객제지향프로그래밍 3장 실습 문제 보고서

글로벌미디어학부20192722 이지희

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 2번 | 4번 | 6번 | 8번 | 10번 | 12번 | 14번 | 16번 |
| O | O | O | O | O | O |  | O |

2.

|  |
| --- |
| public class OPP\_3\_2 {  public static void main(String[] args) {  int n[][]={{1},{1,2,3},{1},{1,2,3,4},{1,2}};  for(int i=0;i<n.length;i++){  for(int j=0;j<n[i].length;j++){  System.*out*.print(n[i][j] + " ");  }  System.*out*.println();  }  } } |

* 실행 화면
* 부연 설명

4.

|  |
| --- |
| import java.util.Scanner;  public class OPP\_3\_4 {  public static void main(String[] args) {  System.*out*.print("소문자 알파벡 하나를 입력하시오.>>");  Scanner scanner = new Scanner(System.*in*);  String s = scanner.next();  char c=s.charAt(0);  for(int i=0;i<c-'a'+1;i++){  for(int j=0;j<c-'a'-i+1;j++){  char outChar =(char)('a' + j);  System.*out*.print(outChar);  }  System.*out*.println();  }  scanner.close();  } } |

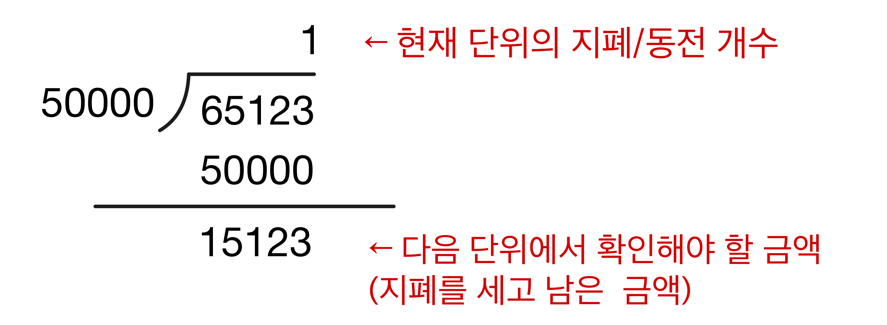
* 실행화면
* 부연설명

6.

|  |
| --- |
| import java.util.Scanner;  public class OPP\_3\_6 {  public static void main(String[] args) {  int [] uint={50000,10000,1000,500,100,50,10,1};  int[] count=new int[8];  System.*out*.print("금액을 입력하시오.>>");  Scanner scanner = new Scanner(System.*in*);  int money=scanner.nextInt();  //지폐 세기  for(int i=0;i<uint.length;i++){  count[i]=money/uint[i];  if(count[i]!=0)  System.*out*.println(uint[i]+"원 짜리: "+count[i]);  money%=uint[i];  }  scanner.close();  } } |

* 실행 화면
* 부연설명

각 지페 / 동전의 개수를 저장하는 count 배열을 만든 후 크기를 단위의 개수인 8로 초기화 한다.



지폐의 개수는 입력된 금액을 각 단위로 나누었을 때 몫이다. 개수가 0이 아니라면 지폐/동전의 개수가 몇개인지 출력한다. 그 다음 단위에서 지폐를 세기 위해서 money의 값을 단위로 나눈 나머지로 변경한다.

8.

|  |
| --- |
| import java.util.Scanner;  public class OPP8 {  public static void main(String[] args) {  int[][] rect = new int[2][2];  Scanner scanner= new Scanner(System.*in*);  for(int i=0;i<rect.length;i++){  for(int j=0;j<rect[i].length;j++){  rect[i][j]=scanner.nextInt();  }  }  if(*inRect*(rect[0][0],rect[0][1],100,100,200,200)||*inRect*(rect[1][0],rect[1][1],100,100,200,200)){  System.*out*.println("사각형이 충돌합니다.");  }  else System.*out*.println("사각형이 충돌하지 않습니다.");  scanner.close();   }   public static boolean inRect(int x,int y, int rectx1, int recty1, int rectx2, int recty2){  if((x>=rectx1 && x<=rectx2) && (y>=recty1 && y<=recty2)){  return true;  }  else return false;   } } |

* 실행화면

텍스트이(가) 표시된 사진

자동 생성된 설명

* 부연 설명

두 직사각형이 충돌하는 경우는 직사각형을 결정하는 두 점 중 한 점이 (100,100),(200,200)의 안에 있는 경우이다. 한 점이 사각형 안에 있는 경우는 입력받은 x의 범위가 100<x<200 이고 y의 범위가 100<y<200를 만족해야 한다.

