**III.THUẬT TOÁN**

# 1.Tìm số nguyên tố

public int sumPrime (int numberPrime) {

int count=1; int check = 0; int sumPrime = 0;

for (int i =2; count <=numberPrime; i++) {

check = 0;

for (int j = 1; j<=i; j++) {

if(i%j==0) {

check ++;}}

if (check == 2) {

sumPrime = sumPrime + i;

count ++;}}

return sumPrime;

# 2. Cho 1 chuỗi, lấy ra số lần xuất hiện của các ký tự có trong chuỗi.

Ví dụ: "aabacsdc": 3a,1b,2c,1d,1s?

String str = "aabacsdc"; String [] str1 = str.split ("");

HashMap<String, Integer> map = new HashMap<> ();

for (String key: str1) {if (! map.containsKey (key)) { map.put(key, 1); } else {

map.put (key, map.get(key) + 1);}}

Set<String> set = map.keySet ();

for (String key: set) {

System.out.println(map.get(key) + key);}

# 3.Thuật toán Dãy fibonacci nhỏ hơn 100?

public static void findFibonacci (int maxNumber) {

int fibo1 = 0; int fibo2 = 1; int fibonacci = 0;

while (fibonacci < maxNumber) {

fibonacci = fibo1; fibo1 = fibo2; fibo2 = fibonacci + fibo1;

if (fibonacci >= maxNumber) {break;}

System.out.println(fibonacci);}}

# 4.Tìm số nguyên tố?

public static boolean isPrime (int number) {

if (number < 2) {

return false;}

for (int i = 2; i < number; i++) {

if (number % i = = 0) {

return false;}}

return true;}

Scanner scanner = new Scanner (System.in);

System.out.print("Vui lòng nhập số: ");

int numberInput = scanner.nextInt ();

boolean check = isPrime(numberInput);

if(check) {System.out.println("Là số nguyên tố");

} else {System.out.println ("Không phải số nguyên tố");}

# 5.Đổi 1 số từ hệ thập phân sang hệ bát phân.?

Scanner scanner = new Scanner (System.in);

System.out.print("Nhập số thập phân: ");

int decimal = scanner.nextInt ();

String octal = "";

while (decimal > 0) {

octal = decimal % 8 + octal;

decimal = decimal / 8;}

System.out.println("Số bát phân: " + octal);

# 6. Cho 2 chuỗi "hh:MM:ss" viết hàm tính tổng 2 chuỗi đó. Vd: "11:20:30" và "07:35:45" => "18:56:15"

public static void sumHour (String hour1, String hour2) {

String [] h1 = hour1.split(":");

String [] h2 = hour2.split(":");

int s = Integer.parseInt(h1[2]) + Integer.parseInt(h2[2]);

int p = Integer.parseInt(h1[1]) + Integer.parseInt(h2[1]);

if (s >= 60) {s = s - 60; p = p + 1;}

int h = Integer.parseInt(h1[0]) + Integer.parseInt(h2[0]);

if (p >= 60) {p = p - 60; h = h + 1;}

System.out.println(h + ":" + p + ":" + s);}

# 7. Cho một mảng số nguyên và một số nguyên N bất kỳ, output: vị trí của 2 số bất kỳ trong mảng có tổng bằng N.

Int [] arr = {1,5,7,3,4,3,2,6,7,9,8,2,4,6};

int num = 10;

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

for (int j = i + 1; j < arr.length; j++)

{if (arr[i] + arr[j] == num) {

System.out.println(i + ""&"" + j + "" ("" + arr[i] + "" + "" + arr[j] + "" = "" + num + "")"");}}}

# 8. Cho một chuổi string nhập vào là các ký tự số la mã, convert thành số nguyên.

Sử dụng HashMap để ánh xạ các ký tự số La Mã thành giá trị số tương ứng

public class RomanToInteger {

public static int romanToInt (String s) {

if (s == null || s.length () == 0) { return 0; }

// Khởi tạo một HashMap để ánh xạ các ký tự số La Mã thành giá trị số tương ứng

HashMap<Character, Integer> map = new HashMap<Character, Integer>();

map.put('I', 1); map.put('V', 5); map.put('X', 10); map.put('L', 50); map.put('C', 100); map.put('D', 500); map.put('M', 1000);

int result = 0; int prevValue = 0;

// Duyệt chuỗi string từ phải sang trái và tính tổng các giá trị số tương ứng

for (int i = s.length() - 1; i >= 0; i--) {

int currValue = map.get(s.charAt(i));

if (currValue < prevValue) {result -= currValue;}

else {result += currValue;}

prevValue = currValue;}

return result;}

public static void main (String[] args) {

String roman = "MMXXI";

int integer = romanToInt(roman);

System.out.println(roman + " = " + integer);}} =>KQ: MMXXI = 2021;

# 9. Cho chuỗi string và sắp xếp chuỗi theo tăng dần.

public class SortString { public static void main(String[] args) {

String str = "Hello World";

char[] chars = str.toCharArray(); // chuyển chuỗi thành mảng ký tự

System.out.println(chars);

