.getModded();
System.out.println("");
System.out.println("");

System.out.println("Part B:");
int[] arrayB = {1, 4, 9, 16, 25, 36};
PracticeExercisesCh7E2 b = new PracticeExercisesCh7E2(arrayB);
b.getOriginal();
System.out.println("");
b.partB();
b.getModded();
System.out.println("");
System.out.println("");

System.out.println("Part C:");
int[] arrayC = {0, 1, 2, 3, 4, 5};
PracticeExercisesCh7E2 c = new PracticeExercisesCh7E2(arrayC);
c.getOriginal();
System.out.println("");
c.partC();
c.getModded();
System.out.println("");
System.out.println("");

System.out.println("Part D:");
Random r = new Random();
int[] arrayD = new int[10];
for(int i = 0; i < 10; i++)
{
        arrayD[i] = r.nextInt();
}
PracticeExercisesCh7E2 d = new PracticeExercisesCh7E2(arrayD);
d.getOriginal();
System.out.println("");
d.partD();
d.getModded();
System.out.println("");
System.out.println("");

System.out.println("Part E:");
int even = 1;
do
```

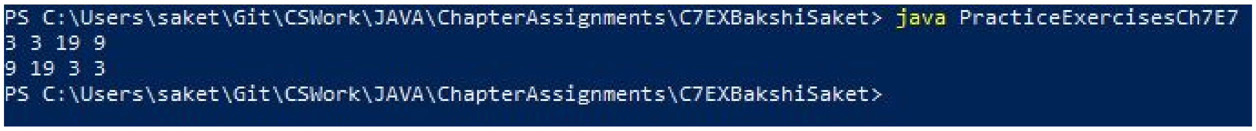
```
{
        even = r.nextInt(10);
} while(even % 2 != 0);
int[] arrayE = new int[even];
for(int i = 0; i < arrayE.length ; i++)
{
        arrayE[i] = r.nextInt();
}
PracticeExercisesCh7E2 e = new PracticeExercisesCh7E2(arrayE);
e.getOriginal();
System.out.println("");
e.partE();
System.out.println("");
System.out.println("");

System.out.println("Part E, sample 2:");
int odd = 0;
do
{
        odd = r.nextInt(10);
} while(odd % 2 != 1);
int[] arrayE2 = new int[odd];
for(int i = 0; i < arrayE2.length; i++)
{
        arrayE2[i] = r.nextInt();
}
PracticeExercisesCh7E2 e2 = new PracticeExercisesCh7E2(arrayE2);
e2.getOriginal();
System.out.println("");
e2.partE();
System.out.println("");
System.out.println("");

System.out.println("Part F:");
int[] arrayF = {0, 1, 2, 3, 4, 5};
PracticeExercisesCh7E2 f = new PracticeExercisesCh7E2(arrayF);
f.getOriginal();
System.out.println("");
f.partF();
f.getModded();
System.out.println("");
System.out.println("");
System.out.println("Part G:");
int[] arrayG = {0, 1, 2, 3, 4, 5};
PracticeExercisesCh7E2 g = new PracticeExercisesCh7E2(arrayG);
g.getOriginal();
System.out.println("");
g.partG();
System.out.println("");
System.out.println("");

System.out.println("Part H (should return true):");
int[] arrayH = {0, 1, 2, 3, 4, 5};
PracticeExercisesCh7E2 h = new PracticeExercisesCh7E2(arrayH);
h.getOriginal();
System.out.println("");
h.partH();
System.out.println("");
System.out.println("");

System.out.println("Part H, sample 2 (should return false):");
int[] arrayH2 = {0, 1, 6, 3, 4, 5};
PracticeExercisesCh7E2 h2 = new PracticeExercisesCh7E2(arrayH2);
h2.getOriginal();
System.out.println("");
h2.partH();
System.out.println("");
System.out.println("");

System.out.println("Part I (should return false):");
int[] arrayI = {0, 1, 2, 3, 4, 5};
PracticeExercisesCh7E2 i = new PracticeExercisesCh7E2(arrayI);
i.getOriginal();
System.out.println("");
i.partI();
System.out.println("");
System.out.println("");

System.out.println("Part I, sample 2 (should return true):");
int[] arrayI2 = {0, 1, 1, 3, 4, 5};
PracticeExercisesCh7E2 i2 = new PracticeExercisesCh7E2(arrayI2);
i2.getOriginal();
System.out.println("");
i2.partI();
System.out.println("");
```

