BABEŞ-BOLYAI UNIVERSITÄT
FAKULTÄT FÜR MATHEMATIK UND INFORMATIK

Zulassungswettbewerb - 19. Juli 2022

Schriftliche Prüfung in Informatik

WICHTIGER HINWEIS:
Falls nicht anders angegeben ist davon auszugehen, dass alle arithmetischen Operationen mit unbegrenzten Datentypen durchgeführt werden (kein *overflow* / *underflow*). Außerdem beginnt die Indexnummerierung aller Zeichenfolgen bei 1.

1. Gegeben sei der Algorithmus `ceFace(a, b)`, wobei ***a*** und ***b*** natürliche Zahlen sind ($1 \leq \boldsymbol{a}, \boldsymbol{b} \leq 10000$ zum Zeitpunkt des Aufrufs).

```
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
```

Der Algorithmus `ceFace(a, b)` gibt *True* zurück, wenn und nur wenn:

A. die Zahlen ***a*** und ***b*** die gleiche Anzahl von Ziffern haben
B. ***a*** und **b** gleich sind
C. ***a*** und ***b*** aus denselben Ziffern bestehen, aber in einer anderen Reihenfolge
D. die letzte Ziffer von ***a*** gleich ist der letzten Ziffer von ***b***

2. Gegeben sei der Algorithmus `f(a, n)`, wobei ***n*** eine natürliche von Null verschiedene Zahl ($2 \leq \boldsymbol{n} \leq 10000$) ist und ***a*** ein Vektor mit ***n*** ganzen Zahlen ($\boldsymbol{a}[1], \boldsymbol{a}[2], ..., \boldsymbol{a}[\boldsymbol{n}]$, $-100 \leq a[\boldsymbol{i}] \leq 100$, für $\boldsymbol{i} = 1, 2, ..., \boldsymbol{n}$). Die lokale Variable ***b*** ist ein Vektor.

```
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
```

Geben Sie an welche der folgenden Aussagen wahr sind:

A. Der Algorithmus liefert die Summe aller Elemente des Vektors ***a***.
B. Der Algorithmus gibt die Summe der letzten beiden Elemente des Vektors ***a*** zurück.
C. Der Algorithmus gibt das letzte Element des Vektors ***a*** zurück.
D. Der Algorithmus liefert die Summe der letzten ***n*** **-** 1 Elemente des Vektors ***a***.

3. Welcher der folgenden Algorithmen liefert die Anzahl der verschiedenen Primfaktoren einer gegebenen natürlichen Zahl ***n*** ($5 < n < 10^5$ zum Zeitpunkt des Aufrufs).

A.

```
// Der Vektor prime hat die Länge n
// prime[i] ist Wahr, wenn die Zahl i
// eine Primzahl ist und sonst Falsch
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. Gegeben sei der Algorithmus `ceFace(n, m)`, wobei ***n*** eine natürliche Zahl ($0 \leq n \leq 1000$) ist, deren letzte Ziffer von 0 verschieden ist.

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

Was ist das Ergebnis des Aufrufs von `ceFace(n, 0)`?

A. 0 (unabhängig vom Wert von ***n***)
B. ***n*** (unabhängig vom Wert von ***n***)
C. Die Summe der Ziffern der Zahl ***n***
D. Das Spiegelbild der Zahl ***n.***

5. Gegeben sei der Algorithmus `f(x, n)`, wobei n eine natürliche Zahl ist ($2 \leq n \leq 10000$) und x eine Folge von n natürlichen Zahlen ($x[1], x[2], ..., x[n]$, $1 \leq x[i] \leq 10000$, für $i = 1, 2, ..., n$).

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

Geben Sie an, welche der folgenden Aussagen wahr sind:

A. Der Algorithmus gibt *False* zurück, wenn zwei beliebige Elemente in der Zeichenfolge x unterschiedlich sind.
B. Der Algorithmus gibt *False* zurück, wenn zwei beliebige Elemente in der Zeichenfolge x gleich sind.
C. Der Algorithmus gibt *False* zurück, wenn zwei aufeinanderfolgende Elemente in der Zeichenfolge x gleich sind.
D. Der Algorithmus gibt *False* zurück, wenn die ersten beiden Elemente der Zeichenfolge x gleich sind.

6. Gegeben sei der Algorithmus `f(x, n)`, wobei x und n natürliche Zahlen sind ($0 \leq n \leq 10000$, $0 < x \leq 10000$).

