Started on	Sunday, 26 April 2020, 3:19 PM
State	Finished
Completed on	Wednesday, 20 May 2020, 10:13 AM
Time taken	23 days 18 hours
Grade	**4.00** out of 10.00 (**40**%)

Question **1**

Correct

Mark 0.67 out of 1.00

Viết chương trình nhập vào n số nguyên và cho biết những số nguyên tố đã nhập

Đầu vào

- Dòng đầu tiên là n - số lượng các số nguyên ($0 < n \leq 1000$).
- Dòng thứ hai chứa n số nguyên, mỗi số cách nhau khoảng trắng

Đầu ra

- Dòng đầu tiên là n số nguyên đã nhập, mỗi số cách nhau khoảng trắng
- Dòng thứ 2 là số lượng các số nguyên tố
- Dòng thứ 3 là các số nguyên tố đã nhập

Ghi chú:

- Dữ liệu đầu vào là hợp lệ.
- Bạn không nên dùng **printf** trước **scanf**.

Xem phần ví dụ để biết chi tiết về đầu vào/đầu ra.

Gợi ý:

- Viết 1 hàm kiểm tra 1 số nguyên m đầu vào có là số nguyên tố hay không?
- Đi qua các phần tử của mảng đầu vào, gọi hàm kiểm tra số nguyên tố trên phần tử vừa đi qua và biện luận giá trị trả về của hàm

For example:

Input	Result
5 1 2 3 4 5	1 2 3 4 5 3 2 3 5
6 54 32 12 45 7 8	54 32 12 45 7 8 1 7
2 6 4	6 4 0
3 11 5 7	11 5 7 3 11 5 7

Answer: (penalty regime: 33.3, 66.7, ... %)

```c
#include <stdio.h>
#include <math.h>
int isPrime(int a){
    int i,flag=1;
    if(a==1){
        flag=0;
    }
    for(i=2; i <= sqrt(a); i++){
        if(a%i==0){
            flag=0;
            break;
        }
    }
    return flag;
}
int main(){
    int A[100],i,n,count=0;
    scanf("%d", &n);
    for(i = 0; i <=n-1; i++){
        scanf("%d", &A[i]);
    }
    for(i = 0; i <= n-1; i++){
        printf("%d ",A[i]);
        if(isPrime(A[i])==1){
            count++;
        }
    }
    printf("\n%d\n",count);
    for(i = 0; i <= n-1; i++){
        if(isPrime(A[i])==1){
            printf("%d ",A[i]);
```

```
32          }
33      }
34      return 0;
35  }
```

	Input	Expected	Got	
✔	5 1 2 3 4 5	1 2 3 4 5 3 2 3 5	1 2 3 4 5 3 2 3 5	✔
✔	6 54 32 12 45 7 8	54 32 12 45 7 8 1 7	54 32 12 45 7 8 1 7	✔
✔	2 6 4	6 4 0	6 4 0	✔
✔	3 11 5 7	11 5 7 3 11 5 7	11 5 7 3 11 5 7	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00. Accounting for previous tries, this gives **0.67/1.00**.

Question **2**

Correct

Mark 0.67 out of 1.00

Viết chương trình tìm kiếm và thay thế tất cả các giá trị X trong mảng số nguyên A thành giá trị Y.

Đầu vào

- Dòng đầu tiên là n - số lượng các số nguyên ($0 < n \leq 1000$).
- Dòng thứ hai chứa n số nguyên, mỗi số cách nhau khoảng trắng
- Dòng thứ ba là 2 giá trị X,Y; mỗi giá trị cách nhau khoảng trắng

Đầu ra

- Dòng đầu tiên là n số nguyên đã nhập, mỗi số cách nhau khoảng trắng
- Dòng thứ 2 là các số nguyên sau khi đã thay thế các giá trị X thành Y

Ghi chú:

- Dữ liệu đầu vào là hợp lệ.
- Bạn không nên dùng **printf** trước **scanf**.

