פתרון מוצע לבחינות מה"ט באלגוריתמיקה

מועד ב' תשפ"ג, שנה: אביב 2023

בעריכת גל -בראון

```
public class Q1 {
public static void printArr(int arr[]) {
for(int j = 0; j < arr.length; j++) {
System.out.print(arr[j] + ",");
}
System.out.println();
}
public static void main(String[] args) {
Scanner in = new Scanner(System.in);
int num;
int total = 0;
int count = 0;
do {
System.out.print("Enter a number: ");
num = in.nextInt();
if(num != 500) {
total += num;
count++;
if(num % 2 == 1) {
int sum = 0;
while(num > 0) {
sum += num % 10;
num /= 10;
}
System.out.println(sum);
}
}
} while(num != 500);
System.out.println("Avg: " + (total / count));
}
}
public class Q2 {
public static boolean isOK(String str, char ch) {
```

```
int count = 0;
int i;
for(i = 0; i < str.length()-1 ; i++) {
if(str.charAt(i) == ch && str.charAt(i+1) != ch ) count++;
if(str.charAt(i) == ch && str.charAt(i+1) == ch ) return false;
}
if(str.charAt(i) == ch) count++;
return count == 2;
}
}
public class Q3 {
public static boolean isSetOfFriends(int[] arr) {
for(int i = 0; i < arr.length ; i++) {
int count = 0;
for(int j = 0; j < arr.length ; j++) {
if(arr[i] == arr[j]) count++;
}
if(count != 2) return false;
}
return true;
}
public static int[] newArr() {
int[] arr = new int[20];
for(int i = 0; i < arr.length ; i++) {
arr[i] = (int)(10 + Math.random() * 90);
}
return arr;
}
public static void main(String[] args) {
int[] arr = null;
int count = 0;
do {
arr = newArr();
++count;
} while(!isSetOfFriends(arr));
System.out.println(count);
Q1.printArr(arr);
}
}
// Q4
class Box {
String color;
int length;
int width;
int height;
public String getColor() {
return color;
}
public void setColor(String color) {
this.color = color;
}
public int getLength() {
return length;
}
public void setLength(int length) {
this.length = length;
}
public int getWidth() {
return width;
}
public void setWidth(int width) {
this.width = width;
}
public int getHeight() {
return height;
}
public void setHeight(int height) {
this.height = height;
}
public Box(String color) {
this.color = color;
this.length = (int)(20 + Math.random()*81);
this.width = (int)(20 + Math.random()*81);
this.height = (int)(20 + Math.random()*81);
}
public static boolean isBlackWhite(Box[] arr) {
boolean b = false;
boolean w = false;
for(int i = 0; i < arr.length; i++) {
if(arr[i].getColor().equals("Black")){
b = true;
} else if(arr[i].getColor().equals("White")) {
w = true;
} else return false;
}
```

```
return b && w;
}
}
public class Q5 {
public static int sod(int[] arr) { // returns array minimum
int m = arr[0];
double k1,k2;
for(int i = 1; i < arr.length ; i++) {
k1 = (m + arr[i])/2.0;
k2 = Math.abs((m - arr[i])/2.0);
m = (int)(k1 - k2);
}
return m;
}
public static int secret(int[] arr) { // returns array maximum
int m = arr[0];
double k1,k2;
for(int i = 1; i < arr.length ; i++) {
k1 = (m + arr[i])/2.0;
k2 = Math.abs((m - arr[i])/2.0);
m = (int)(k1 + k2);
}
return m;
}
public static void main(String[] args) {
int[] arr = {6,2,-8,12,-4};
int[] brr = {-10,-11,-12,-13,-14,-15};
System.out.println(sod(arr)); // returns array minimum
System.out.println(secret(arr)); // returns array maximum
System.out.println(sod(arr) > secret(brr));
}
}
public class Q6 {
public static String what(String s) {
String str = "";
int i;
for(i = 0; i < s.length()-1 ; i++) {
if(s.charAt(i) < 'A' || s.charAt(i) > 'Z') {
str += s.charAt(i);
} else {
if(s.charAt(i) != s.charAt(i+1)) {
```

```
str += s.charAt(i);
}
}
}
str += s.charAt(i);
return str;
}
public static void main(String[] args) {
