

Vol.2

FE Exam

Preparation Book

Preparation Book for Fundamental Information Technology Engineer Examination

Preparation for Afternoon Exam

IPA

Information-Technology Promotion Agency, Japan

Table of Contents

PREPARATION FOR AFTERNOON EXAM

Chapter 1

Hardware	1
Question 1	2

Chapter 2

Software	4
Question 1	5

Chapter 3

Data Structure and Database	7
Question 1	8

Chapter 4

Communication Network	10
Question 1	11

Chapter 5

Information Processing Technology	14
Question 1	15
Question 2	18
Question 3	20

Chapter 6

Algorithms	23
-------------------	-----------

Question 1.....24

Question 2.....28

Chapter 7

Program Design	34
-----------------------	-----------

Question 1.....35

Question 2.....41

Chapter 8

Program Development	46
----------------------------	-----------

Question 1(C Program).....47

Question 2(C Program).....51

Question 3(C Program).....57

Question 4(Java Program).....62

Question 5(Java Program).....65

Question 6(Java Program).....70

PREPARATION FOR AFTERNOON EXAM

The Afternoon Exam questions are formulated from the following eight fields: Hardware, Software, Data Structure and Database, Communication Network, Information Processing Technology, Algorithms, Program Design, and Program Development. Here, the pairs of questions and answers in each field are provided. All questions are used in the past exams.

1 Hardware

[Scope of Questions]

Numeric representation,
Character representation,
Image/sound representation, Processor,
Memory, I/O device,
Execution of operation,
Addressing scheme,
I/O process execution,
System configuration, etc.

Question 1

Q1. Read the following description about the execution of an instruction, and then answer Subquestion.

There is a computer with a main memory capacity of 65,536 words, wherein one word consists of 16 bits. It has four general purpose registers (0 through 3) and a program register ("PR" below). The format of an instruction (2 words in length) is as follows:

OP	R	X	I	D	adr
----	---	---	---	---	-----

OP: Specifies an operation code in 8 bits. In this example, the following three operation codes are used.

20₁₆: Loads the effective address into the general purpose register specified by R.

21₁₆: Loads the contents of the word indicated by the effective address into the general purpose register specified by R.

FF₁₆: Ends execution.

R: Specifies a general purpose register number (0 through 3) in two bits.

X: Specifies an index register number (1 through 3) in two bits. The general purpose register at the specified number is used as an index register. However, if "0" is specified, no index register-based modification is made.

I: Specifies "1" in a single bit for indirect addressing; otherwise, specifies "0".

D: Three extension bits which are always "0".

adr: Specifies an address in 16 bits.

The effective address of instruction is calculated as shown in the following table.

Table Relationships Between X, I, and Effective Addresses

X	I	Effective address
0	0	adr
1 through 3	0	adr + (X)
0	1	(adr)
1 through 3	1	(adr + (X))

Note (): The parentheses are used to denote the content stored in the register or address inside the parentheses.

Subquestion

From the answer group below, select the correct answers to be inserted in the blanks in the following description.

When the contents of the general purpose registers and the main memory had the values shown in the figure below (values are in HEX notation), 0100_{16} was set in PR and the program was executed. In this example, the instruction at the address 0100_{16} loads the contents 0113_{16} at the address $011B_{16}$ (underlined) to general purpose register 0.

After the execution ends, A is set to general purpose register 0, B to general purpose register 1, C to general purpose register 2, and D to general purpose register 3.

General purpose register	0: 0003	1: 0000	2: 0000	3: 0000				
Program register	PR: 0100							
Main memory								
Address	+ 0	+ 1	+ 2	+ 3	+ 4	+ 5	+ 6	+ 7
00F8	0000	0000	0000	0000	0000	0000	0000	0000
0100	2100	011B	20C0	0003	2170	0111	21B8	011A
0108	FF00	0000	0000	0000	0000	0000	0000	0000
0110	0000	0001	0002	0003	0004	0005	0006	0007
0118	0110	0111	0112	<u>0113</u>	0114	0115	0116	0117
0120	0118	0119	011A	011B	011C	011D	011E	011F

Fig. Values in General Purpose Registers, PR, and Main Memory

Answer group:

- | | | |
|----------------|----------------|----------------|
| a) 0000_{16} | b) 0001_{16} | c) 0002_{16} |
| d) 0003_{16} | e) 0004_{16} | f) 0005_{16} |
| g) 0006_{16} | h) 0113_{16} | i) 0115_{16} |

2 Software

[Scope of Questions]

System software, Application software,
Software package, OS functions,
Programming language,
Language processor, Program execution, etc.

Question 1

Q1. Read the following description about a scheduler, and then answer Subquestion.

[Description of Scheduler]

- (1) The scheduler is implemented on a single processor using multiple programming.
- (2) A time slice is 50 milliseconds.
- (3) There is no priority among processes, and round robin scheduling is used. .
- (4) If I/O occurs while a process is running, the next process is executed. Also, when the turn comes to a process waiting for I/O, that process is skipped, and the next process is executed.

Subquestion

From the answer group below, select the correct answers to be inserted in the blanks in the following description.

There are three processes: P1, P2, and P3. The figure shows the sequence in which the processor and I/O are used, and the amount of time during which they are used, when each process is executed independently.

