BABEŞ-BOLYAI UNIVERSITY
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

Admission Exam – July 19 2022

Written Exam for Computer Science

IMPORTANT NOTE:
Unless further clarification is provided, assume that arithmetical operations are performed over boundless data types (no *overflow* / *underflow*).
Furthermore, arrays and vectors are indexed starting from 1.

1. Let us consider the algorithm `ceFace(a, b)`, where ***a*** and ***b*** are natural numbers ($1 \leq a, b \leq 10000$ at the initial call).

```
Algorithm ceFace(a, b):
    While (a MOD 10 = b MOD 10) AND (a ≠ 0) AND (b ≠ 0) execute
        a ← a DIV 10
        b ← b DIV 10
    EndWhile
    If ((a = 0) AND (b = 0)) then
        return True
    else
        return False
    EndIf
EndAlgorithm
```

The algorithm `ceFace(a, b)` returns *True* if and only if:

A. ***a*** and ***b*** have the same number of digits
B. ***a*** and ***b*** are equal
C. ***a*** and ***b*** are written using the same digits, but in different sequence
D. the last digit of ***a*** is equal with the last digit of ***b***

2. Let us consider the algorithm `f(a, n)` where ***n*** is a natural number ($2 \leq n \leq 10000$) and ***a*** is an array of ***n*** natural numbers ($a[1], a[2], ..., a[n]$, $-100 \leq a[i] \leq 100$, for $i = 1, 2, ..., n$). The local variable ***b*** is an array.

```
Algorithm f(a, n):
    i ← 2
    b[1] ← a[1]
    While i ≤ n execute
        b[i] ← b[i - 1] + a[i]
        i ← i + 1
    EndWhile
    return b[n]
EndAlgorithm
```

Which of the following statements are true?

A. The algorithm returns the sum of all elements of array ***a***.
B. The algorithm returns the sum of the last two elements of array ***a***.
C. The algorithm returns the last element of array ***a***.
D. The algorithm returns the sum of the last $n - 1$ elements of array ***a***.

3. Which of the following algorithms returns the number of distinct prime factors of a given natural number ***n*** ($5 < n < 10^5$ at the initial call).

A.

```
// The length of array prime is n
// prime[i] is True, if
// the number i is prime and False
// otherwise
Algorithm nrFactoriPrimi_A(n, prime):
    d ← 2
    nr ← 0
    p ← 0
    While n > 0 execute
        While n MOD d = 0 execute
            p ← p + 1
            n ← n DIV d
        EndWhile
        If p ≠ 0 then
            nr ← nr + 1
        EndIf
        d ← d + 1
        While prime[d] = False execute
            d ← d + 1
        EndWhile
        p ← 0
    EndWhile
    return nr
EndAlgorithm
```

B.

```
Algorithm nrFactoriPrimi_B(n):
    d ← 2
    nr ← 0
    While n > 1 execute
        p ← 0
        While n MOD d = 0 execute
            p ← p + 1
            n ← n DIV d
        EndWhile
        If p > 0 then
            nr ← nr + 1
        EndIf
        If d = 2 then
            d ← d + 1
        else
            d ← d + 2
        EndIf
    EndWhile
    return nr
EndAlgorithm
```

C.

```
Algorithm nrFactoriPrimi_C(n):
    nr ← 0
    For d ← 2, n execute
        If n MOD d = 0 then
            nr ← nr + 1
        EndIf
        While n MOD d = 0 execute
            n ← n DIV d
        EndWhile
    EndFor
    return nr
EndAlgorithm
```

D.

```
Algorithm nrFactoriPrimi_D(n):
    nr ← 0
    d ← 2
    While d * d ≤ n execute
        If n MOD d = 0 then
            nr ← nr + 1
        EndIf
        While n MOD d = 0 execute
            n ← n DIV d
        EndWhile
        d ← d + 1
    EndWhile
    return nr
EndAlgorithm
```

4. Let us consider the algorithm `ceFace(n, m)`, where ***n*** is a natural number ($0 \le n \le 1000$) with the last digit not equal to 0.

```
Algorithm ceFace(n, m):
    If n = 0 then
        return m
    else
        return ceFace(n DIV 10, m * 10 + n MOD 10)
    EndIf
EndAlgorithm
```

What is the result of the call `ceFace(n, 0)`?

