**1a) Nth Prime number**

**import** java.util.Scanner;

**public** **class** AIsPrimeNumber {

**public** **static** **void** main(String args[]) {

Scanner sc= **new** Scanner(System.***in***);

**int** n = sc.nextInt();

sc.close();

*prime*(n);

}

**private** **static** **void** prime(**int** n) {

**boolean** isDivisible = **false**;

**if** (n == 0 || n == 1) {

System.***out***.println(n + " is not a prime number");

} **else** {

**for** (**int** i = 2; i <= n/2; i++) {

**if** (n % i == 0) {

isDivisible = **true**;

**break**;

}

}

**if** (isDivisible) {

System.***out***.println(n + " is not a prime number");

} **else** {

System.***out***.println(n + " is a prime number");

}

}

}

}

**1b) Upto n Prime numbers::**

**import** java.util.Scanner;

**public** **class** APrimeNumers {

**public** **static** **void** main(String args[]) {

Scanner sc= **new** Scanner(System.***in***);

**int** n = sc.nextInt();

sc.close();

**for**(**int** x=1;x<=n;x++) {

*prime*(x);

}

}

**private** **static** **void** prime(**int** n) {

**boolean** isDivisible = **false**;

**if** (n == 0 || n == 1) {

} **else** {

**for** (**int** i = 2; i <= n/2; i++) {

**if** (n % i == 0) {

isDivisible = **true**;

**break**;

}

}

**if** (!isDivisible) {

System.***out***.println(n + " is a prime number");

} } } }

**2. Factorial of a Number**

**package** com.numbers;

**public** **class** BFactorial {

**public** **static** **void** main(String[] args) {

**int** num = 5, fact = 1;

**for** (**int** i = num; i >= 1; i--) {

fact = fact\*i;

}

System.***out***.println("Factorial of " + num + " is: " + fact);

}

}

3. Fibbonacci 0 1 1 2 3 5 8 13

**public** **class** CFibonacci {

**public** **static** **void** main(String[] args) {

**int** n = 6;

CFibonacci.*printFibonacciSeries*(n);

//Upto 6: 0 1 1 2 3 5 (or) first n fibonacci numbers

CFibonacci.*nThFibonacciNumber*(n); //6th fibonacci number is 5

System.***out***.println(n + "th fibonacci number is " + CFibonacci.*nThFibonacciWithRecursion*(n)); //6th fibonacci number is 5

}

**private** **static** **void** printFibonacciSeries(**int** number) {

**int** n = number, t1 = 0, t2 = 1, fib = 0;

System.***out***.print("Upto " + n + ": ");

**for** (**int** i = 1; i <= n; i++) {

System.***out***.print(t1 + " ");

fib = t1 + t2;

t1 = t2;

t2 = fib;

}

System.***out***.println();

}

**public** **static** **int** nThFibonacciWithRecursion(**int** number) {

**int** n = number;

**if** (n == 1) {

**return** 0;

}

**if** (n == 2) {

**return** 1;

}

**return** *nThFibonacciWithRecursion*(n - 1) + *nThFibonacciWithRecursion*(n - 2);

}

**public** **static** **void** nThFibonacciNumber(**int** number) {

**int** n = number, t1 = 0, t2 = 1, fibo = 0;

**if** (n == 1) {

System.***out***.println(n + "th fibonacci number is " + 0);

**return**;

}

**if** (n == 2) {

System.***out***.println(n + "th fibonacci number is " + 1);

**return**;

}

**for** (**int** i = 3; i <= n; i++) {

fibo = t1 + t2;

t1 = t2;

t2 = fibo;

}

System.***out***.println(n + "th fibonacci number is " + fibo);

}

}

**4. Power of a number : 2^3 =2x2x2=8**

**package** com.numbers;

**import** java.util.Scanner;

**public** **class** DPowerOfNumber {

**public** **static** **void** main(String[] args) {

Scanner input = **new** Scanner(System.***in***);

System.***out***.print("Enter base: ");

**int** base = input.nextInt();

System.***out***.print("Enter exponent: ");

**int** exponent = input.nextInt();

**long** result = 1;

**for**(**int** i=1;i<=exponent;i++) {

result \*= base;