Xem phần ví dụ để biết chi tiết về đầu vào/đầu ra.

Gợi ý:

- Nhập mảng số nguyên A gồm n phần tử.
- Hiển thị mảng A
- Nhập số nguyên X, Y
- Duyệt qua mảng A, nếu thấy 1 phần tử là X thì thay đổi phần tử đó là Y
- Hiển thị mảng A để xem kết quả

For example:

Input	Result
5 1 20 30 20 5 20 2	1 20 30 20 5 1 2 30 2 5
6 54 32 12 45 7 8 -12 10	54 32 12 45 7 8 54 32 12 45 7 8
2 6 6 6 600	6 6 600 600

Answer: (penalty regime: 33.3, 66.7, ... %)

```
#include <stdio.h>
int main(){
    int n,i;
    scanf("%d", &n);
    int A[n-1];
    for(i = 0; i <= n-1; i++){
        scanf("%d",&A[i]);
    }
    int x, y;
    scanf("%d%d", &x, &y);
    for(i = 0; i <= n-1; i++){
        printf("%d ",A[i]);
    }
    printf("\n");
    for(i = 0; i <= n-1; i++){
        if(A[i]==x){
            A[i]=y;
        }
        printf("%d ",A[i]);
    }
    return 0;
}
```

	Input	Expected	Got	

	Input	Expected	Got	
✔	5 1 20 30 20 5 20 2	1 20 30 20 5 1 2 30 2 5	1 20 30 20 5 1 2 30 2 5	✔
✔	6 54 32 12 45 7 8 -12 10	54 32 12 45 7 8 54 32 12 45 7 8	54 32 12 45 7 8 54 32 12 45 7 8	✔
✔	2 6 6 6 600	6 6 600 600	6 6 600 600	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00. Accounting for previous tries, this gives **0.67/1.00**.

Question **3**

Correct

Mark 1.00 out of 1.00

Viết chương trình nhập vào n số nguyên và cho biết giá trị lớn nhất, giá trị nhỏ nhất trong mảng

Đầu vào

- Dòng đầu tiên là n - số lượng các số nguyên ($0 < n \leq 1000$).
- Dòng thứ hai chứa n số nguyên, mỗi số cách nhau khoảng trắng

Đầu ra

- Dòng đầu tiên là n số nguyên đã nhập, mỗi số cách nhau khoảng trắng
- Dòng thứ 2 là giá trị nhỏ nhất và giá trị lớn nhất, mỗi giá trị cách nhau khoảng trắng

Ghi chú:

- Dữ liệu đầu vào là hợp lệ.
- Bạn không nên dùng **printf** trước **scanf**.

Xem phần ví dụ để biết chi tiết về đầu vào/đầu ra.

For example:

Input	Result
5 10 2 30 4 5	10 2 30 4 5 2 30
6 54 32 12 45 7 8	54 32 12 45 7 8 7 54
2 6 4	6 4 4 6
1 3	3 3 3

Answer: (penalty regime: 33.3, 66.7, ... %)

```c
#include <stdio.h>
int main(){
    int n,i;
    scanf("%d", &n);
    int A[n-1];
    for(i = 0; i <= n-1; i++){
        scanf("%d",&A[i]);
    }
    for(i = 0; i <= n-1; i++){
        printf("%d ",A[i]);
    }
    printf("\n");
    int max, min;
    max=A[0];
    min=A[0];
    for(i = 0; i <= n-1; i++){
        if(max < A[i]){
            max = A[i];
        }
        if(min > A[i]){
            min = A[i];
        }
    }
    printf("%d %d", min, max);
    return 0;
}
```

	Input	Expected	Got	
✔	5 10 2 30 4 5	10 2 30 4 5 2 30	10 2 30 4 5 2 30	✔
✔	6 54 32 12 45 7 8	54 32 12 45 7 8 7 54	54 32 12 45 7 8 7 54	✔

	Input	Expected	Got	
✔	2 6 4	6 4 4 6	6 4 4 6	✔
✔	1 3	3 3 3	3 3 3	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **4**

Correct

Mark 0.00 out of 1.00

Viết 1 hàm cho phép kiểm tra xem mảng A gồm n số nguyên có biểu diễn 1 cấp số nhân công bội k, với phần tử đầu tiên là 1.