```
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
```

Geben Sie an, welche der folgenden Aussagen wahr sind:

A. Der Algorithmus liefert x hoch n.
B. Wenn in Zeile 7 anstelle von `n MOD 2` der Ausdruck `m MOD 2` erscheinen würde, würde der Algorithmus x hoch n zurückgeben.
C. Wegen des Selbstaufrufs in Zeile 6 werden die Zeilen 7, 8, 9 und 10 nie ausgeführt.
D. Der Algorithmus liefert 1, wenn n eine gerade Zahl ist, oder x, wenn n eine ungerade Zahl ist.

7. Unter der Annahme, dass alle Multiplikations- und Divisionsoperationen in konstanter Zeit durchgeführt werden, was können wir über die Zeitkomplexität des Algorithmus in der Aufgabe **6** sagen?

A. Die Zeitkomplexität hängt von den Parametern x und n ab.
B. Die Zeitkomplexität hängt nicht von dem Parameter x ab.
C. Die Zeitkomplexität ist $O(\log \log n)$.
D. Die Zeitkomplexität ist logarithmisch in Bezug auf den Parameter n ($O(\log n)$).

8. Gegeben sei der Algorithmus `afișare(n)`, wobei n eine natürliche Zahl ist ($1 \leq n \leq 10000$).

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

Was wird beim Aufruf von `afișare(1000)` angezeigt?

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

9. Was könnten die Elemente eines Vektors sein, so dass bei Anwendung der binären Suchmethode für den Wert 36, dieser nacheinander mit den Werten 12, 24, 36 verglichen wird:
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. Welche der folgenden Ausdrücke sind äquivalent zu `x MOD y` für alle streng positiven natürlichen Zahlen ***x*** und ***y*** ($0 < x, y \leq 10000$)?
A. `x DIV y`
B. `x – (y * (x DIV y))`
C. `x – (x * (x DIV y))`
D. `x DIV y + y DIV x`

11. Gegeben sei die Variabel ***n***, die eine natürliche Zahl speichert. Welcher der folgenden Ausdrücke hat den Wahrheitswert *True* genau dann, wenn ***n*** durch 2 und 3 teilbar ist?

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. Gegeben sei die Variabel ***n***, die eine natürliche Zahl speichert. Welcher der folgenden Ausdrücke hat den Wahrheitswert *True* genau dann, wenn ***n*** durch 2 und 3 teilbar ist?
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. Gegeben sei der Algorithmus `f(n)`, wobei ***n*** eine natürliche Zahl ist ($1 \leq n \leq 100$). Der Operator "/" steht für die reelle Division (z. B. 3 / 2 = 1,5). Geben Sie die Wirkung des Algorithmus an.

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

A. Bewertet den Ausdruck 1/1 ∗ 1/2 ∗ 1/3 ∗ ... ∗ 1/***n***
B. Bewertet den Ausdruck 1/1 ∗ 1/(1∗2) ∗ 1/(1∗2∗3) ∗ ... ∗ 1/(1∗2∗3∗…∗***n***)
C. Bewertet den Ausdruck 1/1 ∗ 1/(1+2) ∗ 1/(1+2+3) ∗ ... ∗ 1/(1+2+3+…+***n***)
D. Bewertet den Ausdruck 1/1 + 1/(1∗2) + 1/(1∗2∗3) + ... + 1/(1∗2∗3∗…∗***n***)

14. Gegeben sei der Algorithmus `prelucrare(s1, lung1, s2, lung2)`, wobei ***s1*** und ***s2*** zwei Zeichenketten der Länge ***lung1*** bzw. ***lung2*** sind ($1 \leq$ ***lung1***, ***lung2*** ≤ 1000). Die beiden Zeichenketten enthalten nur Zeichen mit einem ASCII-Code im Bereich [1, 125]. Die lokale Variable ***x*** ist ein Vektor. Gegeben sei der Algorithmus `ascii(s, i)`, die den ASCII-Code des ***i***-ten Zeichens der Zeichenkette ***s*** zurückgibt.

