slip17

```
import java.util.Arrays;
import java.util.Scanner;

public class s17q1 {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of inputs: ");
        int n = scanner.nextInt();

        int[] numbers = new int[n];
        int[] armstrongNumbers = new int[n];
        int armstrongIndex = 0;

        System.out.println("Enter the numbers:");
        for (int i = 0; i < n; i++) {
            numbers[i] = scanner.nextInt();
        }

        for (int number : numbers) {
            if (isArmstrong(number)) {
                armstrongNumbers[armstrongIndex++] = number;
            }
        }
```

```
        int[] trimmedArmstrongNumbers = Arrays.copyOf(armstrongNumbers,
armstrongIndex);

        System.out.println("Armstrong numbers:");
        for (int armstrongNumber : trimmedArmstrongNumbers) {
            System.out.println(armstrongNumber);
        }
    }

    public static boolean isArmstrong(int number) {
        int temp = number;
        int sum = 0;
        int digitCount = String.valueOf(number).length();

        while (temp > 0) {
            int digit = temp % 10;
            sum += (int) Math.pow(digit, digitCount);
            temp /= 10;
        }

        return sum == number;
    }
}
```

```
import java.util.Scanner;
```

```
class Product {
    int pid;
    String pname;
    double price;
    int qty;

    public void acceptDetails(Scanner scanner) {
        System.out.print("Enter Product ID: ");
        pid = scanner.nextInt();
        scanner.nextLine();

        System.out.print("Enter Product Name: ");
        pname = scanner.nextLine();

        System.out.print("Enter Product Price: ");
        price = scanner.nextDouble();

        System.out.print("Enter Product Quantity: ");
        qty = scanner.nextInt();
    }

    public void displayDetails() {
        System.out.println("Product ID: " + pid);
        System.out.println("Product Name: " + pname);
```

```
        System.out.println("Product Price: $" + price);
        System.out.println("Product Quantity: " + qty);
        System.out.println("Total Amount: $" + calculateTotalAmount());
    }

    public double calculateTotalAmount() {
        return price * qty;
    }
}

public class s {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);

        System.out.print("Enter the number of products: ");
        int n = scanner.nextInt();
        scanner.nextLine();

        Product[] p = new Product[n];

        for (int i = 0; i < n; i++) {
            System.out.println("\nEnter details for Product " + (i + 1) + ":");
            p[i] = new Product();
```

```
            p[i].acceptDetails(scanner);
        }


        System.out.println("\nProduct Details:");
        for (int i = 0; i < n; i++) {
            System.out.println("\nProduct " + (i + 1) + ":");
            p[i].displayDetails();
        }

        scanner.close();
    }
}
```

```
import tkinter as tk

class UpperCaseApp:
    def __init__(self, root):
        self.root = root
        self.root.title("Uppercase Converter")

        self.entry = tk.Entry(self.root, width=40)
        self.entry.pack(pady=10)

        self.convert_button = tk.Button(self.root, text="Convert to Uppercase",
command=self.convertuppercase)
```

```
        self.convert_button.pack(pady=5)

    def convertuppercase(self):
        input_text = self.entry.get()
        upper_text = input_text.upper()
        self.entry.delete(0, tk.END)
        self.entry.insert(0, upper_text)

root = tk.Tk()
app = UpperCaseApp(root)
root.mainloop()
```

```
class InvalidDateException(Exception):
    pass
class Date:
    def __init__(self, day, month, year):
        self.day = day
        self.month = month
        self.year = year
    def accept(self):
        try:
            self.day = int(input("Enter day: "))
            self.month = int(input("Enter month: "))
```

```
        self.year = int(input("Enter year: "))

        if not self.is_valid_date():
            raise InvalidDateException("Invalid Date Exception: The date entered is not valid.")
    except ValueError:
        raise InvalidDateException("Invalid Date Exception: Please enter numeric values for day, month, and year.")
  def is_valid_date(self):
    if self.month < 1 or self.month > 12:
        return False
    days_in_month = [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
    if self.month == 2:
        if (self.year % 4 == 0 and self.year % 100 != 0) or (self.year % 400 == 0):
            days_in_month[1] = 29
    if self.day < 1 or self.day > days_in_month[self.month - 1]:
        return False
    return True
  def display(self):
    print(f"Date: {self.day:02d}-{self.month:02d}-{self.year}")

date = Date(1, 1, 2000)
try:
  date.accept()
  date.display()
except InvalidDateException as e:
```

```
print(e)
```