Ví dụ:

- Cấp số nhân công bội 2, phần tử đầu tiên là 1: 1 2 4 8 16 32 ...
- Cấp số nhân công bội 5, phần tử đầu tiên là 1: 1 5 25 125 625 ...

Khuôn dạng (Prototype)

- Tên hàm (function name): **isSeries()**
- Tham số (parameters):
  - **k -** kiểu int (1 <k <= 6)
  - **A[]** - kiểu int
  - **n** - kiểu int (1<n<=10)
- Kiểu trả về (return type): **int**

Thân hàm (Body)

- Kết quả là 1 nếu A biểu diễn cấp số nhân công bội k với phần tử đầu tiên là 1; ngược lại kết quả là 0

Chú ý

- Giá trị của các tham số luôn hợp lệ, không cần kiểm tra.
- Chỉ viết hàm, KHÔNG VIẾT TOÀN BỘ CHƯƠNG TRÌNH
- Xem thêm chi tiết trong phần **For example**.

Gợi ý

- Duyệt qua mảng, kiểm tra phần tử hiện tại có gấp k lần phần tử trước đó hay không?
- Chú ý phần tử đầu tiên phải là 1

For example:

Test	Result

`int A[]={1, 2, 4, 8, 16, 32, 64, 128};`
`int i;`
`int n = sizeof(A)/sizeof(int);`
`for(i=0;i<=n-1;i++)`
`  printf("%d ",A[i]);`
`printf("\n");`
 	
`if (isSeries(2,A,n))`
`  printf("YES");`
`else printf("NO");`
	`1 2 4 8 16 32 64 128`
	`YES`

`int A[]={2, 4, 8, 16, 32, 64, 128};`
`int i;`
`int n = sizeof(A)/sizeof(int);`
`for(i=0;i<=n-1;i++)`
`  printf("%d ",A[i]);`
`printf("\n");`
 	
`if (isSeries(2,A,n))`
`  printf("YES");`
`else printf("NO");`
	`2 4 8 16 32 64 128`
	`NO`

Answer: (penalty regime: 33.3, 66.7, ... %)

```
int isSeries(int k, int A[], int n){
    int i,flag=0;
    for(i = 0; i <= n-1; i++){
        if(A[0]!=1){
            return flag;
        }
        if(A[i]*k==A[i+1]||n==1){
            flag=1;
            break;
        }
```

```
    }
    return flag;
}
```

	Test	Expected	Got	

✔	int A[]={1, 2, 4, 8, 16, 32, 64, 128};
	int i;
	int n = sizeof(A)/sizeof(int);
	for(i=0;i<=n-1;i++)
	  printf("%d ",A[i]);
	printf("\n");
	
	if (isSeries(2,A,n))
	  printf("YES");
	else printf("NO");
		1 2 4 8 16 32 64 128
		YES
			1 2 4 8 16 32 64 128
			YES
				✔

✔	int A[]={2, 4, 8, 16, 32, 64, 128};
	int i;
	int n = sizeof(A)/sizeof(int);
	for(i=0;i<=n-1;i++)
	  printf("%d ",A[i]);
	printf("\n");
	
	if (isSeries(2,A,n))
	  printf("YES");
	else printf("NO");
		2 4 8 16 32 64 128
		NO
			2 4 8 16 32 64 128
			NO
				✔

✔	int A[]={1};
	int i;
	int n = sizeof(A)/sizeof(int);
	for(i=0;i<=n-1;i++)
	  printf("%d ",A[i]);
	printf("\n");
	printf("%d",isSeries(2,A,n));
		1
		1
			1
			1
				✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00. Accounting for previous tries, this gives **0.00/1.00**.

