



October 2025

Fundamental IT Engineer Examination (Subject B)

Questions must be answered in accordance with the following:

Question Nos.	Q1 – Q20
Question Selection	All questions are compulsory.
Examination Time	12:30 - 14:10 (100 minutes)

Instructions:

1. Use a pencil. If you need to change an answer, erase your previous answer completely and neatly. Wipe away any eraser debris.
2. Mark your examinee information and test answers in accordance with the instructions below. Your answer will not be graded if you do not mark properly. Do not mark or write on the answer sheet outside of the prescribed places.

(1) Examinee Number

Write your examinee number in the space provided, and mark the appropriate space below each digit.

(2) Date of Birth

Write your date of birth (in numbers) exactly as it is printed on your examination admission card, and mark the appropriate space below each digit.

(3) Answers

Mark your answers as shown in the sample question below.

[Sample Question]

Which of the following should be used for marking your answer on the answer sheet?

Answer group

- a) Ballpoint pen b) Crayon c) Fountain pen d) Pencil

Since the correct answer is “d) Pencil”, mark the answer as below:

[Sample Answer]

Sample	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="radio"/>				
--------	-----------------------	-----------------------	-----------------------	----------------------------------	-----------------------	-----------------------	-----------------------	-----------------------	-----------------------

Do not open the exam booklet until instructed to do so.

Inquiries about the exam questions will not be answered.

Pseudo programming language notations

In algorithm and programming questions that use pseudo programming language, the following notations are used unless otherwise stated:

[Pseudo programming language notations]

Notation	Description
<code>O procedure(type: arg1, ...)</code>	Declares a <i>procedure</i> and its argument(s) <i>arg1</i> ,
<code>O ret-type: function(type: arg1, ...)</code>	Declares a <i>function</i> , its argument(s) <i>arg1</i> , ... , and type of return value <i>ret-type</i> .
<code>type: var1, ...</code> <code>type []: array1, ...</code>	Declares variables <i>var1</i> , ... and arrays <i>array1</i> , ... by data <i>type</i> such as <i>integer</i> , <i>real</i> , and <i>string</i> .
<code>/* comment */</code>	Describes a comment between /* and */.
<code>// comment</code>	Describes a comment after // till end of line.
<code>variable ← expression</code>	Assigns the value of the <i>expression</i> to the <i>variable</i> .
<code>procedure(arg1, ...)</code>	Calls a <i>procedure</i> by passing arguments <i>arg1</i> ,
<code>function(arg1, ...)</code>	Calls a <i>function</i> by passing arguments <i>arg1</i> , ... , and receiving the return value.
<code>output arg1, ...</code>	Outputs values of <i>arg1</i> , ... to a printing device.
<code>return ret-val</code>	Finishes a function by passing back a return value <i>ret-val</i> .
<code>if (condition-i) process-i elseif (condition-ei) process-ei else process-e endif</code>	Indicates the selection process. *1 If <i>condition-i</i> is true, then execute <i>process-i</i> . Otherwise, proceed to the next elseif or else. *2 If <i>condition-ei</i> is true, then execute <i>process-ei</i> . Otherwise, proceed to the next elseif or else. *3 If all conditions are false, execute <i>process-e</i> . Note: *2 and *3 can be omitted. *2 may exist twice or more.
<code>for (sequence) process endfor</code>	Indicates the “for” iteration process. In the order specified in the <i>sequence</i> , execute the <i>process</i> repeatedly.
<code>while (condition) process endwhile</code>	Indicates the “while” iteration process. While the <i>condition</i> is true, execute the <i>process</i> repeatedly.
<code>do process while (condition)</code>	Indicates the “do - while” iteration process. Execute the <i>process</i> once, and then while the <i>condition</i> is true, execute the <i>process</i> repeatedly.

Pseudo programming language notations (continued)

[Operators and their precedence]

Type of operator	Operators	Precedence	Note
Expression	(), . ⁽¹⁾	High	⁽¹⁾ accessing member or method
Unary operator	+, -, not ⁽²⁾		⁽²⁾ logical negation
Binary operator	×, ÷, mod ⁽³⁾		⁽³⁾ remainder
	+, -		
	>, <, ≥, ≤, =, ≠		
	and ⁽⁴⁾		⁽⁴⁾ logical product
	or ⁽⁵⁾	Low	⁽⁵⁾ logical sum

[Boolean-type constants]

true, false

[Array reference]

	1-dimensional array	2-dimensional array	Array of arrays																																						
Array declaration	<code>type []: name ...</code>	<code>type [,]: name ...</code>	<code>type [][]: name ...</code>																																						
Example	<code>integer []: a1</code> <table style="margin-left: auto; margin-right: auto;"> <tr><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td></tr> <tr><td>1</td><td>3</td><td>5</td><td>7</td><td>9</td></tr> </table>	1	2	3	4	5	1	3	5	7	9	<code>integer [,]: a2</code> <table style="margin-left: auto; margin-right: auto;"> <tr><td>1</td><td>2</td><td>3</td></tr> <tr><td>1</td><td>11</td><td>12</td><td>13</td></tr> <tr><td>2</td><td>14</td><td>15</td><td>16</td></tr> <tr><td>3</td><td>17</td><td>18</td><td>19</td></tr> </table>	1	2	3	1	11	12	13	2	14	15	16	3	17	18	19	<code>integer [][]: aa</code> <table style="margin-left: auto; margin-right: auto;"> <tr><td>1</td><td>2</td><td>3</td></tr> <tr><td>1</td><td>21</td><td>22</td></tr> <tr><td>2</td><td>23</td><td>24</td><td>25</td></tr> <tr><td>3</td><td>26</td><td></td></tr> </table>	1	2	3	1	21	22	2	23	24	25	3	26	
1	2	3	4	5																																					
1	3	5	7	9																																					
1	2	3																																							
1	11	12	13																																						
2	14	15	16																																						
3	17	18	19																																						
1	2	3																																							
1	21	22																																							
2	23	24	25																																						
3	26																																								
Data reference	Data 7 is referred to by <code>a1[4]</code>	Data 16 is referred to by <code>a2[2,3]</code>	Data 25 is referred to by <code>aa[2][3]</code>																																						
Notation of array contents	{1, 3, 5, 7, 9}	{ {11, 12, 13}, {14, 15, 16}, {17, 18, 19} }	{ {21, 22}, {23, 24, 25}, {26} }																																						