A. 0 (regardless of the value of ***n***)
B. ***n*** (regardless of the value of ***n***)
C. The sum of the digits of ***n***
D. The reverse of number ***n***

5. Let us consider the algorithm `f(x, n)` where n is a natural number ($2 \le n \le 10000$), and x is an array of n natural numbers ($x[1], x[2], ..., x[n]$, $1 \le x[i] \le 10000$, for $i = 1, 2, ..., n$).

```
Algorithm f(x, n):
    For i = 1, n – 1 execute
        If x[i] = x[i + 1] then
            return False
        EndIf
    EndFor
    return True
EndAlgorithm
```

Which of the following statements are true?

A. The algorithm returns *False* if two random elements of the array x are distinct.
B. The algorithm returns *False* if two random elements of the array x are equal.
C. The algorithm returns *False* if two consecutive elements of the array x are equal.
D. The algorithm returns *False* if the first two elements of the array x are equal.

6. Let us consider the algorithm `f(x, n)` where x and n are natural numbers ($0 \le n \le 10000$, $0 < x \le 10000$).

```
1. Algorithm f(x, n):
2.     If n = 0 then
3.         return 1
4.     EndIf
5.     m ← n DIV 2
6.     p ← f(x, m)
7.     If n MOD 2 = 0 then
8.         return p * p
9.     EndIf
10.    return x * p * p
11.EndAlgorithm
```

Which of the following statements are true?

A. The algorithm returns x at the power of n.
B. If on line 7, we replace `n MOD 2` with `m MOD 2`, then the algorithm returns x at the power of n.
C. Because of the recursive call on line 6, the lines 7, 8, 9, 10 will never be executed.
D. The algorithm returns 1 if n is an even number or it returns x if n is an odd number.

7. Considering that all multiplications and divisions require a constant amount of time, what can be stated about the time complexity of the algorithm considered in problem statement **6**?

A. The time complexity depends on parameters x and n.
B. The time complexity does not depend on the parameter x.
C. The time complexity is $O(\log \log n)$.
D. The time complexity is logarithmic based on the parameter n ($O(\log n)$).

8. Let us consider the algorithm `afișare(n)`, where n is a natural number ($1 \le n \le 10000$).

```
Algorithm afișare(n):
    If n ≤ 4000 then
        Write n, " "
        afișare(2 * n)
        Write n, " "
    EndIf
EndAlgorithm
```

What will be displayed for the call `afișare(1000)`?

A. 1000 2000 4000
B. 1000 2000 4000 4000 2000 1000
C. 1000 2000 4000 2000 1000
D. 1000 2000 2000 1000

9. Which could be the values of an array so that, applying the binary search method for value 36, it will be successively compared with values 12, 24, 36:

A. [2, 4, 7, 12, 24, 36, 50]
B. [2, 4, 8, 9, 12, 16, 20, 24, 36, 67]
C. [4, 8, 9, 12, 16, 24, 36]
D. [12, 24, 36, 42, 54, 66]

10. Which of the following mathematical expressions are equivalent to `x MOD y` for all strictly positive natural numbers ***x*** and ***y*** (0 < ***x, y*** ≤ 10000)?

A. `x DIV y`
B. `x – (y * (x DIV y))`
C. `x – (x * (x DIV y))`
D. `x DIV y + y DIV x`

11. Let us consider variable ***n*** that stores a natural number. Which of the following expressions is *True* if and only if ***n*** is divisible by 2 and by 3?

A. `(n DIV 2 = 0) OR (n DIV 3 ≠ 0)`
B. `(n MOD 3 = 2) OR (n MOD 2 = 3)`
C. `(n MOD 2 ≠ 1) AND (n MOD 3 = 0)`
D. `(n MOD 2 = 0) AND (n MOD 3 ≠ 1)`

12. Let us consider variable ***n*** that stores a natural number. Which of the following expressions is *True* if and only if ***n*** is divisible by 2 and by 3?

