

☆ソースファイル

```
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.BufferedReader;

/**
 * 演習 7 : 自然対数の底 e の近似計算 (解答例).
 * @author 自分の学籍番号・氏名
 */
class Ex7 {
    public static void main(String[] args) {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        int n = 0;
        System.out.print("正の整数 n を入力: ");
        try {
            String s = br.readLine();
            n = Integer.parseInt(s);
        } catch (NumberFormatException e) {
            System.err.println("整数が入力されていません");
            System.exit(2);
        } catch (IOException e) {
            System.err.println("入力エラー");
            System.exit(1);
        }

        if (n <= 0) {
            System.err.println("n は正の整数でなければなりません");
            System.exit(3);
        }

        double sum = 1.0;
        long factor = 1;
        for (int i = 1; i <= n; i++) {
            factor *= i;
            sum += 1.0 / factor;
        }

        System.out.println();
        System.out.println("近似値 = " + sum);
        System.out.println("Math.E = " + Math.E);
        System.out.println("差 = " + (Math.E - sum));
    }
}
```

以下では、学籍番号を 16C2345 とする。

☆入力 1 のときのプログラム出力

正の整数 n を入力: 5

近似値 = 2.716666666666663

Math.E = 2.718281828459045

差 = 0.0016151617923787498

☆入力 2 のときのプログラム出力

正の整数 n を入力: 17

近似値 = 2.7182818284590455

Math.E = 2.718281828459045

差 = -4.440892098500626E-16

☆入力 2 のときの値が変化しなくなる理由の考察

double 型浮動小数点数の精度は 15~16 桁であり、コード部分の

```
sum += 1.0 / factor;
```

の部分で、sum の値が 2.7 付近の値であるのに対し、n が 17 の繰り返し最後の加算のとき  $1.0/\text{factor}$  の値が  $2.8\text{e-}15$  程度の値となり、精度の限界に達して「情報落ち」により加算の効果がなくなってしまうため。