Question **5**

Correct

Mark 0.00 out of 1.00

Viết 1 hàm cho phép đảo ngược sự xuất hiện của n số nguyên ($0 < n \leq 1000$).

Khuôn dạng (Prototype)

- Tên hàm (function name): **makeReverse()**
- Tham số (parameters):
  - **A[]** - kiểu int
  - **n** - kiểu int
- Kiểu trả về (return type): **void**

Thân hàm (Body)

- Tiến hành đảo ngược các phần tử trong mảng A

Chú ý

- Giá trị của các tham số luôn hợp lệ, không cần kiểm tra.
- Chỉ viết hàm, KHÔNG VIẾT TOÀN BỘ CHƯƠNG TRÌNH
- Xem thêm chi tiết trong phần **For example**.

Chú ý

- Xem lại bài kiểm tra mảng đối xứng

For example:

Test	Result

int A[]={1,2,3,4};
int i;
int n = sizeof(A)/sizeof(int);
for(i=0;i<=n-1;i++)
  printf("%d ",A[i]);
printf("\n");
makeReverse(A,n);
for(i=0;i<=n-1;i++)
  printf("%d ",A[i]);
	1 2 3 4
	4 3 2 1

int A[]={1,2,3,4,5};
int i;
int n = sizeof(A)/sizeof(int);
for(i=0;i<=n-1;i++)
  printf("%d ",A[i]);
printf("\n");
makeReverse(A,n);
for(i=0;i<=n-1;i++)
  printf("%d ",A[i]);
	1 2 3 4 5
	5 4 3 2 1

Answer: (penalty regime: 33.3, 66.7, ... %)

```
void makeReverse(int A[], int n){
    int i, temp;
    for(i = 0; i <= n/2-1; i++){
        temp = A[i];
        A[i] = A[n-i-1];
        A[n-1-i] = temp;

    }
}
```

	Test	Expected	Got	

✔	int A[]={1,2,3,4};
	int i;
	int n = sizeof(A)/sizeof(int);
	for(i=0;i<=n-1;i++)
	  printf("%d ",A[i]);
	printf("\n");
	makeReverse(A,n);
	for(i=0;i<=n-1;i++)
	  printf("%d ",A[i]);
		1 2 3 4
		4 3 2 1
			1 2 3 4
			4 3 2 1
				✔

✔	int A[]={1,2,3,4,5};
	int i;
	int n = sizeof(A)/sizeof(int);
	for(i=0;i<=n-1;i++)
	  printf("%d ",A[i]);
	printf("\n");
	makeReverse(A,n);
	for(i=0;i<=n-1;i++)
	  printf("%d ",A[i]);
		1 2 3 4 5
		5 4 3 2 1
			1 2 3 4 5
			5 4 3 2 1
				✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00. Accounting for previous tries, this gives **0.00/1.00**.

Question **6**

Correct

Mark 0.00 out of 1.00

Viết 1 hàm cho phép lưu trữ n+1 số hạng đầu tiên của dãy Fibonaci vào một mảng A.

Khuôn dạng (Prototype)

- Tên hàm (function name): **getFibo()**
- Tham số (parameters):
  - **A[]** - kiểu int
  - **n** - kiểu int $(0 < n \leq 40)$
- Kiểu trả về (return type): **void**

Thân hàm (Body)

- Lưu trữ n+1 số hạng đầu tiên của dãy Fibonaci vào mảng A, với số hạng thứ i bất kỳ của dãy Fibonaci được tính theo công thức sau:

$$F_i = \begin{cases} 1, & \text{i=0 hoặc i=1} \\ F_{i-2} + F_{i-1}, & \text{i>1} \end{cases}$$

Chú ý

- Giá trị của các tham số luôn hợp lệ, không cần kiểm tra.
- Chỉ viết hàm, KHÔNG VIẾT TOÀN BỘ CHƯƠNG TRÌNH
- Xem thêm chi tiết trong phần **For example**.