Note: The indexes of example arrays start at 1.

[undefined state]

`undefined` is a state in which no value is set to a variable (or an element of an array). By setting `undefined` to a variable, the variable is transformed into `undefined` state.

- Q1.** From the answer group below, select the correct combination of answers to be inserted into **A** and **B** in the program. Here, the array index starts at 1.

The function `sum` receives an integer array `array` with at least two elements and two positive integers `k` and `m` ($k < m \leq$ number of elements in `array`). It calculates the sum of even elements of the array `array` whose indices are from `k` to `m`.

[Program]

```
O integer: sum(integer []: array, integer: k, integer: m)
    integer: s ← 0
    integer: i ← k
    while (i ≤ m)
        if (B)
            s ← s + array[i]
        endif
        B
    endwhile
    return s
```

Answer group

	A	B
a)	array[i] mod 2 = 0	i ← i + 1
b)	array[i] mod 2 = 0	i ← i - 1
c)	array[i] mod 2 ≠ 0	i ← i + 1
d)	array[i] mod 2 ≠ 0	i ← i - 1

- Q2.** From the answer group below, select the correct combination of answers to be inserted into **A** and **B** in the program. Here, the array index starts at 1.

In statistics, the mode is the value that occurs most frequently in a dataset. For example, the mode of the integer array {2, 1, 1, 9, 6, 6, 2, 5, 6} is 6, as it occurs most often. The function `findMode` receives an integer array `arr` as a dataset and returns the mode for it.

[Program]

```

○ integer: findMode(integer []: arr)
    integer: n ← the number of elements in arr
    integer: m ← arr[1] /* Current mode value */
    integer: m_c ← 1      /* Frequency count of mode */
    integer: c, i, j
    for (increase i from 1 to n - 1 by 1)
        c ← 1
        for (increase j from i + 1 to n by 1)
            if (A)
                c ← c + 1
            endif
        endfor
        if (B)
            m_c ← c
            m ← arr[i]
        endif
    endfor
    return m

```

Answer group

	A	B
a)	<code>arr[i] = arr[j]</code>	<code>m < arr[i]</code>
b)	<code>arr[i] = arr[j]</code>	<code>m ≠ arr[i]</code>
c)	<code>arr[i] = arr[j]</code>	<code>m_c < c</code>
d)	<code>arr[i] = arr[j]</code>	<code>m_c > c</code>
e)	<code>m = arr[j]</code>	<code>m < arr[i]</code>
f)	<code>m = arr[j]</code>	<code>m ≠ arr[i]</code>
g)	<code>m = arr[j]</code>	<code>m_c < c</code>
h)	<code>m = arr[j]</code>	<code>m_c > c</code>

- Q3.** From the answer group below, select the correct combination of answers to be inserted into **A** and **B** in the program. Here, the array index starts at 1.

Given an array of positive integers and a target sum, the problem is to find the subarray whose sum is equal to the target sum. A subarray is a part of the given array composed of contiguous elements. For example, when the target sum is 14, the shaded subarray in the figure satisfies the given condition.

6	3	2	5	1	1	7	3
---	---	---	---	---	---	---	---

Figure Example of an array

The procedure `subArraySum` receives an integer array `arr` and an integer value `targetSum` as arguments and finds the subarray whose sum is equal to the target sum. If a subarray that satisfies this condition is found, it prints their starting and ending indices, and finishes the task. Otherwise, it prints the message “No subarray found”.

[Program]

```
O subArraySum(integer []: arr, integer: targetSum)
    integer: sum, start, end
    integer: N ← the number of elements in arr
    for (increase start from 1 to N by 1)
        sum ← 0
        for (increase end from start to A by 1)      // α
            sum ← B
            if (sum = targetSum)
                output start, end
                return
            elseif (sum > targetSum)
                exit the for block marked α
            endif
        endfor
    endfor
    output "No subarray found"
```

Answer group

	A	B
a)	N	arr[end]
b)	N	sum + arr[end]
c)	N	sum + arr[start]
d)	N - 1	arr[end]
e)	N - 1	sum + arr[end]
f)	N - 1	sum + arr[start]
g)	targetSum	arr[start]
h)	targetSum	sum + arr[end]
i)	targetSum	sum + arr[start]

- Q4.** From the answer group below, select the correct combination of answers to be inserted into **A** through **C** in the program.

The procedure `reverse_digits` receives one positive integer argument named `num` and outputs both the original integer and the integer with its digits reversed. Assume that the ones digit of `num` is not 0.