}

System.***out***.println("Result: " + result);

input.close();

}

}

**5. Reverse of Number ( 12345 ---🡪 54321)**

**package** com.numbers;

**public** **class** EReverseNumber {

**public** **static** **void** main(String[] args) {

**int** number = 12345, reverse = 0;

**while** (number != 0) {

**int** remainder = number % 10;

reverse = reverse \* 10 + remainder;

number = number / 10;

//reverse = reverse + remainder;-->sum of the digits of a number }

System.***out***.println("The reverse of the given number is: " + reverse);

}

}

**6. Palindrome :**

**package** com.numbers;

**public** **class** FPalindrome {

**public** **static** **void** main(String[] args) {

**int** number = 121, reverse = 0;

**int** givenNumber = number;

**while** (number != 0) {

**int** remainder = number % 10;

reverse = reverse \* 10 + remainder;

//reverse = reverse + remainder;-->sum of the digits of a number

number = number / 10;

}

System.***out***.println("The reverse of the given number is: " + reverse);

**if**(givenNumber == reverse) {

System.***out***.println(givenNumber +" number is a Palindrome");

} **else** {

System.***out***.println(givenNumber + " number is not a Palindrome");

}

}

}

**7. SumOfDigits :**

**package** com.numbers;

**public** **class** GSumOfDigits {

**public** **static** **void** main(String[] args) {

**int** number = 12345, sum = 0;

**while** (number != 0) {

**int** remainder = number % 10;

sum = sum + remainder;

//reverse = reverse + remainder;-->sum of the digits of a number

number = number / 10;

}

System.***out***.println("The sum of the digits is: " + sum); //15

}

}

**8. SwapNumbers**

**package** com.numbers;

**import** java.util.Scanner;

**class** HSwapNumbers {

//a=a+b;b=a-b;a=a-b;

/\*a=10 b=5

a=a+b=15

b=a-b=15-5=10

a=a-b=15-10=5\*/

**public** **static** **void** main(String args[]) {

**int** a, b, temp;

System.***out***.println("Enter a and b");

Scanner sc = **new** Scanner(System.***in***); // User inputs two numbers

a = sc.nextInt(); // User inputs two numbers

b = sc.nextInt();

System.***out***.println("Before Swapping\na = " + a + "\nb = " + b);

temp = a; // Swapping is done

a = b;

b = temp;

System.***out***.println("After Swapping\na = " + a + "\nb = " + b);

sc.close();

}

}

**9. ArmstrongNumber**

**package** com.numbers;

**import** java.util.Scanner;

**public** **class** IArmstrongNumber {

// 153: 1^3 + 5^3 + 3^3 = 1 + 125+ 27 = 153

// 1634: 1^4 + 6^4 + 3^4 + 4^4 = 1 + 1296 + 81 + 256 = 1643

**static** **boolean** isArmstrong(**int** givenNumber) {

**int** numberLength = *getLenthOfANumber*(givenNumber);

**int** digitsPowerSum = *digitPowerSum*(givenNumber, numberLength);

**if** (givenNumber == digitsPowerSum)

**return** **true**;

**else**

**return** **false**;

}

**private** **static** **int** digitPowerSum(**int** number, **int** numberLength) {

**int** sum = 0;

**while** (number > 0) {

**int** reminder = number % 10;

sum = sum + *pow*(reminder, numberLength);

number = number / 10;

}

**return** sum;

}

**private** **static** **int** getLenthOfANumber(**int** number) {

**int** count = 0;

**while** (number > 0) {

count++;

number = number / 10;

}

**return** count;

}

**private** **static** **int** pow(**int** base, **int** exponent) {

**int** result = 1;

**for** (**int** i = 1; i <= exponent; i++) {

result = result \* base;

}

**return** result;

}

**public** **static** **void** main(String args[]) {

**int** num;

Scanner sc = **new** Scanner(System.***in***);

System.***out***.print("Enter the limit: ");

num = sc.nextInt();

System.***out***.println("Armstrong Number up to " + num + " are: ");

**for** (**int** i = 0; i <= num; i++) {

**if** (*isArmstrong*(i)) {

System.***out***.print(i + ", ");

}

}

sc.close();

