Review of Java programming basics

programs

* Hello world program
* public class Q1 {  
   public static void main (String[]args){  
   for (int i = 1; i < 11; i++) {  
   System.*out*.println(i);  
   }  
    
   }  
  }
* Print numbers from 1-10
* public class Q2 {  
   public static void main (String[]args){  
   for (int i = 1; i < 11; i++) {  
   System.*out*.println(i);  
   }  
    
   }  
  }
* Print array elements
* public class Q3 {  
   public static void main (String[]args){  
   int arr[]={1,2,6,8};  
   for (int i = 0; i <arr.length ; i++) {  
   System.*out*.println(arr[i]);  
    
   }  
   }  
  }
* Input array elements
* import java.util.Scanner;  
    
  public class Q4 {  
   public static void main (String[]args){  
   int x[]=new int[4];  
   Scanner input=new Scanner(System.*in*);  
   System.*out*.println("enter the number");  
   for (int i = 0; i <x.length ; i++) {  
   x[i]=input.nextInt();  
    
   }  
   for (int i = 0; i < x.length; i++) {  
   System.*out*.println(x[i]);  
    
   }  
   }  
  }

Define method to print array elementsimport java.util.Scanner;  
  
public class Q5 {  
 public static void main (String[]args){  
 int nam[]={1,2,6,8};  
 print(nam);  
 *inportAll*(nam);  
 print(nam);  
  
 }  
 public static void inportAll(int arr[]){  
 Scanner input=new Scanner(System.*in*);  
 System.*out*.println("enter the number");  
 for (int i = 0; i < arr.length; i++) {  
 arr[i]=input.nextInt();  
  
 }  
 }  
}

* Define method to Input array elements
* public class Q7 {  
   public static void main (String []args){  
   Student x[]=new Student[4];  
   x[0]=new Student("Eman",10);  
   x[1]=new Student("Ayh",11);  
   x[2]=new Student("Fatma",12);  
   x[3]=new Student("Hand",13);  
   for (int i = 0; i < x.length; i++) {  
   System.*out*.println(x[i]);  
    
   }  
    
    
   }  
  }
* Array of objects (Students)
* public class Student {  
   String name;  
   int id;  
   public Student(String name,int id){  
   this.name = name;  
   this.id =id;  
   }  
    
   public String getName(){  
   return name;  
   }  
   @Override  
   public String toString(){  
   return "Student{" +  
   "name ='"+ name+ '\''+  
   ",id="+id+  
   '}';  
    
   }  
   public void setName(String name){  
   this.name=name;  
   }  
   public int getId(){  
   return id;  
   }  
   public void setId(int id){  
   this.id = id;  
   }  
  }