Example:

```
Number is 456, output 654
Number is 23407, output 70432
```

[Program]

```
O reverse_digits(integer: num)
    integer: rev ← 0
    integer: rm ← 0
    integer: temp ← num
    while (A)
        rm ← temp mod 10
        rev ← B
        temp ← C
    endwhile
    output "Number is ", num, ", output ", rev
```

Answer group

	A	B	C
a)	$\text{num} > 0$	$\text{rm} \times 10 + \text{rev}$	integer part of $(\text{temp} \div 10)$
b)	$\text{num} > 0$	$\text{rm} \times 10 + \text{rev}$	$\text{temp} - \text{rm}$
c)	$\text{num} > 0$	$\text{rev} \times 10 + \text{rm}$	integer part of $(\text{temp} \div 10)$
d)	$\text{num} > 0$	$\text{rev} \times 10 + \text{rm}$	$\text{temp} - \text{rm}$
e)	$\text{temp} > 0$	$\text{rm} \times 10 + \text{rev}$	integer part of $(\text{temp} \div 10)$
f)	$\text{temp} > 0$	$\text{rm} \times 10 + \text{rev}$	$\text{temp} - \text{rm}$
g)	$\text{temp} > 0$	$\text{rev} \times 10 + \text{rm}$	integer part of $(\text{temp} \div 10)$
h)	$\text{temp} > 0$	$\text{rev} \times 10 + \text{rm}$	$\text{temp} - \text{rm}$

- Q5.** From the answer group below, select the correct combination of answers to be inserted into A and B in the program.

Prime numbers are the positive integers that are only divisible by the number itself and 1. Procedure `printPrime` receives the integer N ($N \geq 1$) as an argument and prints the first N prime numbers. For example, `printPrime(6)` will print “2 3 5 7 11 13”.

[Program]

```
O printPrime(integer: N)
    integer: count, number, i
    boolean: isPrime
    count ← 1
    number ← 2
    while (count ≤ N)
        isPrime ← true
        for (increase i from 2 to integer part of the square root of number by 1)
            if (number mod i = 0)
                A
                exit the for block
            endif
        endfor
        if (isPrime = true)
            output " ", number
            B
        endif
        number ← number + 1
    endwhile
```

Answer group

	A	B
a)	<code>count ← count + 1</code>	<code>isPrime ← false</code>
b)	<code>count ← count + i</code>	<code>isPrime ← false</code>
c)	<code>isPrime ← false</code>	<code>count ← count + 1</code>
d)	<code>isPrime ← false</code>	<code>count ← count + i</code>

- Q6.** From the answer group below, select the correct answer to be inserted into [] in the description.

Class Rectangle represents a two-dimensional figure, and class Cuboid represents a three-dimensional figure. The class Cuboid is a subclass of the class Rectangle. Figures 1 and 2 explain the classes Rectangle and Cuboid, respectively.

Member variable	Type	Description
length	integer	Dimension of length.
width	integer	Dimension of width.

Constructor	Description
Rectangle(integer: _length, integer: _width)	Initialize member variable length with _length and member variable width with _width.

Method	Return value	Description
enlarge(Rectangle: shape)	None	To increase the size, add shape.length to length and shape.width to width.
output()	None	Output in format "(length, width)".

Figure 1 Class Rectangle

Member variable	Type	Description
height	integer	Dimension of height.

Constructor	Description
Cuboid(integer: _length, integer: _width, integer: _height)	Constructor of the super class is called with _length and _width as arguments in that order, and member variable height is initialized with the argument _height.

Method	Return value	Description
enlarge(Cuboid: shape)	None	To increase the size, add shape.length to length, shape.width to width and shape.height to height.
output()	None	Output in format "(length, width, height)".

Figure 2 Class Cuboid

When the program is executed, it is expected to produce the output “ ”.

[Program]

```
Rectangle: f1 ← Rectangle(5, 2)
Cuboid: f2 ← Cuboid(9, 4, 7)
f1.enlarge(f2)
f1.output()
```

Answer group

- | | | |
|--------------|--------------|---------------|
| a) (5, 2) | b) (5, 2, 7) | c) (9, 4) |
| d) (9, 4, 7) | e) (14, 6) | f) (14, 6, 7) |

- Q7.** From the answer group below, select the correct combination of answers to be inserted into **A** and **B** in the program.

A singly linked list can be reversed using the following recursive procedure.

Let **head** be the first element of the list, and let **elm** be undefined.

(a) Reverse the order of the elements in the list starting from **head**, and set the next element of the last element in the reversed list—that is, the next element of **head**—to **elm**.

To “reverse the order of the elements in the list starting from **head**,” reverse the list starting from the next element of **head**, and then connect **head** to the end of the reversed list. In other words, execute (a) by setting **head** as **elm**, and the next element of **head** as **head**.

The recursive function **reverseList** reverses a singly linked list and returns the head of the reversed list. Initially, the function **reverseList** is called as **reverseList(head, undefined)**. Here, **head** is the head of the singly linked list before reversal and is of type **ListElement**. The table provides an explanation of the member variables of the class **ListElement**. **ListElement**-type variables store references to instances of the class **ListElement**.

Table Explanation of the member variables of the class **ListElement**

Member variable	Type	Description
val	integer	The value of the element.
next	ListElement	Reference to the next element, if there is no next element, the status is undefined.

[Program]

```
O ListElement: reverseList(ListElement: head, ListElement: elm)
    ListElement: listHead
    if (head.next is not undefined)
        listHead ← reverseList(A, head)
    else
        listHead ← head
    endif
    head.next ← B
    return listHead
```

Answer group

	A	B
a)	head.next	elm
b)	head.next	elm.next
c)	elm.next	head.next.next
d)	elm.next	elm
e)	elm.next	elm.next

- Q8.** From the answer group below, select the correct combination of answers to be inserted into **A** and **B** in the program. Here, the array index starts at 1.