힌트로 주어진 메서드를 이용하여 프로그램을 작성하면, 첫번째 점이 inRect함수의 매개변수로 들어가서 true를 반환하거나 두번째 점이 매개변수가 되었을 때 true를 반환해야 한다.

10

|  |
| --- |
| import java.util.Scanner;  public class OPP10 {  public static void main(String[] args) {  int[][] circle = new int[2][3];  Scanner scanner=new Scanner(System.*in*);  System.*out*.print("첫번째 원의 중심과 반지름 입력>>");  for(int i=0;i<circle[0].length;i++){  circle[0][i]=scanner.nextInt();  }  System.*out*.print("두번째 원의 중심과 반지름 입력>>");  for(int j=0;j<circle[1].length;j++){  circle[1][j]=scanner.nextInt();  }   double d=*distance*(circle[0][0],circle[0][1],circle[1][0],circle[1][1]);  if((circle[0][2]+circle[1][2])>d)  System.*out*.println("두 원은 서로 겹친다.");  else  System.*out*.println("두 원은 서로 겹치지 않는다.");  scanner.close();  }  public static double distance(int x1, int y1, int x2, int y2){  return Math.*sqrt*((x1-x2)\*(x1-x2)+(y1-y2)\*(y1-y2));  } } |

* 실행화면

텍스트이(가) 표시된 사진

자동 생성된 설명

* 부연설명

두 원의 중심점의 x,y 좌표와 반지름을 입력 받을 2차원 배열 circle을 선언하고 for문과 Scanner 객체를 이용하여 각각의 값을 입력 받는다.

두 원이 충돌하는 경우(겹치는 경우)는 두 원의 반지름을 더한 값이 중심거리보다 커야 한다. 따라서, 첫번째 원의 반지름이 r이고 두번째 원의 반지름이 r', 두 중심 사이 거리가 d일 때 r + r' > d을 만족해야 한다.

두 점 사이의 거리를 구하기 위해 distance 메서드를 만들었다. 두 점 사이 거리를 구하는 공식인 을 이용하였다. 이때, Math 클래스의 sqrt 함수를 사용하였다.

12.(1)

|  |
| --- |
| import java.util.Scanner;  public class OPP12\_1 {  public static void main(String[] args) {  System.*out*.print("연산>>");  Scanner scanner= new Scanner(System.*in*);  double op1 = scanner.nextInt();  String op = scanner.next();  double op2 = scanner.nextInt();  double result=0;  if(op.equals("+")) result=op1+op2;  else if(op.equals("-")) result=op1-op2;  else if(op.equals("\*")) result=op1\*op2;  else if(op.equals("/")) {  if(op1==0 || op2==0) {  System.*out*.println("0으로 나눌 수 없습니다.");  scanner.close();  return;  }  result=op1/op2;  }  System.*out*.println(op1+op+op2+"의 계산결과는"+result);  scanner.close();  } } |

* 실행화면

텍스트이(가) 표시된 사진

자동 생성된 설명

* 부연설명 : 연산의 피연산자 부분과 연산자 부분을 나누어서 각 변수에 Scanner 객체를 이용하여 입력받았다. 그 뒤 else if문을 이용하여 각 사칙연산의 케이스를 나누고 equals 메서드를 이용하여 string형 변수 op와 비교하였다. 0으로 나누는 경우는 ("0으로 나눌 수 없습니다."를 출력하고 프로그램을 종료하도록 했다.

12(2)

|  |
| --- |
| import java.util.Scanner;  public class OPP12\_2 {  public static void main(String[] args) {  System.*out*.print("연산>>");  Scanner scanner= new Scanner(System.*in*);  double op1 = scanner.nextInt();  String op = scanner.next();  double op2 = scanner.nextInt();  double result=0;  switch (op){  case "+":  result=op1+op2;  break;  case "-":  result=op1-op2;  break;  case "/":  if(op2==0) {  System.*out*.println("0으로 나눌 수 없습니다.");  scanner.close();  return;  }  result=op1/op2;  break;  case "\*":  result=op1\*op2;  break;  }  System.*out*.println(op1+op+op2+" 의 계산결과는 "+result);  } } |

* 실행화면

텍스트이(가) 표시된 사진

자동 생성된 설명

* 부연설명

12와 풀이 과정은 동일하다. If else 구문을 switch로 바꾸어서 op를 비교하여 각각 연산을 진행한다.