A. `(n MOD 2) - (n MOD 3) = 0`
B. `(n MOD 2) - (n MOD 3) < 0`
C. `(n MOD 2) + (n MOD 3) > 0`
D. `(n MOD 2) + (n MOD 3) = 0`

13. Let us consider the algorithm `f(n)`, where ***n*** is a natural number (1 ≤ ***n*** ≤ 100). The operator "/" stands for real division (ex. 3 / 2 = 1,5). State the effect of the algorithm.

```
Algorithm f(n):
    s ← 0; p ← 1;
    For i ← 1, n execute
        s ← s + i
        p ← p * (1 / s)
    EndFor
    return p
EndAlgorithm
```

A. Evaluates the expression 1/1 ∗ 1/2 ∗ 1/3 ∗ ... ∗ 1/***n***
B. Evaluates the expression 1/1 ∗ 1/(1∗2) ∗ 1/(1∗2∗3) ∗ ... ∗ 1/(1∗2∗3∗…∗***n***)
C. Evaluates the expression 1/1 ∗ 1/(1+2) ∗ 1/(1+2+3) ∗ ... ∗ 1/(1+2+3+…+***n***)
D. Evaluates the expression 1/1 + 1/(1∗2) + 1/(1∗2∗3) + ... + 1/(1∗2∗3∗…∗***n***)

14. Let us consider the algorithm `prelucrare(s1, lung1, s2, lung2)`, where ***s*1** and ***s*2** are two arrays of characters of length ***lung*1**, respectively ***lung*2** (1 ≤ ***lung*1**, ***lung*2** ≤ 1000). The two strings contain only characters having ASCII codes in the interval [1, 125]. The local variable ***x*** is an array. Let us consider the algorithm `ascii(s, i)`, which returns the ASCII code of the ***i***-th character of array ***s***.

```
Algorithm prelucrare(s1, lung1, s2, lung2):
    For i = 1, 125 execute
        x[i] ← 0
    EndFor
    For i = 1, lung1 execute
        x[ascii(s1, i)] ← x[ascii(s1, i)] + 1
    EndFor
    For i = 1, lung2 execute
        x[ascii(s2, i)] ← x[ascii(s2, i)] - 1
    EndFor
    ok ← True
    For i = 1, 125 execute
        If x[i] ≠ 0 then
            ok ← False
        EndIf
    EndFor
    return ok
EndAlgorithm
```

What is the result of the algorithm?

A. The algorithm returns *True* if the arrays of characters ***s*1** and ***s*2** have the same length and *False* otherwise.
B. The algorithm returns *True* if the arrays of characters ***s*1** and ***s*2** contain the same characters having the same corresponding frequency and *False* otherwise.
C. The algorithm returns *True* if in both arrays of characters ***s*1** and ***s*2** all characters having ASCII codes in the interval [1, 125] appear and *False* otherwise.
D. The algorithm returns *True* if the two arrays of characters ***s*1** and ***s*2** use different characters and *False* otherwise.

15. What is the result of converting the binary number 100101100111 into base 10?
A. 2407 B. 2408 C. 1203 D. None of the answers A., B., C.

16. Let us consider an array ***a*** of ***n*** natural numbers (***a***[1], ***a***[2], …, ***a***[***n***]), the natural number ***n*** ($1 \le n \le 10000$) and a natural number ***x***. Which of the following code sequences display the position having the minimal index where the value ***x*** is situated in the array ***a***, or displays -1 if ***x*** is not found in ***a***?

A.
```
i ← 1
While (i ≤ n) AND (a[i] = x) execute
    i ← i + 1
EndWhile
If i ≤ n then
    Write i
else
    Write -1
EndIf
```

B.
```
i ← 1
While (i ≤ n) AND (a[i] ≠ x) execute
    i ← i + 1
EndWhile
If i = n + 1 then
    Write i
else
    Write -1
EndIf
```

C.
```
i ← 1
While (i ≤ n) AND (a[i] = x) execute
    i ← i + 1
EndWhile
If i = n + 1 then
    Write i
else
    Write -1
EndIf
```

D.
```
i ← 1
While (i ≤ n) AND (a[i] ≠ x) execute
    i ← i + 1
EndWhile
If i ≤ n then
    Write i
else
    Write -1
EndIf
```

17. Let us consider the algorithm `f(x)`, where ***x*** is an integer:

```
Algorithm f(x):
    If x = 0 then
        return 0
    else
        If x MOD 3 = 0 then
            return f(x DIV 10) + 1
        else
            return f(x DIV 10)
        EndIf
    EndIf
EndAlgorithm
```

For which value of ***x*** does the algorithm return the value 4?

