```
public class JasonFanPset2 {
	public static void main(String[] args) {
	}
	//1
	public static String minCat(String s1, String s2) {
		int minLen = Math.min(s1.length(), s2.length());
		return s1.substring(s1.length() - minLen) + s2.substring(s2.length() -
minLen);
	}

	//2
	public static int countJava(String s) {
		int count = 0;
		for (int i = 0; i < s.length() - 4; i++) {
			if ((s.substring(i, i + 2).equals("ja") && s.substring(i + 3, i +
4).equals("a"))) {
				count++;
			}
		}
		return count;
	}
	//3
	public static boolean isPalindrome(String s) {
		String rev = "";
		for (int i = s.length() - 1; i > -1; i--) {
			rev += s.substring(i, i + 1);
		}

		return rev.equals(s);
	}
	//4
	public static int sumString(String s) {
		int num = 0;
		String curr = "";

		for (int i = 0; i < s.length(); i++) {
		char c = s.charAt(i);
			if (Character.isDigit(c)) {
				curr += c;
			}
			else {
				if (!curr.equals("")) {
				num += Integer.valueOf(curr);
				curr = "";
				}
			}
		}
		if (!curr.equals("")) {
			num += Integer.valueOf(curr);
		}
		return num;

	}
	//5
	public static boolean sameStarChars(String s) {
		for (int i = 1; i < s.length() - 1; i++) {
			if ((s.charAt(i) == '*') && !(s.charAt(i - 1) == s.charAt(i +
1))) {
```

```
                return false;
            }
        }
        return true;
    }

    //6
    public static boolean areAnagrams(String s1, String s2) {

        if (s1.length() != s2.length()) {
            return false;
        }

        StringBuilder sb2 = new StringBuilder(s2);

        for (int i = 0; i < s1.length(); i++) {
            int ind = sb2.indexOf(String.valueOf(s1.charAt(i)));
            if (ind == -1) {
                return false;
            }
            else {
                sb2.deleteCharAt(ind);
            }
        }

        return sb2.length() == 0;
    }
    //7
    public static int longSubStringLen(String s) {
        int l = 0;
        String curr = "";
        for (int i = 0; i < s.length(); i++) {
            boolean isIn = false;
            for (int j = 0; j < curr.length(); j++) {
                if (curr.indexOf(s.charAt(i)) != -1) {
                    isIn = true;
                }
            }
            if (isIn) {
                curr = "";
            }
            else {
                curr += s.charAt(i);
            }

            if (curr.length() > l) {
                l = curr.length();
            }
        }
        if (curr.length() > l) {
            l = curr.length();
        }
        return l;
    }

    //8 absolute genius btw
    public static boolean areRotations(String s1, String s2) {
        String con = s2 + s2;
        return con.contains(s1);
```

```
}

//9
public static String longestWord(String s) {
	String l = "";
	String curr = "";
	for (int i = 0; i < s.length(); i++) {
		if (s.charAt(i) != ' ') curr += s.charAt(i);
		else if (curr.length() >= l.length()) {
			l = curr;
			curr = "";
		}
	}
	if (curr.length() >= l.length()) {
		l = curr;
	}
	return l;
}
//10
public static int isHarshad(int n) {
	int count = 0;
	for (int i = 1; i <= n; i++) {
		int s = 0;
		int cn = i;
		while (cn > 0) {
			int digit = cn % 10;
			s += digit;
			cn /= 10;
		}
		if (i % s == 0) {
			count += 1;
		}
	}
	return count;
}

//11
public static double sqrt(double x) {
	double t = x;
	while (Math.abs(t * t - x) >= Math.pow(10, -15)) {
		t = (t + (x / t)) / 2;
	}
	return t;
}

//12
public static double harmonicSum(int n) {
	double s = 0.0;

	for (int k = 1; k <= n; k++) {
		s += 1.0 / k;
	}

	return s;
}

//13
public static double estimatePi(int n) {
	double s = 0.0;
```

```
	for (int k = 1; k <= n; k++) {
		s += 1.0 / (k * k);
	}

	return Math.sqrt(6 * s);
}

//14
public static void makeSquares(int n) {
	for (int row = 0; row < n; row++) {

		for (int col=0; col < n; col++) {
			System.out.print("*");
		}

		System.out.print(" ");

		for (int col=0; col < n; col++) {
			if (row == 0 || row == n-1 || col==0 || col == n -1) {
				System.out.print("*");
			}
			else {
				System.out.print(" ");
			}
		}

		System.out.println();
	}
}
//15
public static void makePyramid(int n) {
	for (int row = 1; row <= n; row++) {
		int stars = 2 * row - 1;

		for (int j = 0; j < n - row; j++) {
			System.out.print(" ");
		}

		for (int k = 0; k < stars; k++) {
			System.out.print("*");
		}

		System.out.println();

	}
}
//16
public static void makeDiamond(int n) {
	for (int row = 1; row < n * 2; row++) {
		int stars;
		if (row <= n) {
			stars = row;
		}
		else {
			stars = n * 2 - 1 - row + 1;
		}

		int space = n - stars;
```

```
            for (int j = 0; j < space; j++) {
                  System.out.print(" ");
            }

            for (int k = 1; k <= stars * 2 - 1; k++) {
                  if (k == 1 || k == stars * 2 - 1) System.out.print("*");
                  else System.out.print(" ");
            }
            System.out.println();
      }

}
//17
private static int randInt(int min, int max) {
      return (int) (Math.random() * (max - min)) + min;
}

public static int nEncounters() {
      final int pokemon = 150;
      int caught = 0;
      int encounters = 0;
      while (caught < pokemon) {
            encounters++;

            int rand = randInt(0, pokemon);

            if (rand < pokemon - caught) {
                  caught++;
            }
      }
      return encounters;
}
//18
public static int sameBirthday() {
      final int days = 365;
      int people = 0;
      int daysLeft = days;

      while (true) {
            int chance = randInt(1, daysLeft);
            if (chance <= people) return people;
            people++;
            daysLeft--;
      }
}

//19
public static double pHasDisease(double x, double g) {
      int successes = 0;
      int totalPositive = 0;
      for (int i = 0; i < 100000; i++) {
            boolean isPositive = Math.random() < x;
            boolean testPositive;

            if (isPositive) testPositive = Math.random() < g;
            else testPositive = Math.random() < (1 - g);

            if (testPositive) {
```

```
                        totalPositive++;
                        if (isPositive) {
                              successes++;
                        }
                  }

            }
            return (double) successes / totalPositive;
      }

      //20
      public static void getTraj(double h, double vi, double g, double theta,
double dt) {
            theta *= Math.PI / 180;
            double time = 0;
            double vy = vi * Math.sin(theta);
            double vx = vi * Math.cos(theta);
            while (true) {
                  double y = h + vy * time + (0.5 * g * time * time);
                  double x = vx * time;
                  System.out.print(x + " " + y);
                  if (y < 0) break;
                  System.out.println();
                  time += dt;
            }
      }
}
```