Stack stores data using first-in, last-out ordering. Here, the items stored in the stack are integer values, and the stack is controlled by the procedure push and the function pop. The procedure push adds an item given as an argument to the top of the stack, and the function pop removes the top item from the stack and returns it. The global variable **stck** is an array of 10 integers that stores the stack items, and the global variable **tos** is the pointer that points to the topmost item of the stack.

[Program]

```
global: integer []: stck ← {10 undefined}  
global: integer: tos ← 0
```

```
○ push(integer: item)  
    if (tos = 10)  
        output "Stack is Full"  
    else  
        A  
        stck[tos] ← item  
    endif
```

```
○ integer: pop()  
    integer: item  
    if (tos < 1)  
        output "Stack is Empty"  
        return undefined  
    else  
        B  
    endif  
    return item
```

Answer group

	A	B
a)	$\text{tos} \leftarrow \text{tos} + 1$	$\text{item} \leftarrow \text{stck}[\text{tos}]$ $\text{tos} \leftarrow \text{tos} + 1$
b)	$\text{tos} \leftarrow \text{tos} + 1$	$\text{item} \leftarrow \text{stck}[\text{tos}]$ $\text{tos} \leftarrow \text{tos} - 1$
c)	$\text{tos} \leftarrow \text{tos} + 1$	$\text{tos} \leftarrow \text{tos} + 1$ $\text{item} \leftarrow \text{stck}[\text{tos}]$
d)	$\text{tos} \leftarrow \text{tos} + 1$	$\text{tos} \leftarrow \text{tos} - 1$ $\text{item} \leftarrow \text{stck}[\text{tos}]$
e)	$\text{tos} \leftarrow \text{tos} - 1$	$\text{item} \leftarrow \text{stck}[\text{tos}]$ $\text{tos} \leftarrow \text{tos} + 1$
f)	$\text{tos} \leftarrow \text{tos} - 1$	$\text{item} \leftarrow \text{stck}[\text{tos}]$ $\text{tos} \leftarrow \text{tos} - 1$
g)	$\text{tos} \leftarrow \text{tos} - 1$	$\text{tos} \leftarrow \text{tos} + 1$ $\text{item} \leftarrow \text{stck}[\text{tos}]$
h)	$\text{tos} \leftarrow \text{tos} - 1$	$\text{tos} \leftarrow \text{tos} - 1$ $\text{item} \leftarrow \text{stck}[\text{tos}]$

- Q9.** From the answer group below, select the correct answer to be inserted into [] in the description. Here, the array index starts at 1.

The procedure `traverse` traces through a vertex of the graph in Figure 1, and outputs all vertex numbers in the graph. One of the vertex numbers of the graph is specified with the argument `k`. The global 2-dimensional array `graph` represents the graph in Figure 1. Each element `graph[i,j]` is equal to 1 if there is an edge between vertices `i` and `j`, and 0 otherwise. The array `visited` stores boolean values, where `visited[i]` indicates whether vertex `i` of the graph has been visited or not during the procedure. When the procedure is called as `traverse(1)`, the output is in the order [].

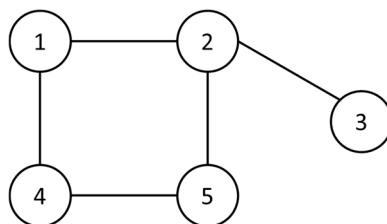


Figure 1 Graph that is handled by the program

Here, the procedure `traverse` uses a queue represented by the class `Queue`. Figure 2 provides an explanation of the class `Queue`.

Constructor	Description
<code>Queue()</code>	Creates an empty queue.

Method	Return value	Description
<code>enqueue(integer: elm)</code>	None	Adds the integer <code>elm</code> as an element to the queue.
<code>dequeue()</code>	<code>integer</code>	Extracts the element from the queue and returns it.
<code>isEmpty()</code>	<code>boolean</code>	Returns <code>true</code> if the queue is empty; otherwise, returns <code>false</code> .

Figure 2 Explanation of the class `Queue`

[Program]

```

global: integer [,]: graph ← {{0, 1, 0, 1, 0},
                                {1, 0, 1, 0, 1},
                                {0, 1, 0, 0, 0},
                                {1, 0, 0, 0, 1},
                                {0, 1, 0, 1, 0}}
  
```

```

O traverse(integer: k)
    Queue: queue ← Queue()
    boolean []: visited ← {false, false, false, false, false}
    integer: v, i
    queue.enqueue(k)
    visited[k] ← true
    while (not queue.isEmpty())
        v ← queue.dequeue()
        output v
        for (increase i from 1 to 5 by 1)
            if (graph[v,i] = 1 and visited[i] = false)
                queue.enqueue(i)
                visited[i] ← true
            endif
        endfor
    endwhile

```

Answer group

- | | |
|----------------------------|----------------------------|
| a) “1”, “2”, “3”, “4”, “5” | b) “1”, “2”, “3”, “5”, “4” |
| c) “1”, “2”, “4”, “3”, “5” | d) “1”, “2”, “4”, “5”, “3” |
| e) “1”, “2”, “5”, “4”, “3” | f) “1”, “4”, “5”, “2”, “3” |

Q10. From the answer group below, select the correct combination of answers to be inserted into **A** and **B** in the program.

The procedure `insertAfter` inserts an element into a singly linked list at the position after an existing element with a specific data value. The argument `targetData` is a string type data that represents the value of the existing element in the list, after which a new element is to be inserted. If no corresponding element exists, no action is taken. The argument `newData` is also a string type data represents the value of the new element to be inserted.