```
        System.out.println("");

        System.out.println("Part J (should return false):");
        int[] arrayJ = {0, 1, 2, 3, 4, 5};
        PracticeExercisesCh7E2 j = new PracticeExercisesCh7E2(arrayJ);
        j.getOriginal();
        System.out.println("");
        j.partJ();
        System.out.println("");
        System.out.println("");

        System.out.println("Part J, sample 2 (should return true):");
        int[] arrayJ2 = {0, 1, 1, 3, 4, 5};
        PracticeExercisesCh7E2 j2 = new PracticeExercisesCh7E2(arrayJ2);
        j2.getOriginal();
        System.out.println("");
        j2.partJ();
    }
}
```



```
PS C:\Users\saket\Git\CSWork\JAVA\ChapterAssignments\C7EXBakshiSaket> java PracticeExercisesCh7E2Tester
Part A:
0 1 2 3 4 5
5 1 2 3 4 0

Part B:
1 4 9 16 25 36
36 1 4 9 16 25

Part C:
0 1 2 3 4 5
0 1 0 3 0 5

Part D:
1099797891 -1181563576 -2122570768 -1388453357 987494509 -1835490355 1952014889 -498286708 -1380586451 -621425209
1099797891 1099797891 -1181563576 987494509 -1388453357 1952014889 -498286708 1952014889 -498286708 -621425209

Part E:
-1659045935 -252785959 1475728528 1991257150
-1659045935 1991257150

Part E, sample 2:
664368618 -91022779 -849693960
664368618 -91022779

Part F:
0 1 2 3 4 5
4 2 0 1 3 5

Part G:
0 1 2 3 4 5
4


Part H (should return true):
0 1 2 3 4 5
true


Part H, sample 2 (should return false):
0 1 6 3 4 5
false


Part I (should return false):
0 1 2 3 4 5
false


Part I, sample 2 (should return true):
0 1 1 3 4 5
true


Part J (should return false):
0 1 2 3 4 5
false


Part J, sample 2 (should return true):
0 1 1 3 4 5
true
PS C:\Users\saket\Git\CSWork\JAVA\ChapterAssignments\C7EXBakshiSaket>
```

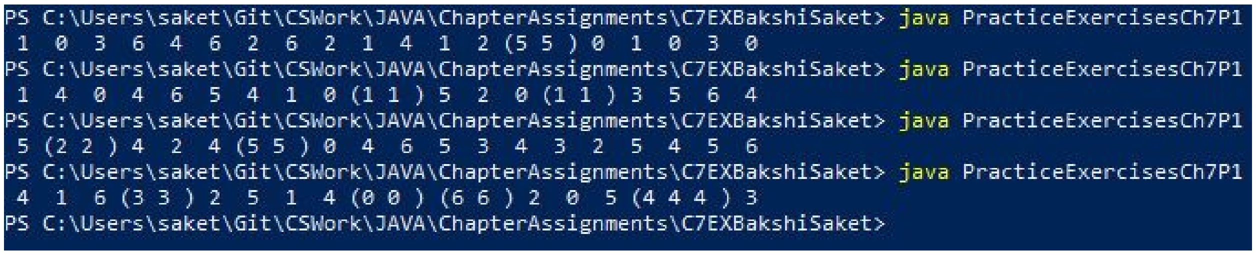
```
/* Saket Bakshi 12/10/18. Period 6
This program, for #7 of Ch 7, reverses the order of an array
*/
import java.util.Random;
import java.util.Arrays;

public class PracticeExercisesCh7E7
{
    public static void main(String[] args)
    {
        Random r = new Random();

        int lengthArray = 0;
        do
        {
            lengthArray = r.nextInt(20);
        } while(lengthArray % 2 != 0 && lengthArray > 0); //gets an even number greater
than 0

        int[] original = new int[lengthArray]; //constructs array
        for(int i = 0; i < original.length; i++) //fills array with random integers from 0 to 19,
inclusive
            original[i] = r.nextInt(20);

        for(int i = 0; i < original.length; i++) //prints original array values
            System.out.print(original[i] + " ");
        System.out.println();

        int[] temp = Arrays.copyOf(original, original.length); //makes a copy of the original
array

        int i = 0; //sets first index
        int j = original.length-1; //sets last index
        while(i < original.length/2) //takes first index to midpoint
        {
            original[i] = temp[j]; //sets lower index value to higher index value
            original[j] = temp[i]; //vice versa
            i++; //increases lower index
            j--; //decreases higher index
        }

        for(int k = 0; k < original.length; k++) //prints new array
            System.out.print(original[k] + " ");
    }
}
```