A. 13369 B. 21369 C. 4 D. 1233

18. Let us consider the algorithm `f(n, i, j)` where ***n***, ***i*** and ***j*** are natural numbers (1 ≤ ***n, i, j*** ≤ 10000 at the initial call).

```
Algorithm f(n, i, j):
    If i > j then
        Write '*'
    else
        If n MOD i = 0 then
            f(n, i - 1, j)
        else
            If n DIV i ≠ j then
                f(n, i + 1, j - 1)
                Write '0'
            else
                f(n, i + 2, j - 2)
                Write '#'
            EndIf
        EndIf
    EndIf
EndAlgorithm
```

What will be displayed upon the execution of the call `f(15, 3, 10)`?

A. *000000
B. *0#000
C. *0#0000
D. *0000000

19. Let us consider algorithm `ceFace(n, x)`, where ***n*** is a natural number (1 ≤ ***n*** ≤ 100) and ***x*** is an array of ***n*** natural numbers (***x***[1], ***x***[2], …, ***x***[***n***]).

```
Algorithm ceFace(n, x):
    For i = 1, n execute
        c ← x[i]
        x[i] ← x[n – i + 1]
        x[n – i + 1] ← c
    EndFor
EndAlgorithm
```

What will be the new content of array ***x*** after executing the algorithm if ***n*** = 6 and ***x*** = [5, 3, 2, 1, 1, 1]?

A. [1, 1, 2, 1, 3, 5]   B. [1, 1, 1, 2, 3, 5]   C. [5, 3, 2, 1, 1, 1]
D. None of the other options is correct.

20. Let us consider the algorithm `what(n)`, where ***n*** is a natural number ($1 \leq n \leq 1000$ at the initial call).

```
Algorithm what(n):
    If n = 0 then
        return True
    EndIf
    If (n MOD 10 = 3) OR (n MOD 10 = 7) then
        return what(n DIV 10)
    else
        return False
    EndIf
EndAlgorithm
```

Which of the following statements are true?

A. The algorithm returns *True* if and only if either ***n*** can be written using only the digit 3, or ***n*** can be written using only the digit 7
B. The algorithm returns *False* if ***n*** contains at least an even digit
C. The algorithm returns *False* if and only if ***n*** contains at least one digit ***c*** where $c \neq 3$ and $c \neq 7$
D. The algorithm returns *True* if and only if ***n*** does not contain any digit from the set {0, 1, 2, 4, 5, 6, 8, 9}

`

21. Let us consider the algorithm `calcul(x, n)`, where ***x*** and ***n*** are natural numbers ($1 \leq x \leq 10000$, $1 \leq n \leq 10000$), and $x \leq n$.

```
Algorithm calcul(x, n):
    b ← 1
    For i ← 1, n – x execute
        b ← b * i
    EndFor
    a ← b
    For i ← n - x + 1, n execute
        a ← a * i
    EndFor
    return a DIV b
EndAlgorithm
```

Which of the following statements are true?

A. If $x = 2$ and $n = 5$, then the algorithm returns 10.
B. The algorithm returns the number of subsets having ***x*** elements from the set $\{1, 2, \ldots, n\}$.
C. The algorithm returns the number of partial permutations of ***n*** elements taken ***x*** at a time.
D. The algorithm returns the number of combinations of ***n*** elements taken ***x*** at a time.

22. At a farm there are chickens and rabbits, each chicken has two legs and each rabbit has four legs. The total number of heads is ***n*** and the total number of legs is ***m*** ($0 \leq n, m \leq 10^4$). Which of the following algorithms returns *True* and displays all possible pairs of numbers of chickens and rabbits at the farm, or returns *False* if there are no solutions?

A.

```
Algorithm ferma_A(n, m):
    found = False
    For i ← 0, n execute
        j ← n - i
        If 2 * i + 4 * j = m then
            found ← True
            Write i, ' ', j
            Write newline
        EndIf
    EndFor
    return found
EndAlgorithm
```

B.

```
Algorithm ferma_B(n, m):
    found ← False
    For i ← 0, n execute
        For j ← 0, n execute
            If 2 * i + 4 * j = m AND
                                 i + j = n then
                found ← True
                Write i, ' ', j
                Write newline
            EndIf
        EndFor
    EndFor
    return found
EndAlgorithm
```