The class `Element` represents each element of the linked list. The figure explains the constructor and the member variables of the class `Element`. `Element`-type variables store references to instances of the class `Element`. A reference to the first element in the list is pre-stored in the global variable `head`.

Constructor	Description
<code>Element(string: data)</code>	Initialize the element with the data passed as an argument, which stored in the member variable <code>data</code> , with the member variable <code>next</code> set to undefined.

Member variable	Type	Description
<code>data</code>	<code>string</code>	Data associated with the element.
<code>next</code>	<code>Element</code>	This attribute represents the reference to the next element.

Figure Class Element

[Program]

```

global: Element: head      // stores first element in the list
O insertAfter(string: targetData, string: newData)
    Element: x, y
    x ← head
    while(x is A)
        if (x.data = targetData)
            y ← Element(newData)
            y.next ← B
            x.next ← y
            exit the while block
        else
            x ← x.next
        endif
    endwhile

```

Answer group

	A	B
a)	not undefined	head
b)	not undefined	x
c)	not undefined	x.data
d)	not undefined	x.next
e)	not undefined	x.next.next
f)	undefined	head
g)	undefined	x
h)	undefined	x.data
i)	undefined	x.next
j)	undefined	x.next.next

Q11. From the answer group below, select the correct combination of answers to be inserted into **A** and **B** in the program. Here, the array index starts at 1.

The bubble sort compares adjacent elements in an array and swaps them if they are out of order. It makes multiple passes through an array. The bubble sort can be modified to stop early if it finds that the array has become sorted. The function quickBubble is modified from conventional bubble sort, which sorts the elements in ascending order, to recognize a sorted array and stop early. For example, the sorting of the unordered array {18, 1, 8, 6, 2, 9, 12, 14, 7, 11} is completed in six passes.

[Program]

```
O integer []: quickBubble(integer []: array)
    integer []: arraySorted ← array
    boolean: exchange
    integer: maxIdx, i
    exchange ← true
    maxIdx ← (the number of elements in arraySorted) - 1
    while (A and B)
        exchange ← false
        for (increase i from 1 to maxIdx by 1)
            if (arraySorted[i] > arraySorted[i + 1])
                exchange ← true
                swap the values of arraySorted[i] and arraySorted[i + 1]
            endif
        endfor
        maxIdx ← maxIdx - 1
    endwhile
    return arraySorted
```

Answer group

	A	B
a)	maxIdx > 0	exchange = true
b)	maxIdx > 0	exchange = false
c)	maxIdx > 1	exchange = true
d)	maxIdx > 1	exchange = false

Q12. From the answer group below, select the correct combination of answers to be inserted into A and B in the program. Here, the array index starts at 1.

The function `containsSubstring` receives two-character arrays `str` and `seq` as arguments, and returns a boolean value indicating whether the character array `str` contains the sequence `seq` as a substring. The character array `str` contains `seq` as a substring if every sequence of consecutive characters in `seq` exists somewhere in the sequence of characters in `str`, in the same order. For example, `containsSubstring({"a", "p", "p", "l", "e"}, {"p", "l", "e"})` returns `true`, and `containsSubstring({"a", "p", "p", "l", "e"}, {"a", "e"})` returns `false`. Assume that the number of elements in `str` and `seq` is one or more.

[Program]

```
O boolean: containsSubstring(character []: str, character []: seq)
    integer: strlen, seqlen, len, i, j
    strlen ← the number of elements in str
    seqlen ← the number of elements in seq
    if (seqlen > strlen)
        return false
    endif
    len ← strlen - seqlen + 1
    for (increase i from 1 to len by 1)
        j ← 1
        while (j ≤ seqlen)
            if (A ≠ seq[j])
                exit the while block
            endif
            j ← j + 1
        endwhile
        if (B)
            return true
        endif
    endfor
    return false
```

Answer group

	A	B
a)	<code>str[i + j]</code>	<code>j = seqlen</code>
b)	<code>str[i + j]</code>	<code>j = seqlen + 1</code>
c)	<code>str[i + j]</code>	<code>j = seqlen - 1</code>
d)	<code>str[i + j - 1]</code>	<code>j = seqlen</code>
e)	<code>str[i + j - 1]</code>	<code>j = seqlen + 1</code>
f)	<code>str[i + j - 1]</code>	<code>j = seqlen - 1</code>

Q13. From the answer group below, select the correct combination of answers to be inserted into **A** through **C** in the program.

Binary Search Trees (BST) are a fundamental data structure where each node has at most two children: a left child and a right child. The key property of a BST is that each value of all nodes in its left subtree are less than the value of the node, and each value of all nodes in its right subtree are greater than the value of the node. The table shows the member variables of the class `Node`. `Node`-type variable holds a reference to an instance of the class `Node`. Each node in a BST has a different value.

Table Member variables of class `Node`

Member variable	Type	Description
<code>key</code>	<code>integer</code>	Integer to be stored in the node.
<code>left</code>	<code>Node</code>	Reference to the instance that holds the left child of a binary tree. If no left child exists, it is <code>undefined</code> .
<code>right</code>	<code>Node</code>	Reference to the instance that holds the right child of a binary tree. If no right child exists, it is <code>undefined</code> .