Arrays.sort(chars); // sắp xếp mảng ký tự theo thứ tự tăng dần

String sortedStr = new String(chars); // tạo chuỗi mới từ mảng ký tự đã sắp xếp

System.out.println(sortedStr); // in ra chuỗi đã sắp xếp } }

# 10. Cho hai mảng số nguyên. tìm những số trùng nhau. (Sử dụng Map)

public class NumberDuplicateByMap {

public static void main(String[] args) {

int[] arr1 = {1, 2, 3, 4, 4, 4, 5, 99};

int[] arr2 = {4, 4, 5, 5, 6, 6, 99, 7, 8};

Map<Integer, String> map = new HashMap<>();

Set<Integer> result = new HashSet<>();

for (int num : arr1) {

map.put(num, "Đây là số không trùng của mảng 1 được put vào map"); } }

System.out.println(map);

for (int num : arr2) {

if (map.containsKey(num)) {

map.remove(num);

result.add(num); }}

System.out.println("Result = " + result); } }

# 11. Thuật toán đảo chuỗi.

public class ReverseString {

public static String reverseString(String str) {

String result = "";

for (int i = str.length() - 1; i >= 0; i--) {

result += str.charAt(i); }

return result; }

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Nhập chuỗi cần đảo ngược: ");

String str = scanner.nextLine(); System.out.printf(reverseString(str)); } }

# 12.Vẽ tam giác cân?

public class PrintIsoscelesTriangle {

public static void main(String[] args) {

Scanner sc = new Scanner(System.in);

int hight, k = 0;

System.out.printf("Nhập vào chiều cao của kim tự tháp: ");

hight = sc.nextInt();

/\*- Vòng lặp for đầu tiên lặp từ 1 đến chiều cao hight của kim tự tháp, với chức năng để in xuống dòng các dấu \*.

- Trong vòng lặp đầu tiên này sẽ có một vòng lặp for và một vòng lặp while. \*/

for (int i = 1; i <= hight; ++i, k = 0) { /\*Vòng lặp for thứ hai lặp từ 1 đến chiều cao hight - 1, với chức năng in khoảng trắng.\*/

for (int space = 1; space <= hight - i; ++space) { System.out.print(" "); }

/\*Và vòng lặp while với điều kiện k != 2 \* i -1, để in dấu "\*" .\*/

while (k != 2 \* i - 1) { System.out.print("\* "); k++; } System.out.println(); } } }

# 13. Cho 1 mảng số. Tìm số lớn thứ 2 trong mảng ?

public class SecondLargest {

static void print2largest(int arr[], int size) {

int i;

if (size < 2) { System.out.print("Không tồn tại phần tử lớn thứ 2"); return; }

Arrays.sort(arr);

for (i = size - 2; i >= 0; i--) {

if (arr[i] != arr[size - 1]) { System.out.printf("Phần tử lớn thứ 2: " + arr[i]);

return; } }

System.out.print("Không tồn tại phần tử lớn thứ 2"); }

public static void main(String[] args) { int arr[] = {35, 35, 35, 10, 34, 35}; int n = arr.length; print2largest(arr, n); } }

# 14. Cho 1 số nhập từ bàn phím, in ra số ngược lại (chỉ xử lý số, không sử dụng chuỗi hoặc mảng ). VD: nhập vào 1523 kết quả là 3251.

public class ReverseNumberPrinting {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Vui lòng nhập số: ");

int number = scanner.nextInt();

int reverseNumber = 0;

while (number != 0) {

int temp = number % 10;

reverseNumber = reverseNumber \* 10 + temp; number = number / 10; }

System.out.print("Số ngược lại: " + reverseNumber); } }

# 15. Cho 5 số tự nhiên, tìm tổng 4 số lớn nhất và nhỏ nhất của mảng đó, dùng 1 vòng for.

public class Sum4MaxAndSum4Min {

public static void main(String[] args) {

int total = 0; int sumMax = 0; int sumMin = 0; int[] arrInt = {1, 5, 7, 10, 6};

Arrays.sort(arrInt);

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

total += arrInt[i];}

sumMax = total - arrInt[0];

sumMin = total - arrInt[4];

System.out.println("Tổng 4 số lớn nhất: " + sumMax);

System.out.println("Tổng 4 số nhỏ nhất: " +sumMin);}

# 16. Thuật toán xử lý **mảng:** loại bỏ giá trị trùng lặp, đếm số xuất hiện của mỗi từ trong mảng.

public class RemoveAndCountWordsInArrays {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

System.out.print("Nhập số lượng phần tử trong mảng: ");

int quanlity = Integer.parseInt(scanner.nextLine());

String[] input = new String[quanlity];

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

System.out.print("Nhập phần tử thứ " + (i + 1) + " ");

String ele = scanner.nextLine(); input[i] = ele; }

System.out.println(Arrays.toString(input));

Set<String> set = new LinkedHashSet<>(Arrays.asList(input));

System.out.println("Kết quả đã loại bỏ phần tử trùng lặp" + set);

Map<String, Integer> map = new LinkedHashMap<>();

for (String item : input) {

if (!map.containsKey(item)) { map.put(item, 1); }

else { map.put(item, map.get(item) + 1); } }

System.out.println("Kết quả đếm giá trị trùng lặp" + map); }}