C.

```
Algorithm ferma_C(n, m):
    found ← False
    For i ← 0, n execute
        For j ← 0, n - i execute
            If 2 * i + 4 * j = m AND
                           i + j = n then
                found ← True
                Write i, ' ', j
                Write newline
            EndIf
        EndFor
    EndFor
    return found
EndAlgorithm
```

D.

```
Algorithm ferma_D(n, m):
    found ← False
    For i ← 0, n execute
        For j ← 0, i execute
            If 2 * i + 4 * j = m AND
                                 i + j = n then
                found ← True
                Write i, ' ', j
                Write newline
            EndIf
        EndFor
    EndFor
    return found
EndAlgorithm
```

23. Let us consider a natural number ***n***, which can be written as the product of three natural numbers ***a***, ***b***, ***c***, (***n*** = ***a*** * ***b*** * ***c***). Which of the following expressions has as result the remainder of the division of ***n*** by the natural number ***d*** ($1 \leq n, a, b, c, d \leq 10000$)?

A. `(a MOD d) * b * c`
B. `((a MOD d) * (b MOD d) * (c MOD d)) MOD d`
C. `(a MOD d) * (b MOD d) * (c MOD d)`
D. `(a DIV d) * (b DIV d) * (c DIV d)`

24. Let us consider the algorithm `det(a, n, m)`, where ***a*** is an array of ***n*** natural numbers (***a***[1], ***a***[2], ..., ***a***[***n***] if $n \geq 1$) or an empty array if ***n*** = 0. ***n*** and ***m*** are natural numbers ($0 \leq n \leq 100$, $0 \leq m \leq 10^6$).

```
Algorithm det(a, n, m):
    For i ← 1, n – 1 execute
        For j ← i + 1, n execute
            If a[i] > a[j] then
                tmp ← a[i]
                a[i] ← a[j]
                a[j] ← tmp
            EndIf
        EndFor
    EndFor
    i ← 1
    j ← n
    b ← False
    While i < j execute
        If a[i] + a[j] = m then
            b ← True
        EndIf
```

```
18.         If a[i] + a[j] < m then
19.             i ← i + 1
20.         else
21.             j ← j - 1
22.         EndIf
23.     EndWhile
24.     return b
25. EndAlgorithm
```

Which of the following statements are true?

A. The algorithm returns *True* if array ***a*** contains a pair of numbers having their sum equal to ***m***.
B. The algorithm always returns *False*.
C. The algorithm returns *False* if ***n*** = 0.
D. Lines 2, ..., 10 of the algorithm sort array ***a*** in ascending order.

25. Let us consider the algorithm `magic(n, a)`, where ***a*** is an array of ***n*** natural numbers (***a***[1], ***a***[2], ..., ***a***[***n***], $1 \leq n \leq 10000$).

```
Algorithm magic(n, a):
    If n < 2 then
        return False
    EndIf
    For i ← 2, n execute
        If a[i – 1] = a[i] then
            return True
        EndIf
    EndFor
    return False
EndAlgorithm
```

Which of the following statements are true?

A. For `magic(5, [2, 5, 4, 5, 4])` the algorithm returns *False*.
B. The algorithm indicates if there are duplicates in the array ***a***, if and only if array ***a*** is sorted ascending/descending.
C. For `magic(9, [1, 2, 3, 4, 4, 5, 6, 7, 9])` the algorithm returns *True*.
D. For `magic(5, [9, 5, 5, 2, 4])` the algorithm returns *True*.

26. Let us consider the algorithm `f(n, a, b, c)` where ***n*** is a natural number ($n \leq 20$) and ***a***, ***b***, ***c*** three integer numbers.

```
Algorithm f(n, a, b, c):
    If n = 0 then
        return 1
    else
        return f(n - 1, a * a, b + 1, c * 2) + f(n - 1, a - 1, b * b, c + 1) + 1
    EndIf
EndAlgorithm
```

What is the return value of the call `f(n, 1, 1, 2)`?

A. $2^{n+1} - 1$
B. n
C. $2^0 + 2^1 + 2^2 + ... + 2^n$
D. 2^{n+1}

27. Let us consider the algorithms `f(n, p)` and `g(n)`, where ***n*** and ***p*** are initially natural numbers ($1 \leq$ ***n, p*** $\leq 10^6$ at the initial call).

```
Algorithm g(n):
    If n < 2 then
        return False
    EndIf
    i ← 2
    While i * i ≤ n execute
        If n MOD i = 0 then
            return False
        EndIf
        i ← i + 1
    EndWhile
    return True
EndAlgorithm
```

```
Algorithm f(n, p):
    If n = 0 then
        return 1
    EndIf
    If n > 0 AND n ≥ p then
        c ← 0
        If g(p) = True then
            c ← c + f(n - p, p + 1)
        EndIf
        return c + f(n, p + 1)
    EndIf
    return 0
EndAlgorithm
```

Which of the following statements are true?