For example:

Test	Result

int A[50];
int i,n;
n=10;
getFibo(A,n);
for(i=0;i<=n;i++)
  printf("%d ",A[i]);
	1 1 2 3 5 8 13 21 34 55 89

Answer: (penalty regime: 33.3, 66.7, ... %)

```
void getFibo(int A[], int n){
    int i,f1=0,f2=1;
    int fn=1;
    for(i = 0; i <= n; i++){
        A[i] = fn;
        fn=f1+f2;
        f1=f2;
        f2=fn;
    }
}
```

	Test	Expected	Got	

✔	int A[50];
	int i,n;
	n=10;
	getFibo(A,n);
	for(i=0;i<=n;i++)
	  printf("%d ",A[i]);
		1 1 2 3 5 8 13 21 34 55 89	1 1 2 3 5 8 13 21 34 55 89	✔

	Test	Expected	Got	

✔	int A[50];
	int i,n;
	n=40;
	getFibo(A,n);
	for(i=0;i<=n;i++)
	printf("%d
	",A[i]);
	\t
		1 1 2 3 5 8 13 21 34 55 89 144 233 377
		610 987 1597 2584 4181 6765 10946
		17711 28657 46368 75025 121393 196418
		317811 514229 832040 1346269 2178309
		3524578 5702887 9227465 14930352
		24157817 39088169 63245986 102334155
		165580141
			1 1 2 3 5 8 13 21 34 55 89 144 233 377
			610 987 1597 2584 4181 6765 10946
			17711 28657 46368 75025 121393 196418
			317811 514229 832040 1346269 2178309
			3524578 5702887 9227465 14930352
			24157817 39088169 63245986 102334155
			165580141
				✔

✔				✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00. Accounting for previous tries, this gives **0.00/1.00**.

Question **7**

Correct

Mark 1.00 out of 1.00

Viết hàm **đệ quy** readArray để đọc các số nguyên và lưu nó vào trong một mảng

Khuôn dạng (prototype)

- Tên hàm: readArray
- Tham số:
  - A: int[] - mảng dùng để lưu các số nguyên
  - n: int - số phần tử cần đọc
- Kiểu trả về: void

Thân hàm:

- Lần lượt đọc n số và lưu nó vào mảng A

Chú ý

- Dữ liệu đầu vào luôn hợp lệ.
- Chỉ viết **HÀM**, không viết TOÀN BỘ CHƯƠNG TRÌNH
- PHẢI sử dụng đệ quy, KHÔNG ĐƯỢC sử dụng vòng lặp
- Xem thêm phần **For example** để biết thêm chi tiết về đầu vào/đầu ra

Gợi ý

- Xem bài in các chữ số của số n
  - Nếu n = 0, không làm gì cả
  - Ngược lại,
    - Đọc n-1 số đầu (đệ quy)
    - Đọc số cuối và lưu vào A[n-1]

For example:

Input	Result
5 1 2 3 4 5	1 2 3 4 5
6 54 32 12 45 7 8	54 32 12 45 7 8
2 6 10	6 10
3 9 5 7	9 5 7

Answer: (penalty regime: 33.3, 66.7, ... %)

```
void readArray(int A[], int n){
    if(n == 0){
    }
    else{
    readArray(A, n-1);
    scanf("%d",&A[n-1]);
    }
}
```

	Input	Expected	Got	
✔	5 1 2 3 4 5	1 2 3 4 5	1 2 3 4 5	✔

	Input	Expected	Got	
✔	6 54 32 12 45 7 8	54 32 12 45 7 8	54 32 12 45 7 8	✔
✔	2 6 10	6 10	6 10	✔
✔	3 9 5 7	9 5 7	9 5 7	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00.

Question **8**

Correct

Mark 0.00 out of 1.00

Viết chương trình nhập vào n số nguyên và cho biết những số nguyên lẻ trong đã nhập

Đầu vào

- Dòng đầu tiên là n - số lượng các số nguyên ($0 < n \leq 1000$).
- Dòng thứ hai chứa n số nguyên, mỗi số cách nhau khoảng trắng

Đầu ra

- Dòng đầu tiên là n số nguyên đã nhập, mỗi số cách nhau khoảng trắng
- Dòng thứ 2 là số lượng các số nguyên lẻ
- Dòng thứ 3 là các số nguyên lẻ đã nhập

Ghi chú:

- Dữ liệu đầu vào là hợp lệ.
- Bạn không nên dùng **printf** trước **scanf**.