// a) removes repeating big letters from string
System.out.println(what("%%ABBCCC??DD"));
// b)
System.out.println(what("%X%YE"));
// c)
System.out.println(what("$$ABBCCDD66abc"));
System.out.println(what("$$ABCD66abc"));
// a) removes repeating big letters from string
}
}
// Q7
class Product{
String name;
String category;
int count;
double price;
public Product(String name, String category, int count, double price) {
this.name = name;
this.category = category;
this.count = count;
this.price = price;
}
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
public String getCategory() {
return category;
}
public void setCategory(String category) {
this.category = category;
}
```

```
public int getCount() {
return count;
}
public void setCount(int count) {
this.count = count;
}
public double getPrice() {
return price;
}
public void setPrice(double price) {
this.price = price;
}
public boolean isCheaper(Product other) {
return price > other.getPrice();
}
public boolean isSame(Product other) {
return name.equals(other.getName())
&& category.equals(other.getCategory());
}
}
class Stock{
Product[] stock;
int numOfProducts;
public Stock() {
this.stock = new Product[100];
this.numOfProducts = 0;
}
public Product[] getStock() {
return stock;
}
public void setStock(Product[] stock) {
this.stock = stock;
}
public int getNumOfProducts() {
return numOfProducts;
}
public void setNumOfProducts(int numOfProducts) {
this.numOfProducts = numOfProducts;
}
public Product mostCheaper(String category) {
int ind = -1;
```

```
double min = 0.0;
for(int i = 0; i < stock.length ; i++) { // find first item in category
if(stock[i].getCategory().equals(category)) {
min = stock[i].getPrice();
ind = i;
break;
}
}
if(ind == -1) return null; // if not found return null
for(int i = ind+1; i < stock.length ; i++) { // find min item in category
if(stock[i].getCategory().equals(category) && min >
stock[i].getPrice()) {
min = stock[i].getPrice();
ind = i;
}
}
return stock[ind];
}
public void updateStock(String name, String category, int count, double price) {
int i;
for(i = 0; stock[i] != null || i < stock.length ; i++) {
if(stock[i].getName().equals(name) &&
stock[i].getCategory().equals(category)) {
stock[i].setCount(count);
if(price < stock[i].getPrice()) stock[i].setPrice(price);
return;
}
}
if(i < stock.length-1) stock[i] = new Product(name,category,count,price);
}
}
public class Q8 {
public static void what(int[] arr) {
int i = 0;
int j = arr.length -1;
int temp;
while(i<j) {
if(arr[i] <= 0) i++;
else if(arr[j] > 0) j--;
else {
temp = arr[i];
arr[i] = arr[j];
arr[j] = temp;
}
}
}
public static void why(int[] arr) {
int[] newArr = new int[arr.length];
int ind = 0;
for(int i = 0; i < arr.length; i++)
if(arr[i] <= 0) newArr[ind++] = arr[i];
for(int i = 0; i < arr.length; i++)
if(arr[i] > 0) newArr[ind++] = arr[i];
for(int i = 0; i < arr.length; i++)
arr[i] = newArr[i];
}
public static void main(String[] args) {
// 1a)
int[] arr = {123,45,-15,0,15,-8};
what(arr);
Q1.printArr(arr); // -8,-15,45,15,123,
// 1b)
int[] brr = {-123,-45,-15,15,8};
what(brr);
Q1.printArr(brr); // -123,-45,-15,15,8,
// 1c) moves all negative numbers to the beginning of the array
// 2a)
int[] crr = {-12,55,0,-46,67};
why(crr);
Q1.printArr(crr); // -12,0,-46,55,67,
// 2b doing same as what
// 2c any same arrays with with all same signs +/-
int[] drr = {-10,-111,-97,-12,-11111};
int[] err = {-10,-111,-97,-12,-11111};
why(drr);
Q1.printArr(drr);
what(err);
Q1.printArr(err);
// 3) O(n)
}
}
public class Q9 {
public static boolean isDivisible(String s, int k) {