The function `remove` removes a node with the same key value as the argument `key` from the BST specified by the argument `node` and returns the resulting BST. The function searches for a node with the same `key` value, and if the found node (call it X here) does not have a right subtree, the function returns its left subtree. If X has a right subtree, the function locates the node (call it Y here) with the smallest key value in the right subtree, replaces the `key` value of X with that of Y, and then removes Y. If there is no node with the same `key` value in the BST, the function performs no operations. The procedure `test` demonstrates this behavior by removing 1 node from a BST containing 7 nodes. The figure shows the BST before and after node removal.

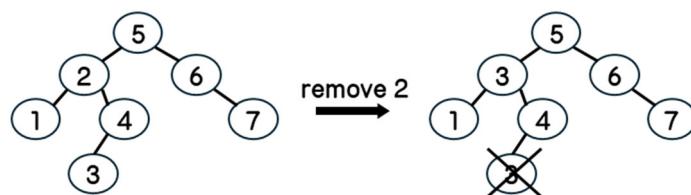


Figure BST before and after node removal

[Program]

```

O Node: remove(Node: node, integer: key)
    Node: node2
    if (node is undefined)

```

```

        return undefined
    elseif (key < node.key)
        node.left ← remove(node.left, key)
    elseif (key > node.key)
        node.right ← remove(node.right, key)
    elseif (node.right is undefined)
        return node.left
    else
        node2 ← [A]
        while ([B])
            node2 ← node2.left
        endwhile
        node.key ← node2.key
        node.right ← remove(node.right, [C])
    endif
    return node

```

O test()

Node: root ← the root node of BST on the left side of the figure
root ← remove(root, 2)

Answer group

	A	B	C
a)	node.left	node2.key < key	key
b)	node.left	node2.key < key	node2.key
c)	node.left	node2.left is not undefined	key
d)	node.left	node2.left is not undefined	node2.key
e)	node.right	node2.key < key	key
f)	node.right	node2.key < key	node2.key
g)	node.right	node2.left is not undefined	key
h)	node.right	node2.left is not undefined	node2.key

Q14. From the answer group below, select the correct answer to be inserted into in the program. Here, the array index starts at 1.

A 3×3 normal magic square is a 3×3 matrix where the sum of the elements for each row, each column, and each diagonal is the same. The square contains the numbers 1 to 9, exactly as shown in the figure.

4	9	2	→ 15
3	5	7	→ 15
8	1	6	→ 15
15	15	15	15

Figure 3×3 normal magic square

The function `checkMagicSquare` receives a two-dimensional 3×3 integer array (matrix) `m` containing the numbers 1 to 9, and returns whether the given matrix is a magic square or not.

[Program]

```

O boolean: checkMagicSquare(integer [,]: m)
    integer: i, j, k
    integer: dia1Sum ← 0
    integer: dia2Sum ← 0
    integer []: rowSum ← {0, 0, 0}
    integer []: colSum ← {0, 0, 0}
    for (increase i from 1 to 3 by 1)
        for (increase j from 1 to 3 by 1)
            rowSum[i] ← rowSum[i] + m[i,j]
            colSum[i] ← colSum[i] + m[j,i]
        endfor
        dia1Sum ← dia1Sum + m[i,i]
        dia2Sum ← dia2Sum + m[i,3 - i + 1]
    endfor
    if (dia1Sum ≠ dia2Sum)
        return false
    endif
    for (increase k from 1 to 3 by 1)
        if()
            return false
        endif
    endfor
    return true

```

Answer group

- a) $(\text{rowSum}[k] = \text{colSum}[k]) \text{ and } (\text{rowSum}[k] = \text{dia1Sum})$
- b) $(\text{rowSum}[k] = \text{colSum}[k]) \text{ or } (\text{rowSum}[k] = \text{dia1Sum})$
- c) $(\text{rowSum}[k] \neq \text{colSum}[k]) \text{ and } (\text{rowSum}[k] \neq \text{dia1Sum})$
- d) $(\text{rowSum}[k] \neq \text{colSum}[k]) \text{ or } (\text{rowSum}[k] \neq \text{dia1Sum})$

Q15. From the answer group below, select the correct combination of answers to be inserted into **A** and **B** in the program. Here, the array index starts at 1.

Inverse document frequency (IDF) is a metric used to evaluate the importance of a word in a document relative to a document collection. The function `calcIDF` receives a string `term` and an array of strings `corpus` as arguments, and calculates and returns the IDF score for the term based on the given corpus. The corpus of documents `corpus` is given as a string array. An example of `corpus` that indicates 3 documents is as follows:

```
{"ITPEC includes members from 6 countries",
"many students prepare for the exam",
"The ITPEC exam is essential for IT professionals"}
```

The table shows the description of the functions used in the program.

Table Functions

Function	Return value	Description
<code>split(string: str)</code>	<code>string []</code>	Returns words separated by a space in the text <code>str</code> .
<code>log(real: x)</code>	<code>real</code>	Returns natural logarithm (base e) of the value <code>x</code> .

Traditionally, IDF is computed using the following formula:

$$IDF(t) = \log\left(\frac{N}{1 + df(t)}\right)$$

Where N is the total number of documents, t refers to a specific term or word in a document, and $df(t)$ is the number of documents containing the term t .

For example, the return value of `calcIDF("ITPEC", corpus)` is $\log(3 \div (1 + 2)) = 0$ when array `corpus` is assigned as shown in the above example. Because the total number of documents is 3, and the total number of documents that include the word “ITPEC” is 2.