Xem phần ví dụ để biết chi tiết về đầu vào/đầu ra.

For example:

Input	Result
5 1 2 3 4 5	1 2 3 4 5 3 1 3 5
6 54 32 12 45 7 8	54 32 12 45 7 8 2 45 7
2 6 4	6 4 0
3 9 5 7	9 5 7 3 9 5 7

Answer: (penalty regime: 33.3, 66.7, ... %)

```c
#include <stdio.h>
void readArray(int A[], int n){
    if(n == 0){

    }
    else{
        readArray(A, n-1);
        scanf("%d", &A[n-1]);
    }
}
void xuatArray(int A[], int n){
    if(n == 0){

    }
    else{
        xuatArray(A, n-1);
        printf("%d ", A[n-1]);
        }
}
int main(){
    int n;
    scanf("%d", &n);
    int A[n-1];
    readArray(A, n);
    int i;  int count =0;
    xuatArray(A, n);
    for(i = 0; i <= n-1; i++){
        if(A[i] % 2 != 0){
            count++;
        }
    }
    printf("\n%d\n", count);
    for(i = 0; i <= n-1; i++){
        if(A[i] % 2 != 0){
            printf("%d ", A[i]);
        }
    }

    return 0;
}
```

	Input	Expected	Got	
✔	5 1 2 3 4 5	1 2 3 4 5 3 1 3 5	1 2 3 4 5 3 1 3 5	✔
✔	6 54 32 12 45 7 8	54 32 12 45 7 8 2 45 7	54 32 12 45 7 8 2 45 7	✔
✔	2 6 4	6 4 0	6 4 0	✔
✔	3 9 5 7	9 5 7 3 9 5 7	9 5 7 3 9 5 7	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00. Accounting for previous tries, this gives **0.00/1.00**.

Question **9**

Correct

Mark 0.67 out of 1.00

Viết chương trình nhập vào n số nguyên và sắp xếp các số nguyên đã nhập tăng dần

Đầu vào

- Dòng đầu tiên là n - số lượng các số nguyên ($0 < n \leq 1000$).
- Dòng thứ hai chứa n số nguyên, mỗi số cách nhau khoảng trắng

Đầu ra

- Dòng đầu tiên là n số nguyên đã nhập, mỗi số cách nhau khoảng trắng
- Dòng thứ 2 là n số nguyên đã được sắp xếp theo thứ tự tăng

Ghi chú:

- Dữ liệu đầu vào là hợp lệ.
- Bạn không nên dùng **printf** trước **scanf**.

Xem phần ví dụ để biết chi tiết về đầu vào/đầu ra.

For example:

Input	Result
5 1 20 30 4 5	1 20 30 4 5 1 4 5 20 30
6 54 32 12 45 7 8	54 32 12 45 7 8 7 8 12 32 45 54
2 6 4	6 4 4 6
3 9 5 7	9 5 7 5 7 9

Answer: (penalty regime: 33.3, 66.7, ... %)

```c
#include <stdio.h>
void readArray(int A[], int n){
    if(n==0){
    }
    else{
        readArray(A,n-1);
        scanf("%d",&A[n-1]);
    }
}
int main(){
    int n;
    scanf("%d", &n);
    int A[n-1];
    readArray(A, n);
    int i;
    for(i = 0; i <= n-1; i++){
        printf("%d ", A[i]);
    }
    printf("\n");
    int j, temp;
    for(i = 0; i <= n-1; i++){
        for(j = 0; j < i+1; j++){
            if(A[j] >= A[i+1]){
                temp = A[j];
                A[j] = A[i+1];
                A[i+1] = temp;
            }
        }
    }
    for(j = 0; j <= n-1; j++){
        printf("%d ", A[j]);
    }

    return 0;
}
```