A. The algorithm `g(n)` returns *True* if ***n*** is prime and *False* otherwise.
B. The call `f(n, 2)` returns the number of distinct ways of writing ***n*** as a sum of at least one term of distinct prime numbers in strictly ascending order.
C. The call `f(n, 2)` returns the sum of the prime divisors of ***n***.
D. The calls `f(n, 1)` and `f(n, 2)` will return the same result, regardless of ***n***.

28. Let us consider the algorithm `AlexB(value, n, k, p)`, where ***value*** is an array of ***n*** natural numbers (***value***[1], ***value***[2], ..., ***value***[***n***]), and ***n***, ***k*** and ***p*** are natural numbers. Initially the array ***value*** has ***n*** elements equal to zero. The algorithm `afișare(value, n)` displays the array ***value*** on a single line.

```
Algorithm AlexB(value, n, k, p):
    p ← p + 1
    value[k] ← p
    If p = n then
        afișare(value, n)
    else
        For i ← 1, n execute
            If value[i] = 0 then
                AlexB(value, n, i, p)
            EndIf
        EndFor
    EndIf
    p ← p - 1
    value[k] ← 0
EndAlgorithm
```

What will be displayed on the 10th line, if ***n*** = 5 and the algorithm is called like: `AlexB(value, 5, 1, 0)`.

A. 1 5 2 3 4
B. 1 5 4 0 4
C. 5 5 5 5 5
D. 1 2 5 4 3

29. Let us consider the algorithm `f(n)` where ***n*** is a natural number ($1 \leq$ ***n*** ≤ 10000 at the initial call).

```
Algorithm f(n):
    c ← 0
    If n ≠ 0 then
        c ← c + 1
        n ← n & (n - 1)       // bitwise AND
        While n ≠ 0 execute
            c ← c + 1
            n ← n & (n - 1)   // bitwise AND
        EndWhile
    EndIf
    return c
EndAlgoritm
```

The & operator is the bitwise AND operator; the truth table is:

&	0	1
0	0	0
1	0	1

Example:
2 & 7 in binary: 010 & 111 = 010 which is 2 in base 10.
6 & 1 in binary: 110 & 001 = 000 which is 0 in base 10.

Which of the following statements are **NOT** true?

A. If ***n*** is a power of 2, then `f(n)` returns value 1.
B. If ***n*** >16 and ***n*** < 32, then `f(n)` returns a value from the {2, 3, 4, 5} set.
C. The algorithm returns the number of even numbers strictly smaller than ***n***.
D. The algorithm returns the number of odd numbers smaller than ***n.***

30. Let us consider algorithm `calcul(v, n)`, where ***n*** is a non zero natural number ($1 \leq n \leq 10000$) and ***v*** is an array of ***n*** integer numbers (***v***[1], ***v***[2], ..., ***v***[***n***]). The instruction **`return`** `x, y` returns the pair of values (***x***, ***y***).

```
Algorithm calcul(v, n):
    i ← n DIV 2 + 1
    j ← i + 1
    k ← i
    p ← j
    While j ≤ n execute
        While (j ≤ n) AND (v[i] = v[j]) execute
            j ← j + 1
        EndWhile
        If j - i > p - k then
            k ← i
            p ← j
        EndIf
        i ← j
        j ← j + 1
    EndWhile
    If j - i > p - k then
        k ← i
        p ← j
    EndIf
    return p - k, k
EndAlgorithm
```

Which of the following statements are true?

A. If the array contains only one element, the algorithm returns 0, -1
B. If ***n*** = 2 and the array's two elements are symmetric with respect to 0 (for example -5, 5), the result will be -1, 1
C. If ***n*** = 2 and the array's two elements have consecutive values (for example 3, 4), the algorithm will always return the values 1, 2
D. One of the numbers returned by the algorithm represents the length of the longest sequence containing equal values from the second half of the array for any even number ***n*** > 1