Homework

Create a github account and upload lecture programs to your account

|  |  |
| --- | --- |
| R-1.1 | Write a short Java method, inputAllBaseTypes, that inputs a different value of each base type from the standard input device and prints it back to the standard output device.  import java.util.Scanner;  public class Main {  public static void inputAllBaseTypes() {  Scanner scanner = new Scanner(System.in);  System.out.print("Enter a boolean value: ");  boolean boolValue = scanner.nextBoolean();  System.out.println("Boolean value: " + boolValue);  System.out.print("Enter a byte value: ");  byte byteValue = scanner.nextByte();  System.out.println("Byte value: " + byteValue);  System.out.print("Enter a short value: ");  short shortValue = scanner.nextShort();  System.out.println("Short value: " + shortValue);  System.out.print("Enter an int value: ");  int intValue = scanner.nextInt();  System.out.println("Int value: " + intValue);  System.out.print("Enter a long value: ");  long longValue = scanner.nextLong();  System.out.println("Long value: " + longValue);  System.out.print("Enter a float value: ");  float floatValue = scanner.nextFloat();  System.out.println("Float value: " + floatValue);  System.out.print("Enter a double value: ");  double doubleValue = scanner.nextDouble();  System.out.println("Double value: " + doubleValue);  System.out.print("Enter a char value: ");  char charValue = scanner.next().charAt(0);  System.out.println("Char value: " + charValue);  scanner.close();  }  public static void main(String[] args) {  inputAllBaseTypes();  }  } |
| R-1.2 | Suppose that we create an array A of GameEntry objects, which has an integer scores field, and we clone A and store the result in an array B. If we then immediately set A[4].score equal to 550, what is the score value of the GameEntry object referenced by B[4]?  Copy  class GameEntry {  int score;  public GameEntry(int score) {  this.score = score;  }  }  public class Main {  public static void main(String[] args) {  GameEntry[] A = new GameEntry[5];  A[4] = new GameEntry(500);  GameEntry[] B = A.clone();  A[4].score = 550;  System.out.println("Score in A: " + A[4].score);  System.out.println("Score in B: " + B[4].score);  }  } |
| R-1.3 | Write a short Java method, isMultiple, that takes two long values, n and m, and returns true if and only if n is a multiple of m, that is, n = mi for some integer i.  public class Main {  public static boolean isMultiple(long n, long m) {  if (m == 0) {  // Division by zero is undefined, so m cannot be zero  return false;  }  return n % m == 0;  }  public static void main(String[] args) {  long n = 12;  long m = 4;  boolean result = isMultiple(n, m);  System.out.println(n + " is a multiple of " + m + ": " + result);  }  } |
| R-1.4 | Write a short Java method, isEven, that takes an int i and returns true if and only if i is even. Your method cannot use the multiplication, modulus, or division operators, however.  public class Main {  public static boolean isEven(int i) {  return (i & 1) == 0;  }  public static void main(String[] args) {  int number = 6;  boolean result = isEven(number);  System.out.println(number + " is even: " + result);  }  } |
| R-1.5 | Write a short Java method that takes an integer n and returns the sum of all positive integers less than or equal to n.  public class Main {  public static int sumOfPositiveIntegers(int n) {  if (n <= 0) {  // If n is zero or negative, return zero  return 0;  }  int sum = 0;  for (int i = 1; i <= n; i++) {  sum += i;  }  return sum;  }  public static void main(String[] args) {  int n = 5;  int result = sumOfPositiveIntegers(n);  System.out.println("Sum of positive integers up to " + n + ": " + result);  }  } |
| R-1.6 | Write a short Java method that takes an integer n and returns the sum of all the odd positive integers less than or equal to n.  public class Main {  public static int sumOfOddPositiveIntegers(int n) {  if (n <= 0) {  // If n is zero or negative, return zero  return 0;  }  int sum = 0;  for (int i = 1; i <= n; i += 2) {  sum += i;  }  return sum;  }  public static void main(String[] args) {  int n = 10;  int result = sumOfOddPositiveIntegers(n);  System.out.println("Sum of odd positive integers up to " + n + ": " + result);  }  } |
| R-1.7 | Write a short Java method that takes an integer n and returns the sum of the squares of all positive integers less than or equal to n.  public class Main {  public static int sumOfSquares(int n) {  if (n <= 0) {  // If n is zero or negative, return zero  return 0;  }  int sum = 0;  for (int i = 1; i <= n; i++) {  sum += i \* i;  }  return sum;  }  public static void main(String[] args) {  int n = 4;  int result = sumOfSquares(n);  System.out.println("Sum of squares of positive integers up to " + n + ": " + result);  }  } |
| R-1.8 | Write a short Java method that counts the number of vowels in a given character string.  public class Main {  public static int countVowels(String str) {  if (str == null || str.isEmpty()) {  // If the string is null or empty, there are no vowels  return 0;  }  // Convert the string to lowercase to handle both uppercase and lowercase vowels  str = str.toLowerCase();  int count = 0;  for (int i = 0; i < str.length(); i++) {  char ch = str.charAt(i);  if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u') {  count++;  }  }  return count;  }  public static void main(String[] args) {  String str = "Hello, World!";  int vowelCount = countVowels(str);  System.out.println("Number of vowels in the string: " + vowelCount);  }  } |