}

}

**10. ArmstsrongNumbersUpto:**

package com.numbers;

import java.util.Scanner;

import java.lang.Math;

public class IArmstsrongNumbersUpto {

// 153: 1^3 + 5^3 + 3^3 = 1 + 125+ 27 = 153

// 1634: 1^4 + 6^4 + 3^4 + 4^4 = 1 + 1296 + 81 + 256 = 1643

static boolean isArmstrong(int n) {

int temp, numberLength = 0, reminder = 0, sum = 0;

temp = n;

while (temp > 0) {

temp = temp / 10;

numberLength++;

}

temp = n;

while (temp > 0) {

reminder = temp % 10;

sum += (Math.pow(reminder, numberLength));

temp = temp / 10;

}

if (n == sum)

return true;

else

return false;

}

public static void main(String args[]) {

int num;

Scanner sc = new Scanner(System.in);

System.out.print("Enter the limit: ");

num = sc.nextInt();

System.out.println("Armstrong Number up to " + num + " are: ");

for (int i = 0; i <= num; i++)

if (isArmstrong(i)) {

System.out.print(i + ", ");

}

}

}

**11. LargestOf3 :**

**class** JLargestOf3 {

**public** **static** **void** main(String args[]) {

**int** a, b, c;

System.***err***.println("Enter three integers ");

Scanner sc = **new** Scanner(System.***in***);

a = sc.nextInt();

b = sc.nextInt(); // User Input

c = sc.nextInt();

**if** (a > b && a > c) // Condition check for Largest number

System.***out***.println("1st number is largest.");

**else** **if** (b > a && b > c)

System.***out***.println("2nd number is largest.");

**else** **if** (c > a && c > b)

System.***out***.println("3rd number is largest.");

**else**

System.***out***.println("Numbers are not distinct.");

sc.close();

}

}

**12. LargestAndSmallestInArray:**

**import** java.util.Arrays;

**public** **class** KLargestAndSmallestInArray {

**public** **static** **void** largestAndSmallest(**int**[] numbers) {

**int** largest = Integer.***MIN\_VALUE***;

**int** smallest = Integer.***MAX\_VALUE***;

**for** (**int** number : numbers) {

**if** (number > largest) {

largest = number;

}

**if** (number < smallest) {

smallest = number;

}

}

System.***out***.println("Given integer array : " + Arrays.*toString*(numbers));

System.***out***.println("Largest number in array is : " + largest);

System.***out***.println("Smallest number in array is : " + smallest);

}

**public** **static** **void** secondLargest(**int**[] numbers) {

**int** largest = Integer.***MIN\_VALUE***;

**int** scondLarge=Integer.***MIN\_VALUE***;

**for** (**int** number : numbers) {

**if** (number > largest) {

scondLarge=largest;

largest = number;

} **else** **if** (scondLarge < number) {

scondLarge = number;

}

}

System.***out***.println("Largest number in array is : " + largest);

System.***out***.println("Second Largest number in array is : " + scondLarge);

}

**public** **static** **void** secondMin(**int**[] numbers) {

**int** min = Integer.***MAX\_VALUE***;

**int** secondMin= Integer.***MAX\_VALUE***;

**for** (**int** number : numbers) {

**if** (number < min) {

secondMin=min;

min = number;

} **else** **if** (secondMin > number) {

secondMin = number;

}

}

System.***out***.println("Min number in array is : " + min);

System.***out***.println("Second Min number in array is : " + secondMin);

}

**public** **static** **void** main(String[] args) {

**int**[] a = { 1 ,2, 5, 7, 3 ,9 };

**int**[] a1 = **new** **int**[] { 1 ,2, 5, 7, 3 ,9 };

**int**[] a2 = **new** **int**[5];

*largestAndSmallest*(a);

//largestAndSmallest(new int[] { 10, Integer.MIN\_VALUE, -2 });

//largestAndSmallest(new int[] { Integer.MAX\_VALUE, 40, Integer.MAX\_VALUE });

//largestAndSmallest(new int[] { 1, -1, 0 });

*secondLargest*(**new** **int**[] { 1 ,2 ,5 ,7 ,3, 9});

*secondMin*(**new** **int**[] { 1 ,2 ,5, 7, 3, 9});

