Q1. Write a function named *largestDigit* that returns the largest digit of its integer argument. For example largestDigit(3185) returns 8 and largestDigit(-65665) returns 6.

The function signature is  
   int largestDigit (int n)

Hint: use modulo base 10 arithmetic and integer division to isolate the digits of the number.

Copy and paste your answer here and click the "Submit answer" button

Answer 1:

public static void main(String[] args) {

System.out.println(largestDigit(3185));

System.out.println(largestDigit(-65665));

}

private static int largestDigit(int n){

int remainder;

int largestDigit = 0;

n= Math.abs(n);

while(n>0){

  remainder = n % 10;

largestDigit = remainder >largestDigit?remainder:largestDigit;

n =n/10;

}

return largestDigit;

}

Q2.2. A **Daphne array** is defined to be an array where the minimum and maximum values occur exactly once and the minimum value occurs after the maximum value in the array. For example {10, -2, 3, -6, 9} is a Daphne array because it satisfies all the conditions:

1. The maximum value 10 occurs only once
2. The minimum value -6 occurs only once
3. -6 occurs after 10 in the array.

Write a function named *isDaphneArray* that returns 1 if its array argument is a Daphne array, otherwise it returns 0.

If you are programming in Java or C#, the function signature is  
   int isDaphneArray(int[ ] a)

If you are programming in C or C++, the function signature is  
   int isDaphneArray(int a[ ], int len) where len is the number of elements in a.

Copy and paste your answer here and click the "Submit answer" button

A2.

public static void main(String[] args) {

System.out.println(isDaphneArray(new int[]{10, -2, 3, -6, 9}));

}

private static int isDaphneArray(int[ ] a){

int max = a[0];

int min = a[0];

int indexMax=0;

int indexMin=0;

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

{

if(a[i]>max){

max=a[i];

indexMax = i;

}

else if(a[i] == max || a[i]==min){

return 0;

}

else if(a[i]<min) { // here the only case left is if the value is less than min

min = a[i];

indexMin = i;

}

}

return indexMin>indexMax?1:0;

}

Q3.3. An **Olympic array** is defined to be an array in which every value is greater than or equal to the sum of the values less than it. The sum of the values less than the minimum value in the array is defined to be 0.

For example, {3, 2, 1} is an Olympic array because

1. 1 is the minimum value and by definition the sum of the values less than it is 0. Since 1 is greater than 0, it satisfies the condition.
2. There is only one value less than 2 and 2 is greater than it, so the value 2 satisfies the condition.
3. The values 1 and 2 are less than 3 and 3 is equal to their sum, so the value 3 satisfies the condition.

Hence all elements of the array satisfy the conditions and the array is an Olympic array.

{2, 2, 1, 1} is also an Olympic array because the values less than 2 sum to 2.

{1, 1000, 100, 10000, 2} is also an Olympic array. However, {1, 99, 99, 1000, 100, 10000, 2} is **not** an Olympic array because the sum of the numbers less than 100 (99+99+1) is greater than 100. Please be sure that your function detects that this is not an Olympic array!

{1, 2, 1, 3, 2} is not an Olympic array because 3 is not greater than or equal to 1+2+1+2.

{1, 2, -1, 2, 2} is not an Olympic array because -1 is the minimum value but it is not greater than or equal to 0.

Write a function named *isOlympic* that returns 1 if its array argument is an Olympic array, otherwise it returns 0.

If you are writing in Java or C#, the function signature is  
   int isOlympic (int[ ] a)

If you are writing in C or C++, the function signature is  
   int isOlympic(int a[ ], int len) where len is the number of elements in the array.

Hint: use a nested loop.

Copy and paste your answer here and click the "Submit answer" button

A3.public static void main(String[] args) {

   System.out.println(isOlympic(new int[]{3, 2, 1}));

   System.out.println(isOlympic(new int[]{2, 2, 1, 1} ));

   System.out.println(isOlympic(new int[]{1, 1000, 100, 10000, 2}));

   System.out.println(isOlympic(new int[]{1, 99, 99, 1000, 100, 10000, 2}));

}

private static int isOlympic(int a[ ]) {

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

int sumMin=0;

for(int j=0;j<a.length;j++)

{

if(a[j]<a[i]){

sumMin+=a[j];

}

if(a[i]<sumMin){

return 0;

}

}

}

return 1;  // here what is returned is minimum is greater than 0 and every element is greater than or equal to sum of numbers below it

}