1. 保留两位小数：

Double number = input.nextDouble();

System.out.println( (int)(number\*100)/100.0);

1. 总秒数——>时间：

Seconds = Total % 60;

totalMinutes = Total / 60;…

3 judge the big or small and change to a>=b;

if( a<b ){

a = b - a;

b = b - a;

a = b + a; }

1. 计算持续时间，程序开始时调用 long start = System.currentTimeMillis();

结束时调用 long end = System.currentTimeMillis();

End – start

1. 在循环控制中，不要用浮点值来比较value是否相等，因为浮点值都是某些值的近似值，使用它们可能导致不精确的循环次数和不准确的结果。
2. 一个数N的最大公约数不可能大于N/2。
3. 蒙特卡罗模拟 :

public class Pi{

public static void main(String[] args){

double number = 100000000 ;

double getNumber = 0;

for (int n = 0; n < number; n++){

double x = Math.random()\*2.0-1;

double y = Math.random()\*2.0-1;

if(x\*x+y\*y<1)

getNumber++;

}

double pi = getNumber/number\*4;

System.out.println("the value of the PI is :" + pi);

}

}

1. 数组随意打乱：

For(int i=0; i< myList.lenth; i++){

//generante an index j randomly

Int index = (int)(Math.random()\*myList.length);

//swap myList[i] with myList[j]

Double temp = myList[i];

myList[i] = myList[index];

myList[index] = temp;

}

1. 移动元素

Double temp = myList[0];

//shift elements left

For(int i = 1; i <myList.length; i++){

myList[i-1] = myList[i];

}

//move the first element to fill in the last position

myList[myList.length-1] = temp;

1. 数组线性查找

Public class LinearSearch{

Public static int linearSearch(int[] list, int key){

For(int i = 0 ; i<list.length; i ++){

If(list[i]== key)

Return i ;

}

Return -1;

}

}

10 数组的二分查找

Public class BinarySearch{

Public static int binarySearch(int[] list , int key){

int low = 0 ;

int high = list.length – 1;

while(high >= low){

int mid = (low+high)/2;

if (key < list[mid])

high = mid -1;

else if (key == list[mid])

return mid;

else

low = mid + 1;

}

Return –low-1;//Now high < low, key not found

}

}

1. 数组的选择排序

Public class SelectionSort{

Public static void selectionSort(double[] list){

For (int i = 0; I < list.length-1 ;i++){

//find the minimum in the list[i…list.length-1]

Double currentMin = list[i];

Int currentMinIndex = I ;

For (int j = i+1 ; J< list.length; j++){

If (currentMin > list[j]){

currentMin = list[j];

currentMinIndex = j;

}

}

//Swap list[i] with list[currentMinIndex] if necessary;

If (currentMinIndex != list[i]){

List[currentMinIndex] = list[i];

List[i] = currentMin;

}

}

}

}

1. 数组插入排序：

Public class InsertionSort{

Public static void insertionSort(double[] list){

For(int i= 1; i<list.length; i++){

/\*\*insert list[i] into a sorted subist list[0..i-1] so that list[0…i] is sorted.\*/

Double currenElement = list[i];

Int k;

For (k = i-1;k >=0&&list[k]>currentElement; k--){

List[k+1] = list[k];

}

//insert the current element into list[k+1]

List[k+1]= currentElement;

}

}

}