}

}

13. R**everseArray**

**package** com.numbers;

**import** java.util.Arrays;

**public** **class** LReverseArray {

**public** **static** **void** main(String[] args) {

**int**[] array = { 1, 2, 3, 4, 5 };

*reverseArray*(array);

System.***out***.println("Reversed array: " + Arrays.*toString*(array));

**int**[] array1 = *reverseArrayInNewArray*(array);

System.***out***.println("original array: " + Arrays.*toString*(array));

System.***out***.println("Reversed array: " + Arrays.*toString*(array1));

}

**public** **static** **void** reverseArray(**int**[] array) {

**int** start = 0;

**int** end = array.length - 1;

**while** (start < end) {

**int** temp = array[start];

array[start] = array[end];

array[end] = temp;

start++;

end--;

}

}

**public** **static** **int**[] reverseArrayInNewArray(**int**[] array) {

**int**[] n = **new** **int**[array.length];

**for**(**int** i = array.length - 1, j=0 ;i>=0;i--, j++) {

n[j]=array[i];

}

**return** n;

}

}

**14. RemoveDuplicatesOfSorted**

**package** com.numbers;

**public** **class** MRemoveDuplicatesOfSorted {

**public** **static** **int** removeduplicates(**int** a[], **int** length) {

**if** (length == 0 || length == 1) {

**return** length;

}

**int**[] temp = **new** **int**[length];

**int** j = 0;

**for** (**int** i = 0; i < length - 1; i++) {

**if** (a[i] != a[i + 1]) {

temp[j++] = a[i];

}

}

temp[j++] = a[length - 1];

// Changing the original array

**for** (**int** k = 0; k < j; k++) {

a[k] = temp[k];

}

**return** j;

}

**public** **static** **void** main(String[] args) {

**int** a[] = { 1, 1, 2, 2, 2, 4, 4 };

**int** length = a.length;

length = *removeduplicates*(a, length);

// Printing The array elements

**for** (**int** i = 0; i < length; i++) {

System.***out***.print(a[i] + " ");

}

}

}

**15. RepeatElement**

class MRepeatElement {

void printRepeatingElements(int arr[], int size) {

int count[] = new int[size];

int i;

System.out.print("Repeated elements are : ");

for (i = 0; i < size; i++) {

int element = arr[i];

if (count[element] == 1) {

System.out.print(element + " ");

}

count[element] = count[element] + 1;

}

System.out.println(" ");

}

void printRepeatingElementsMap(int arr[], int size) {

Map<Integer, Integer> m = new HashMap<>();

System.out.println("Repeated counts are : ");

for (int i = 0; i < size; i++) {

int element = arr[i];

Integer c = m.get(element);

if ( c== null) {

m.put(element, 1);

} else {

if (c == 1) {

System.out.println(element + " is duplicate");

}

m.put(element, c + 1);

}

}

}

void printRepeatingElementsCounts(int arr[], int size) {

int count[] = new int[size];

System.out.println("Repeated counts are : ");

for (int i = 0; i < size; i++) {

int element = arr[i];

count[element] = count[element] + 1;

}

for (int i = 0; i < size; i++) {

int element = arr[i];

// if (count[element] > 0) then gives each element count

if (count[element] > 1) {

System.out.println(element + " : " + count[element]);

count[element] = 0;

}

}

}

void printRepeatingElementsCountsMap(int arr[], int size) {

Map<Integer, Integer> m = new HashMap<>();

int i;

System.out.println("Repeated counts are : ");

for (i = 0; i < size; i++) {

int element = arr[i];

if (m.get(element) == null) {

m.put(element, 1);

} else {

m.put(element, m.get(element) + 1);

}

}

for (i = 0; i < size; i++) {

int element = arr[i];

if (m.get(element) > 1) {

System.out.println(element + " : " + m.get(element));

m.put(element, 0);

}

}

}

public static void main(String[] args) {

MRepeatElement repeat = new MRepeatElement();

int arr[] = { 4, 2, 4, 5, 4, 2, 1 };

int arr\_size = arr.length;

repeat.printRepeatingElements(arr, arr\_size);

repeat.printRepeatingElementsCounts(arr, arr\_size);

}

}