int phases = s.length()/k;
```

```
if(s.length() % k != 0) return false;
for(int i = 0; i < phases; i++) {
for(int j = 0; j < k; j++) {
if(s.charAt(i) != s.charAt(j*phases+i)) return false;
}
}
return true;
}
public static int maxDivisor(String s) {
for(int i = s.length(); i > 1 ; i--) {
if(isDivisible(s,i)) return i;
}
return -1;
}
public static String[] unDividable(String[] arr) {
String[] ret = new String[arr.length];
int ind = 0;
for(int i = 0; i < arr.length; i++) {
if(maxDivisor(arr[i]) == -1) ret[ind++] = arr[i];
}
return ret;
}
public static void main(String[] args) {
System.out.println(isDivisible("ABCABCABCABC",4));
System.out.println(maxDivisor("ABABABABABABA"));
}
}
//Q10
class Ball {
String color;
int size;
String material;
public Ball(String color, int size, String material) {
super();
this.color = color;
this.size = size;
this.material = material;
}
public String getColor() {
return color;
}
```

```
public void setColor(String color) {
this.color = color;
}
public int getSize() {
return size;
}
public void setSize(int size) {
this.size = size;
}
public String getMaterial() {
return material;
}
public void setMaterial(String material) {
this.material = material;
}
}
class BallPack {
Ball[] balls;
String[] colors;
int numOfBalls;
String material;
int minSize;
public BallPack(int num, String[] cols, int min, String mat) {
this.balls = new Ball[num];
this.colors = cols;
this.numOfBalls = 0;
this.material = mat;
this.minSize = min;
}
public boolean isFit(Ball b) {
if(!(b.getMaterial().equals(material) && b.getSize() >= minSize)) return false;
for(int i = 0; i < colors.length; i++) {
if(b.getColor().equals(colors[i])) return true;
}
return false;
}
public boolean add(Ball b) {
if(isFit(b)) {
balls[numOfBalls++] = b;
return true;
} else return false;
```

```
}
public int countColor(String color) {
int count = 0;
for(int i = 0; i < balls.length; i++) {
if(balls[i].getColor().equals(color)) count++;
}
return count;
}
public String[] missinColors() {
String[] ret = new String[colors.length];
int ind = 0;
for(int i = 0; i < colors.length; i++) {
boolean exists = false;
for(int j = 0; j < balls.length; j++) {
if(balls[j].getColor().equals(colors[i])) {
exists = true;
break;
}
}
if(!exists) ret[ind++] = colors[i];
}
return ret;
}
}
public class Q11 {
public static boolean inArray(int[] arr, int num) {
for(int i = 0; i < arr.length; i++) {
if(arr[i] == num) return true;
}
return false;
}
public static int[] diffArray(int[] arr, int[] brr) {
int[] ret = new int[arr.length];
int ind = 0;
for(int i = 0; i < arr.length; i++) {
if(!Q11.inArray(brr, arr[i]) && !Q11.inArray(ret, arr[i])) ret[ind++] =
arr[i];
}
return ret;
}
public static int[] completeArray(int[] arr, int[] brr) {
```

```
int[] temp = new int[arr.length + brr.length];
int tind = 0;
for(int i = 0; i < arr.length; i++) {
if(!Q11.inArray(temp, arr[i])) temp[tind++] = arr[i];
}
for(int i = 0; i < brr.length; i++) {
if(!Q11.inArray(temp, brr[i])) temp[tind++] = brr[i];
}
int[] ret = new int[90-tind];
int rind = 0;
for(int i = 10; i < 100; i++) {
if(!Q11.inArray(temp, i)) ret[rind++] = i;
}
return ret;
}
// c) O(n^2)
}
public class Q12 {
public static boolean isCute(int num) {
int left = num;
int right = num % 10;
while(left > 9) left /= 10;
return right % left == 0;
}
public static boolean isMotek(int[][] mat) {
for(int i = 0; i < mat[0].length; i++) {
boolean ret = true;
for(int j = 0; j < mat.length; j++) {
if(!Q12.isCute(mat[j][i])) {
ret = false;
break;
}
}
if(ret) return true;
}
return false;
}
}
// b) O(n^3)
```