| R-1.9 | Write a short Java method that uses a StringBuilder instance to remove all the punctuation from a string s storing a sentence, for example, transforming the string "Let’s try, Mike!" to "Lets try Mike".  public class Main {  public static String removePunctuation(String s) {  if (s == null || s.isEmpty()) {  // If the string is null or empty, return it as is  return s;  }  StringBuilder sb = new StringBuilder();  for (int i = 0; i < s.length(); i++) {  char ch = s.charAt(i);  if (Character.isLetter(ch) || Character.isWhitespace(ch)) {  sb.append(ch);  }  }  return sb.toString();  }  public static void main(String[] args) {  String s = "Let’s try, Mike!";  String result = removePunctuation(s);  System.out.println("String after removing punctuation: " + result);  }  } |
| R-1.10 | Write a Java class, Flower, that has three instance variables of type String, int, and float, which respectively represent the name of the flower, its number of petals, and price. Your class must include a constructor method that initializes each variable to an appropriate value, and your class should include methods for setting the value of each type, and getting the value of each type.  public class Flower {  private String name;  private int petals;  private float price;  public Flower(String name, int petals, float price) {  this.name = name;  this.petals = petals;  this.price = price;  }  public void setName(String name) {  this.name = name;  }  public void setPetals(int petals) {  this.petals = petals;  }  public void setPrice(float price) {  this.price = price;  }  public String getName() {  return name;  }  public int getPetals() {  return petals;  }  public float getPrice() {  return price;  }  public static void main(String[] args) {  Flower rose = new Flower("Rose", 12, 2.99f);  System.out.println("Flower: " + rose.getName());  System.out.println("Petals: " + rose.getPetals());  System.out.println("Price: $" + rose.getPrice());  }  } |
| R-1.11 | Modify the CreditCard class from Code Fragment 1.5 to include a method that updates the credit limit.  public class CreditCard {  private String customer;  private String bank;  private String account;  private int limit;  protected double balance;  public CreditCard(String customer, String bank, String account, int limit, double balance) {  this.customer = customer;  this.bank = bank;  this.account = account;  this.limit = limit;  this.balance = balance;  }  public void updateCreditLimit(int newLimit) {  limit = newLimit;  }  public String getCustomer() {  return customer;  }  public String getBank() {  return bank;  }  public String getAccount() {  return account;  }  public int getLimit() {  return limit;  }  public double getBalance() {  return balance;  }  public boolean charge(double price) {  if (price + balance > limit) {  return false;  }  balance += price;  return true;  }  public void makePayment(double amount) {  balance -= amount;  }  public static void main(String[] args) {  CreditCard card = new CreditCard("John Doe", "Bank of America", "1234567890", 5000, 1000.0);  System.out.println("Customer: " + card.getCustomer());  System.out.println("Bank: " + card.getBank());  System.out.println("Account: " + card.getAccount());  System.out.println("Limit: $" + card.getLimit());  System.out.println("Balance: $" + card.getBalance());  card.updateCreditLimit(6000);  System.out.println("New Limit: $" + card.getLimit());  }  } |
| R-1.12 | Modify the CreditCard class from Code Fragment 1.5 so that it ignores any request to process a negative payment amount.  public class CreditCard {  private String customer;  private String bank;  private String account;  private int limit;  protected double balance;  public CreditCard(String customer, String bank, String account, int limit, double balance) {  this.customer = customer;  this.bank = bank;  this.account = account;  this.limit = limit;  this.balance = balance;  }  public void updateCreditLimit(int newLimit) {  limit = newLimit;  }  public String getCustomer() {  return customer;  }  public String getBank() {  return bank;  }  public String getAccount() {  return account;  }  public int getLimit() {  return limit;  }  public double getBalance() {  return balance;  }  public boolean charge(double price) {  if (price + balance > limit) {  return false;  }  balance += price;  return true;  }  public void makePayment(double amount) {  if (amount < 0) {  System.out.println("Invalid payment amount. Payment amount cannot be negative.");  return;  }  balance -= amount;  }  public static void main(String[] args) {  CreditCard card = new CreditCard("John Doe", "Bank of America", "1234567890", 5000, 1000.0);  System.out.println("Customer: " + card.getCustomer());  System.out.println("Bank: " + card.getBank());  System.out.println("Account: " + card.getAccount());  System.out.println("Limit: $" + card.getLimit());  System.out.println("Balance: $" + card.getBalance());  card.makePayment(200);  System.out.println("Updated Balance: $" + card.getBalance());  card.makePayment(-100); // Negative payment amount, ignored  System.out.println("Balance after negative payment: $" + card.getBalance());  }  } |
| R-1.13 | Modify the declaration of the first for loop in the main method in Code Fragment 1.6 so that its charges will cause exactly one of the three credit cards to attempt to go over its credit limit. Which credit card is it?  public static void main(String[] args) {  CreditCard[] wallet = new CreditCard[3];  wallet[0] = new CreditCard("John Bowman", "California Savings",  "5391 0375 9387 5309", 5000);  wallet[1] = new CreditCard("John Bowman", "California Federal",  "3485 0399 3395 1954", 3500);  wallet[2] = new CreditCard("John Bowman", "California Finance",  "5391 0375 9387 5309", 2500);  for (int val = 1; val <= 16; val++) {  wallet[0].charge(1250); // Charging an amount equal to the credit limit  wallet[1].charge(125); // Charging a smaller amount  wallet[2].charge(100); // Charging a smaller amount  }  for (int i = 0; i < 3; i++) {  System.out.println("Credit Card " + (i + 1) + " balance: $" + wallet[i].getBalance());  }  } |