```
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
```

Geben Sie die Wirkung des Algorithmus an.

A. Der Algorithmus gibt *True* zurück, wenn die Zeichenketten ***s1*** und ***s2*** die gleiche Länge haben, und andernfalls *False*.
B. Der Algorithmus gibt *True* zurück, wenn die Zeichenketten ***s1*** und ***s2*** aus denselben Zeichen mit denselben entsprechenden Häufigkeiten bestehen, und ansonsten *False*.
C. Der Algorithmus gibt *True* zurück, wenn jede der beiden Zeichenketten ***s1*** und ***s2*** alle Zeichen mit ASCII-Code im Bereich [1, 125] enthält, und ansonsten *False*.
D. Der Algorithmus gibt *True* zurück, wenn die beiden Zeichenketten ***s1*** und ***s2*** aus unterschiedlichen Zeichen bestehen, und ansonsten *False*.

15. Was ist das Ergebnis der Umwandlung der Binärzahl 100101100111 zur Basis 10?

A. 2407 B. 2408 C. 1203 D. Keine der Antworten A., B., C.

16. Man betrachte einen Vektor ***a*** mit ***n*** natürlichen Zahlen (***a***[1], ***a***[2], ..., ***a***[n]), ***n*** eine natürliche Zahl ($1 \leq$ ***n*** ≤ 10000) und ***x*** eine natürliche Zahl. Welche der folgenden Codesequenzen zeigt die Stelle mit dem kleinsten Index an, an der der Wert ***x*** im Vektor ***a*** vorkommt, oder zeigt -1 an, wenn ***x*** nicht im Vektor ***a*** vorkommt?

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. Gegeben sei der Algorithmus `f(x)`, mit ***x*** eine ganze Zahl:

```
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
```

Für welchen Wert von ***x*** wird der Algorithmus den Wert 4 zurückgeben?

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

18. Gegeben sei der Algorithmus `f(n, i, j)` mit ***n***, ***i*** und ***j*** natürliche Zahlen (1 ≤ ***n, i, j*** ≤ 10000 zum Zeitpunkt des Aufrufs).

```
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
```

Was wird nach der Ausführung des Aufrufs `f(15, 3, 10)` angezeigt?

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

19. Gegeben sei der Algorithmus `ceFace(n, x)`, wobei ***n*** eine natürliche Zahl (1 ≤ ***n*** ≤ 100) und ***x*** ein Vektor mit ***n*** Elemente natürliche Zahlen (***x***[1], ***x***[2], …, ***x***[***n***]) ist.

```
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
```

Wie lautet der neue Inhalt des Vektors ***x*** nach der Ausführung des angegebenen Algorithmus, wenn ***n*** = 6 und ***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. Keine der obigen Varianten ist richtig.

20. Gegeben sei der Algorithmus `what(n)`, wobei ***n*** eine natürliche Zahl ($1 \leq \boldsymbol{n} \leq 1000$ zum Zeitpunkt des Aufrufs) ist.

```
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
```

Geben Sie an, welche der folgenden Aussagen wahr sind:

A. Der Algorithmus liefert *True* genau dann, wenn entweder ***n*** nur aus Ziffern mit dem Wert 3 oder ***n*** nur aus Ziffern mit dem Wert 7 besteht
B. Der Algorithmus liefert *False*, wenn ***n*** mindestens eine gerade Ziffer enthält
C. Der Algorithmus liefert *False* genau dann, wenn ***n*** mindestens eine Ziffer ***c*** enthält, wobei $\boldsymbol{c} \neq 3$ und $\boldsymbol{c} \neq 7$
D. Der Algorithmus liefert *True* genau dann, wenn ***n*** keine Ziffern aus der Menge {0, 1, 2, 4, 5, 6, 8, 9} enthält.

21. Gegeben sei der Algorithmus `calcul(x, n)`, mit ***x*** und ***n*** natürliche Zahlen ($1 \leq \boldsymbol{x} \leq 10000$, $1 \leq \boldsymbol{n} \leq 10000$) und $\boldsymbol{x} \leq \boldsymbol{n}$.