	Input	Expected	Got	

	Input	Expected	Got	
✔	5 1 20 30 4 5	1 20 30 4 5 1 4 5 20 30	1 20 30 4 5 1 4 5 20 30	✔
✔	6 54 32 12 45 7 8	54 32 12 45 7 8 7 8 12 32 45 54	54 32 12 45 7 8 7 8 12 32 45 54	✔
✔	2 6 4	6 4 4 6	6 4 4 6	✔
✔	3 9 5 7	9 5 7 5 7 9	9 5 7 5 7 9	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00. Accounting for previous tries, this gives **0.67/1.00**.

Question **10**

Correct

Mark 0.00 out of 1.00

Viết 1 hàm cho phép kiểm tra n số nguyên đầu vào ($0 < n \leq 1000$) có được sắp xếp tăng dần hay không?

Khuôn dạng (Prototype)

- Tên hàm (function name): **isSort()**
- Tham số (parameters):
  - **A[]** - kiểu int
  - **n** - kiểu int
- Kiểu trả về (return type): **int**

Thân hàm (Body)

- Nếu mảng A tăng dần, kết quả của hàm là 1, ngược lại kết quả là 0

Chú ý

- Giá trị của các tham số luôn hợp lệ, không cần kiểm tra.
- Chỉ viết hàm, KHÔNG VIẾT TOÀN BỘ CHƯƠNG TRÌNH
- Xem thêm chi tiết trong phần **For example**.

For example:

Test	Result

`int A[]={-1,1,4, 5,10, 15};`
`int n = sizeof(A)/sizeof(int);`
`if (isSort(A,n))`
`  printf("YES");`
`else`
`  printf("NO");`
	YES

`int A[]={-1};`
`int n = sizeof(A)/sizeof(int);`
`printf("%d",isSort(A,n));`
	1

`int A[]={-1,-3, -1, 5,7};`
`int n = sizeof(A)/sizeof(int);`
`printf("%d",isSort(A,n));`
	0

`int A[]={-1,0,2,3,8,9,10,20,30,29};`
`int n = sizeof(A)/sizeof(int);`
`printf("%d",isSort(A,n));`
	0

`int A[]={-1,2,5,0,2,4,2,5};`
`int n = sizeof(A)/sizeof(int);`
`printf("%d",isSort(A,n));`
	0

Answer: (penalty regime: 33.3, 66.7, ... %)

```
int isSort(int A[], int n){
    int i,flag = 1;
    for(i=0; i<n-1; i++)
        if(A[i] > A[i+1]){
            flag = 0;
            break;
        }
    return flag;
}
```

	Test	Expected	Got	

✔	`int A[]={-1,1,4, 5,10, 15};`
	`int n = sizeof(A)/sizeof(int);`
	`if (isSort(A,n))`
	`  printf("YES");`
	`else`
	`  printf("NO");`
		YES	YES	✔

✔	`int A[]={-1};`
	`int n = sizeof(A)/sizeof(int);`
	`printf("%d",isSort(A,n));`
		1	1	✔

✔	`int A[]={-1,-3, -1, 5,7};`
	`int n = sizeof(A)/sizeof(int);`
	`printf("%d",isSort(A,n));`
		0	0	✔

✔	`int A[]={-1,0,2,3,8,9,10,20,30,29};`
	`int n = sizeof(A)/sizeof(int);`
	`printf("%d",isSort(A,n));`
		0	0	✔

✔	`int A[]={-1,2,5,0,2,4,2,5};`
	`int n = sizeof(A)/sizeof(int);`
	`printf("%d",isSort(A,n));`
		0	0	✔

Passed all tests! ✔

Correct

Marks for this submission: 1.00/1.00. Accounting for previous tries, this gives **0.00/1.00**.

◄ Hàm

Jump to...

Một số thao tác trên mảng 1 chiều ►