[Program]

```
O real: calcIDF(string: term, string []: corpus)
    integer: i, j, numDocs, numWords, termCount
    boolean: isContainsTerm
    real: idf
    string []: words
    termCount ← 0
    numDocs ← the number of elements in corpus
    for (increase i from 1 to numDocs by 1)
```

```

isContainsTerm ← false
words ← split(corpus[i])
numWords ← the number of elements in words
for (increase j from 1 to numWords by 1) // α
    if ( [ A ] )
        isContainsTerm ← true
        exit the for block marked α
    endif
endfor
if ( [ B ] )
    termCount ← termCount + 1
endif
endfor
idf ← log(numDocs ÷ (1 + termCount)) /* Division is performed
                                         in data type real */
return idf

```

Answer group

	A	B
a)	corpus[j] = term	isContainsTerm = false
b)	corpus[j] = term	isContainsTerm = true
c)	corpus[j] ≠ term	isContainsTerm = false
d)	corpus[j] ≠ term	isContainsTerm = true
e)	words[j] = term	isContainsTerm = false
f)	words[j] = term	isContainsTerm = true
g)	words[j] ≠ term	isContainsTerm = false
h)	words[j] ≠ term	isContainsTerm = true

Q16. From the answer group below, select the correct combination of answers to be inserted into in the program. Here, the array index starts at 0.

The Jaccard similarity index, J , measures the similarity between two sets by computing the ratio of the number of elements in their intersection to the number of elements in their union, i.e., $J(A, B) = |A \cap B| / |A \cup B|$, where A and B represent sets, \cap and \cup represent the intersection and the union set operations, respectively, and $|...|$ is the operator to calculate the number of elements in a set. The intersection operation builds a new set taking members that are common to both sets; while the union operation builds a set taking all members of both sets, where each member will appear only once. For instance, $A = \{3, 4, 6, 7\}$ and $B = \{2, 4, 5\}$, then $A \cap B = \{4\}$, $A \cup B = \{2, 3, 4, 5, 6, 7\}$, $|A| = 4$, $|B| = 3$, $|A \cap B| = 1$, and $|A \cup B| = 6$.

The function `jaccardSimilarity` calculates the Jaccard similarity stated above and returns it. The definitions of variables used in the function are stated in the below table.

Table Variable definitions

Variable	Explanation
nA	Number of elements in set A.
nB	Number of elements in set B.
iCount	Variable for counting members of intersection set, i.e., for counting $ A \cap B $.
uCount	Variable for counting members of union set, i.e., for counting $ A \cup B $.
found	Variable used to indicate whether the member has already been considered for union set or not.

[Program]

```

O real: jaccardSimilarity(integer []: A, integer []: B)
    integer: nA ← number of elements in A
    integer: nB ← number of elements in B
    integer: iCount ← 0
    integer: uCount ← 
    boolean: found
    for (increase i from 0 to nA - 1 by 1)
        found ← false
        for (increase j from 0 to nB - 1 by 1)      // α
            if (A[i] = B[j])
                iCount ← iCount + 1
                found ← true
            exit the for block marked α

```

```
        endif
    endfor
    if (found = false)
        uCount ← uCount + 1
    endif
endfor
return iCount ÷ uCount /* Division is done in data type real */
```

Answer group

- a) 0
- b) nA
- c) nB
- d) nA + nB
- e) nA - nB
- f) nA ÷ nB
- g) nA × nB

Q17. From the answer group below, select the most appropriate combination of answers to be inserted into **A** and **B** in the description.

Company P is a small business that recently formed a data analytics team to perform business data analytics using internal business data. This data is stored on a database system configured with a synchronous (real-time) mirror. The database is backed up regularly. The database servers and IT infrastructure are maintained by a third-party IT support provider.

A junior data analyst was mistakenly granted database superuser privileges to the production database. This decision was made due to a lack of security understanding in the company. The data analyst, unfamiliar with the sensitivity of their access level, was tasked with executing a series of routine database queries and migrations. However, due to an error in a script, the data analyst accidentally ran a command that dropped critical tables, resulting in a significant loss of data. This caused the company's internal applications to stop working.

The company engaged their IT support provider, and the database system was quickly put back in service by **A**. To prevent this incident from happening again, their IT support provider recommended that the company implement **B**.

Answer group

	A	B
a)	restarting the database server	database encryption
b)	restarting the database server	multi-factor authentication
c)	restarting the database server	principle of least privilege
d)	restoring from backups	database encryption
e)	restoring from backups	multi-factor authentication
f)	restoring from backups	principle of least privilege
g)	switching to the database mirror	database encryption
h)	switching to the database mirror	multi-factor authentication
i)	switching to the database mirror	principle of least privilege

Q18. From the answer group below, select the most appropriate combination of answers to be inserted into **A** and **B** in the description.

Company Q is a mid-sized online retailer selling a wide range of daily necessities to a large number of general consumers. Customers are required to create an account and log in to the company's e-commerce website to make purchases. The e-commerce website is constructed using an open-source content management system (CMS) with various plugins. It is hosted on the company's on-premise web servers located in the Demilitarized Zone (DMZ). Customer data is stored in the database installed on the web server as well. Since the e-commerce site received orders from many customers, availability of the e-commerce website is critical, and the maximum tolerable downtime (MTD) is limited to one hour per week.

The server is protected by a packet filtering firewall that permits access only through ports 80 and 443. Mr. K is an administrator responsible for both the operations and security of the company's e-commerce website. He maintains the configuration of the firewall and other security measures as well as patch management. For patching the company's e-commerce website, compatibility testing is important to maintain website stability. It takes at least three days to complete compatibility testing for one component. Due to limited human resource, it is unrealistic to perform compatibility tests in parallel and finish them quickly. Therefore, the patch management rule is to apply patches for the web server's OS and the core CMS components within a week of their release, while updates for other components including plugins are scheduled to be applied within six months. Patches for the web server's OS and the core CMS components are released on a monthly basis, while updates for other components including plugins are released each month as well.