```
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
```

Geben Sie an, welche der folgenden Aussagen wahr sind:

A. Wenn $\boldsymbol{x} = 2$ und $\boldsymbol{n} = 5$ ist, dann liefert der Algorithmus 10.
B. Der Algorithmus liefert die Anzahl der Untermengen der Menge {1, 2, ..., ***n***}, die ***x*** Elemente haben.
C. Der Algorithmus gibt die Anzahl der geordneten Stichproben von ***x*** Objekten aus einer Menge mit ***n*** Elementen zurück (A_n^x).
D. Der Algorithmus gibt die Anzahl der ungeordneten Stichproben von ***x*** Objekten aus einer Menge mit ***n*** Elementen zurück (C_n^x).

22. Auf einem Bauernhof werden Hühner und Kaninchen gezüchtet, wobei jedes Huhn zwei Beine und jedes Kaninchen vier Beine hat. Die Gesamtzahl der Köpfe ist ***n*** und die Gesamtzahl der Beine der Tiere auf dem Bauernhof ist ***m*** ($0 \leq \boldsymbol{n}, \boldsymbol{m} \leq 10^4$). Welcher der folgenden Algorithmen liefert *True* und zeigt alle möglichen Zahlenpaare für die Anzahl der Hühner und Kaninchen auf dem Bauernhof an, oder liefert *False*, wenn es keine Lösung gibt?

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. Gegeben sei eine natürliche Zahl ***n***, die als Produkt der drei natürlichen Zahlen ***a, b, c*** geschrieben werden kann (***n = a * b * c***). Welcher der folgenden Ausdrücke hat als Wert den Rest der Division von ***n*** durch die natürliche Zahl ***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. Gegeben sei der Algorithmus `det(a, n, m)`, wobei ***a*** eine Folge von ***n*** natürlichen Zahlen ist (***a***[1], ***a***[2], ..., ***a***[***n***], wenn $n \geq 1$) oder eine leere Folge, wenn $n = 0$. ***n*** und ***m*** sind natürliche Zahlen ($0 \leq n \leq 100$, $0 \leq m \leq 10^6$).

```
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
```

Geben Sie an, welche der folgenden Aussagen wahr sind:

A. Der Algorithmus gibt *True* zurück, wenn es in der Zeichenkette ***a*** ein Zahlenpaar gibt, dessen Summe gleich ***m*** ist.
B. Der Algorithmus gibt immer *False* zurück.
C. Der Algorithmus gibt *False* zurück, wenn ***n*** = 0 ist.
D. In den Zeilen 2, ..., 10 sortiert der Algorithmus die Zeichenfolge ***a*** in aufsteigender Reihenfolge.

25. Gegeben sei der Algorithmus `magic(n, a)`, wobei ***a*** ein Vektor bestehend aus ***n*** natürliche Zahlen (***a***[1], ***a***[2], ..., ***a***[***n***], $1 \leq n \leq 10000$) ist.

```
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
```

Geben Sie an, welche der folgenden Aussagen wahr sind:

A. Für `magic(5, [2, 5, 4, 5, 4])` liefert der Algorithmus *False*.
B. Der Algorithmus zeigt an, ob es Duplikate in der Zeichenkette ***a*** gibt genau dann, wenn der Vektor ***a*** auf- oder absteigend sortiert ist.
C. Für `magic(9, [1, 2, 3, 4, 5, 6, 7, 9])` liefert der Algorithmus *True*.
D. Für `magic(5, [9, 5, 5, 2, 4])` liefert der Algorithmus *True*.

26. Gegeben sei der Algorithmus `f(n, a, b, c)`, wobei ***n*** eine natürliche Zahl ($n \leq 20$) und ***a***, ***b***, ***c*** drei ganze Zahlen sind.