```
PS C:\Users\saket\Git\CSWork\JAVA\ChapterAssignments\C7EXBakshiSaket> java PracticeExercisesCh7E7
3 3 19 9
9 19 3 3
PS C:\Users\saket\Git\CSWork\JAVA\ChapterAssignments\C7EXBakshiSaket>
```

/* Saket Bakshi 12/10/18. Period 6

```
This program, for project 1 of Ch 7, simulates 20 dice rolls, prints them in order, and markes
runs (sequences of adjacent repeated values) with parentheses.
*/
import java.util.Random;

public class PracticeExercisesCh7P1
{
	public static void main(String[] args)
	{
    Random r = new Random();

    int[] diceRoll = new int[20]; //makes 20 length array
    for (int i = 0; i < diceRoll.length; i++) //fills in random rolls
    {
      diceRoll[i] = r.nextInt(7);
    }


    runs(diceRoll); //goes through method
  }

  /** checks array for runs
	*/
  public static void runs(int[] diceRoll)
  {
    boolean inRun = false; //sets boolean to not in a run
    int previousValue = diceRoll[0]; //sets temporary index to the 0th index

    for (int i = 0; i < diceRoll.length - 1; i++) //for each element in the array
    {
      if (inRun) //if already in a run
      {
        if (diceRoll[i] != previousValue) //if the new value doesn't continue the run
        {
          System.out.print(") "); //end the run
          inRun = false; //set run to false
          if(diceRoll[i] == diceRoll[i+1]) //if next number continues a new streak
          {
            System.out.print("("); //start a new run
            inRun = true; //set run to true
          }
        }
      }
```

```
            else if (!inRun) //if not in a run
            {
                if (diceRoll[i] == diceRoll[i + 1]) //if a streak will start
                {
                    System.out.print("("); //start a run
                    inRun = true; //set the boolean
                }
                else //if no streak
                {
                    System.out.print(" "); //add a space
                }
            }

            previousValue = diceRoll[i]; //sets current roll to previous roll for next round of the loop
            System.out.print(diceRoll[i] + " "); //prints out the current roll value
        }

        if(inRun && diceRoll[diceRoll.length - 1] == previousValue) //for last number, if there is a
run and last number continues it
        {
            System.out.print(" " + diceRoll[diceRoll.length - 1] + ") "); //put number then end run
        }
        else if(inRun && diceRoll[diceRoll.length - 1] != previousValue) //if run and not continued
        {
            System.out.print(") " + diceRoll[diceRoll.length - 1]); //end run then put number
        }
        else //if no run
        {
            System.out.print(" " + diceRoll[diceRoll.length - 1]); //print number
        }
    }
}
```


```
PS C:\Users\saket\Git\CSWork\JAVA\ChapterAssignments\C7EXBakshiSaket> java PracticeExercisesCh7P1
 1  0  3  6  4  6  2  6  2  1  4  1  2 (5 5 ) 0  1  0  3  0
PS C:\Users\saket\Git\CSWork\JAVA\ChapterAssignments\C7EXBakshiSaket> java PracticeExercisesCh7P1
 1  4  0  4  6  5  4  1  0 (1 1 ) 5  2  0 (1 1 ) 3  5  6  4
PS C:\Users\saket\Git\CSWork\JAVA\ChapterAssignments\C7EXBakshiSaket> java PracticeExercisesCh7P1
 5 (2 2 ) 4  2  4 (5 5 ) 0  4  6  5  3  4  3  2  5  4  5  6
PS C:\Users\saket\Git\CSWork\JAVA\ChapterAssignments\C7EXBakshiSaket> java PracticeExercisesCh7P1
 4  1  6 (3 3 ) 2  5  1  4 (0 0 ) (6 6 ) 2  0  5 (4 4 4 ) 3
PS C:\Users\saket\Git\CSWork\JAVA\ChapterAssignments\C7EXBakshiSaket>
```