

Practice Problems: Loop

1. Loop development (Source code java files are needed)

- Write a program that reads in an integer N from the keyboard, and displays a diamond shape on the screen with width $2N$ and height $2N$. For example, if $N=5$, it should display the following figure on the screen:

```
import java.util.Scanner;
public class Diamond {
    public static void main(String [] args) {
        Scanner kb = new Scanner(System.in);
        int N = kb.nextInt();

        // First part:
        // A loop that goes N times, to write the first N lines
        // Counter-controlled loop for each line?
        // Is body another loop?
            // Given the ith line, know how many spaces ( ` ` )
            //         before *, in the middle before the 2nd *?
            //         i.e., i from 0 to n-1, we need n-1-i and
            //         2*i here, respectively!
            // Each part of space display needs a loop.

        int n = keyboard.nextInt();
        int line;
        int space;

        for(line = 0; line < n; line++){
            for (space = 0; space< n-line-1; space++)
                System.out.print(" ");
            System.out.print("*");
            for (space = 0; space < 2*line; space ++ )
                System.out.print(" ");
            System.out.println("*");
        }

        // Second Part:
        // A loop that goes N times, to write the second N lines
        // This is basically a repeat of the loop above, except for the
        // change of counter control (values).

        for(line = 0; line < n; line++){
            for (space = 0; space< line; space++)
                System.out.print(" ");
            System.out.print("*");
            for (space = 0; space < 2*(n-line-1); space ++ )
                System.out.print(" ");
            System.out.println("*");
        }
    }
}
```

}

- b. Write a program that reads in an integer N from the keyboard, and displays whether N is a prime number or not. A number is "prime" if its only factors are 1 and itself. A "factor" is a number that divides another number evenly.

Hint: Event control loop, what condition to terminate? ... (Need to search for the next factor, until this factor reaches N ! Then what is the expression in loop? How to control the event/factor change?)

```
int n = keyboard.nextInt();
int factor=2;
boolean searchPrime = true;

while (factor < n && searchPrime){
    if (n%factor==0)
        searchPrime = false;
    else
        factor ++;
}
System.out.println(searchPrime);
```

Modify your program by adding a loop to find the first prime number larger than 1000.

Hint: event control until the prime number is found. Event change: Reuse the above check process. If the current number is prime, then the number is found. Otherwise, set the number for next round ++.

```
int n = 1001;
int factor;
boolean searchPrime = true;
boolean foundNumber = false;
while (!foundNumber){
    searchPrime = true;
    for (factor=2; factor<n && searchPrime; factor++){
        if (n%factor==0)
            searchPrime = false;
    }
    if(searchPrime)
        foundNumber = true;
    else
        n ++;
}
System.out.println("The next prime number after 1000 is "+n);
```

- c. Write a program that reads in an integer N from the keyboard, and displays whether N is a "perfect number" or not. A number is "perfect" if it is equal to the sum of all of its factors (not including itself as a factor, but including 1 as a factor). 6 is the first perfect number, because its factors are 1, 2, and 3, and $1+2+3 = 6$.

Hint: Counter control loop to add any possible factor to the sum (a check is needed to identify the required factor)!

```
int n = keyboard.nextInt();
int factor=2;
int total = 1;

for(; factor < n; factor ++){
    if (n%factor ==0)
        total += factor;
}
System.out.println(n == total);
```

Add a loop to your program to find the next perfect number after 6.

```
boolean foundPerfect = false;
int n = 7, total, factor;
while (!foundPerfect){
    total = 1;
    for(factor = 2; factor < n; factor ++){
        if(n%factor == 0)
            total += factor;
    }
    if (total == n)
        foundPerfect = true;
    else
        n++;
}
System.out.println("Next perfect number is "+n);
```

5. Practice – Write a simple program to simulate the dice game of “Craps”.

The program should roll two 6-sided dice and compute the sum. If the sum is 7, it should keep rolling until the sum is something different than a 7. That value is called the “point”.

```
Random rand = new Random();
int die1, die2;
do {
    die1 = rand.nextInt(6)+1;
    die2 = rand.nextInt(6)+1;
} while (die1+die2 == 7);
int point = die1+die2;

do {
    die1 = rand.nextInt(6)+1;
    die2 = rand.nextInt(6)+1;
} while (die1+die2 != 7 && die1+die2!=point);

if(die1+die2==7)
    System.out.println("You Lose!");
else
    System.out.println("You win!");
```