```
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
```

Was ist das Ergebnis des Aufrufs `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. Gegeben seien die Algorithmen `f(n, p)` und `g(n)`, wobei ***n*** und ***p*** initial natürliche Zahlen ($1 \leq$ ***n***, ***p*** $\leq 10^6$ zum Zeitpunkt des Aufrufs) sind.

```
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
```

Geben Sie an, welche der folgenden Aussagen wahr sind:

A. Der Algorithmus `g(n)` liefert *True*, wenn die Zahl ***n*** eine Primzahl ist, und *False*, wenn nicht.
B. Der Aufruf `f(n, 2)` gibt die Anzahl der verschiedenen Möglichkeiten zurück, in der die Zahl ***n*** als Summe von mindestens einem Term verschiedener Primzahlen in streng aufsteigender Reihenfolge geschrieben werden kann.
C. Bei dem Aufruf `f(n, 2)` wird die Summe der Primteiler der Zahl ***n*** zurückgegeben.
D. Die Aufrufe `f(n, 1)` und `f(n, 2)` liefern dasselbe Ergebnis, für jedwelchen Wert ***n***.

28. Gegeben sei der Algorithmus `AlexB(value, n, k, p)`, wobei ***value*** eine Folge von ***n*** natürlichen Zahlen ist (***value***[1], ***value***[2], ..., ***value***[n]) und ***n***, ***k*** und ***p*** natürliche Zahlen sind. Anfänglich hat die Zeichenkette ***value n*** Elemente gleich Null. Der Algorithmus `afișare(value, n)` zeigt auf einer Zeile die Zeichenkette ***value*** an.

```
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
```

Geben Sie die Zeichenfolge in der zehnten Zeile an, wenn ***n*** = 5 und der Algorithmus mit `AlexB(value, 5, 1, 0)` aufgerufen wird.

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. Gegeben sei der Algorithmus `f(n)` mit ***n*** eine natürliche Zahl ($1 \leq$ ***n*** ≤ 10000 zum Zeitpunkt des Aufrufs).

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

Der Operator & ist der Operator AND auf Bits; die Wahrheitstabelle ist die folgende:

&	0	1
0	0	0
1	0	1

Beispiel:

2 & 7 umgerechnet in Binärzahlen: 010 & 111 = 010, was 2 zur Basis 10 entspricht.

6 & 1 umgewandelt in Binärzahlen: 110 & 001 = 000, was 0 zur Basis 10 ist.

Geben Sie an, welche der folgenden Aussagen **NICHT** wahr sind:

A. Wenn ***n*** eine Potenz von 2 ist, dann liefert `f(n)` den Wert 1.
B. Wenn ***n*** >16 und ***n*** < 32 ist, dann gehört der von f(n) zurückgegebene Wert zu der Menge {2, 3, 4, 5}.
C. Der Algorithmus liefert die Anzahl der geraden Zahlen, die streng kleiner als ***n*** sind.
D. Der Algorithmus gibt die Anzahl der ungeraden Zahlen kleiner als ***n*** zurück.

30. Gegeben sei der Algorithmus `calcul(v, n)`, wobei ***n*** eine natürliche von Null verschiedene Zahl ist (1 ≤ ***n*** ≤ 10000) und ***v*** eine Folge von ***n*** ganzen Zahlen ist (***v***[1], ***v***[2], ..., ***v***[***n***]). Die Anweisung **`return`** `x, y` gibt das Wertepaar (***x, y***) zurück.

```
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
```

Geben Sie an, welche der folgenden Aussagen wahr sind:

A. Wenn die Zeichenkette nur ein Element hat, gibt der Algorithmus die Werte 0, -1 zurück.
B. Wenn ***n*** = 2 ist und die beiden Elemente der Folge symmetrisch zu 0 sind (z. B. -5, 5), dann ist das Ergebnis -1, 1.
C. Wenn ***n*** = 2 ist und die beiden Elemente der Folge aufeinanderfolgenden Werte haben (z. B. 3, 4), dann werden immer die Werte 1, 2 zurückgegeben.
D. Eine der vom Algorithmus zurückgegebenen Zahlen ist die Länge der längsten Folge mit gleichen Werte, in der zweiten Hälfte der Zeichenkette für jede gerade Zahl ***n*** > 1.