One day, Company Q's e-commerce website was compromised and the database on the webserver was accessed by an unauthorized third party. The investigation revealed that this was caused by an unpatched vulnerability in a plugin enabled on the CMS. This plug-in was critical to the functioning of the e-commerce website. The latest version of the plugin, which includes a fix for this vulnerability, was released by the vendor three months ago. The exploitation of this vulnerability has been reported publicly since last month. Company Q was still conducting compatibility testing, and the update to the latest version was scheduled for a later time.

To prevent similar incidents from occurring, Mr. K is required to revise the patch management rules for Q's e-commerce website. In consideration of the workload associated with compatibility testing, Mr. K proposed updating the patch management rule for the e-commerce website to **A**. Furthermore, as a protective measure to prevent the servers from exploitation during the interim period before patches are applied, Mr. K proposed **B**.

Answer group

	A	B
a)	apply patches to all components including plugins within three days of their release	disabling the affected components until the patches can be applied
b)	apply patches to all components including plugins within three days of their release	implementing web application firewall (WAF) on the webserver
c)	prioritize applying patches for vulnerabilities for which exploitation methods have been identified, within one week from the date such methods are publicly confirmed	disabling the affected components until the patches can be applied
d)	prioritize applying patches for vulnerabilities for which exploitation methods have been identified, within one week from the date such methods are publicly confirmed	implementing web application firewall (WAF) on the webserver
e)	run a vulnerability scanner to verify that no vulnerabilities with confirmed exploits are detected after applying patches	disabling the affected components until the patches can be applied
f)	run a vulnerability scanner to verify that no vulnerabilities with confirmed exploits are detected after applying patches	implementing web application firewall (WAF) on the webserver

Q19. From the answer group below, select the correct combination of answers to be inserted into **A** and **B** in the description.

Company R is a software company that sells video editing software for consumers. The company has branch offices in multiple countries, all managed by its head office. In Company R's customer support department, customer satisfaction surveys are sent to selected customers who have received support. The survey results are aggregated at each branch, and must be sent to the customer survey analysis team at the head office every day. Ms. T at the customer survey analysis team at the head office is responsible for managing and securing the survey data. The considerations for collecting survey data from each branch office are as follows:

- The employees responsible for sending the survey data at each branch office changes daily.
- At the head office, Ms. T in the customer survey analysis team is responsible for receiving and analyzing the survey data from the branch offices. After receiving the survey data from the branch offices, she saves the data to a shared folder that only the customer survey analysis team can access.
- The file size of survey data is large, so using inefficient encryption methods would take too much time and is therefore not suitable.
- Since the survey data contains confidential information, it must be ensured that no one other than the sender and the intended recipients can view the contents during transmission between the head office and branch offices.
- The key used to encrypt the files must be transmitted in a way that prevents it from being stolen by a man-in-the-middle attacker.
- There are many branch offices and it is not practical to visit each office in person to distribute keys.

Ms. T decided to adopt a new method in which the branch offices upload the survey data, encrypted using file encryption, to a shared folder. She set up this shared folder with access permissions granted to both the head office and branch office staff, and instructed the branch office staff to upload the encrypted survey data to it. Taking into account the considerations related to collecting information from the branches, Ms. T decided to use **A** as the encryption method. In this method, the key used for encrypting the survey data is encrypted using the public key of **B**.

Answer group

	A	B
a)	hybrid encryption	Ms. T
b)	hybrid encryption	the branch office staff
c)	public key encryption only	Ms. T
d)	public key encryption only	the branch office staff
e)	symmetric key encryption only	Ms. T
f)	symmetric key encryption only	the branch office staff

Q20. From the answer group below, select the most appropriate combination of answers to be inserted into **A** and **B** in the description.

Company S is a software company that develops web applications. Some of web applications that Company S develops are open-sourced. The web application code is made publicly available in a repository accessible to anyone. These open-source web applications include features that interact with external services using API keys.

One day, an external security researcher reported to Company S's security team that the code published in the company's repository contained an API key. Company S immediately made the repository private and deactivated the API key. Upon further investigation, it was found that a developer on the web application development team accidentally hard-coded the API key when testing a new feature, and committed the code to the source code repository. As a result, this allowed unauthorized users to access external services using the API key.

To prevent a recurrence of such an incident, the security team has proposed adding a coding rule that instead of hard-coding the secrets including the API, use environment variables that allow the application code to be accessed dynamically. Additionally, the team suggested **A** to detect any secrets that might have been missed by this rule before committing code to a publicly accessible repository. Furthermore, to minimize damage in case any secrets still evade detection, he proposed **B**.

Answer group

	A	B
a)	checking logs of APIs used by the development team to see if there is any unauthorized use	adding a rule that mandates regular rotation of secrets
b)	checking logs of APIs used by the development team to see if there is any unauthorized use	requiring web application developers to use multi-factor-authentication to log in to the repository
c)	integrating automated secret scanning tools	adding a rule that mandates regular rotation of secrets
d)	integrating automated secret scanning tools	requiring the web application developers to use multi-factor-authentication to log in to the repository
e)	running antivirus full scans on developers' PCs	adding a rule that mandates regular rotation of secrets
f)	running antivirus full scans on developers' PCs	requiring web application developers to use multi-factor-authentication to log in to the repository

Company names and product names appearing in the test questions are trademarks or registered trademarks of their respective companies. Note that the ® and ™ symbols are not used within the text.