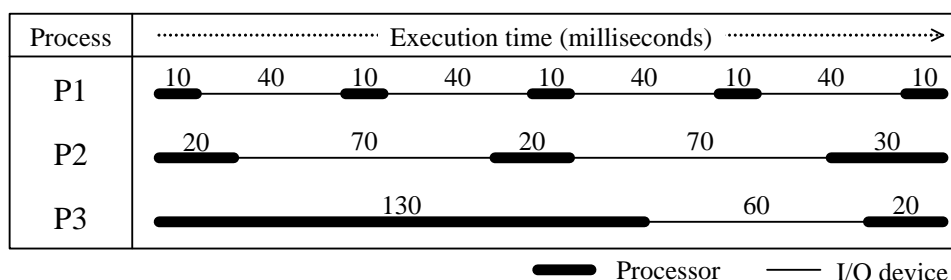


Fig. Sequence and Time Length of Use of the Processor and I/O Devices

Now, assume that the three processes P1, P2, and P3 are executed according to round robin scheduling in the sequence P1 → P2 → P3 → P1 → ... Overhead caused by switching between processes can be ignored.

- (1) When the three processes use different I/O devices α , β and γ respectively, the process that finishes first is A and the time from the start of P1 to the end of all the processes is B milliseconds.
- (2) When process P1 uses I/O device α and processes P2 and P3 both use I/O device β (one process has to wait until the other that starts to use the I/O device earlier finishes), the process(es) C take(s) a longer time than the case of (1) and the extended duration is D milliseconds .

Answer group for A and C:

- | | | |
|-------|--------------|--------------|
| a) P1 | b) P1 and P2 | c) P1 and P3 |
| d) P2 | e) P2 and P3 | f) P3 |

Answer group for B:

- | | | | |
|--------|--------|--------|--------|
| a) 250 | b) 260 | c) 270 | d) 280 |
| e) 290 | f) 300 | g) 310 | h) 320 |

Answer group for D:

- | | | | |
|-------|-------|-------|-------|
| a) 10 | b) 20 | c) 30 | d) 40 |
| e) 50 | f) 60 | g) 70 | h) 80 |

3 Data Structure and Database

[Scope of Questions]

Fundamental data structure,
Types and features of storage media,
File organization methods,
Types and characteristics of database,
Database language,
Data manipulation using SQL, etc.

Question 1

Q1. Read the following description about a relational database, and then answer Subquestions 1 and 2.

There are two tables named PRODUCT_TABLE and WHOLESALE_TABLE illustrated below. The name of each field is shown at the top. The underlined field indicates the primary key.

PRODUCT_TABLE

<u>PRODUCT_CODE</u>	PRODUCT_NAME	UNIT_PRICE	WHOLESALE_CODE
S0001	Pencil	10	T0306
S0002	Eraser	50	T1020
⋮			

WHOLESALE_TABLE

<u>WHOLESALE_CODE</u>	WHOLESALE_NAME	PHONE_NO	REMARKS
T0001	Smith Company	03-1234-5678	
T0002	Regan Company	03-8765-4321	
⋮			

Subquestion 1

From the answer group below, select the correct answers to be inserted in the blanks in the SQL statement that defines the following ORDER_TABLE.

ORDER_TABLE

<u>ORDER_NO</u>	ORDER_DATE	PRODUCT_CODE	QUANTITY
3001	2007-03-01	S0110	1000
3002	2007-03-02	S0054	2000
⋮			

CREATE TABLE ORDER_TABLE

(ORDER_NO NUMERIC(4) NOT NULL,

ORDER_DATE DATE NOT NULL,

PRODUCT_CODE CHAR(5) NOT NULL,

QUANTITY NUMERIC(5) NOT NULL,

A

KEY (ORDER_NO),

B

KEY (PRODUCT_CODE) C PRODUCT_TABLE)

Answer group:

- | | | |
|------------|---------------|-----------|
| a) FOREIGN | b) INDEX | c) MAIN |
| d) PRIMARY | e) REFERENCES | f) UNIQUE |
| g) USING | | |

Subquestion 2

From the answer group below, select the correct answers to be inserted in the blanks in the SQL statement that defines order slip view, which has the same number of lines as the order table so that it can be used to confirm the contents of the order.

ORDER_SLIP

WHOLESALE_NAME	ORDER_DATE	PRODUCT_NAME	QUANTITY	AMOUNT
Thomson Company	2007-03-01	Notebook	100	10000
Holland Company	2007-03-02	Stick glue	200	20000
:				

```
CREATE VIEW ORDER_SLIP
```

```
AS SELECT W.WHOLESALE_NAME, O.ORDER_DATE,
```

```
      P.PRODUCT_NAME, O.QUANTITY, 
```

```
FROM ORDER_TABLE O, PRODUCT_TABLE P, WHOLESALE_TABLE W 
```

Answer group for D:

- a) P.UNIT_PRICE * O.QUANTITY AS AMOUNT
- b) SUM(P.UNIT_PRICE * O.QUANTITY) AS AMOUNT
- c) AMOUNT IS P.UNIT_PRICE * O.QUANTITY
- d) AMOUNT IS SUM(P.UNIT_PRICE * O.QUANTITY)

Answer group for E:

- a) IN O.PRODUCT_CODE = P.PRODUCT_CODE
AND P.WHOLESALE_CODE = W.WHOLESALE_CODE
- b) IN O.PRODUCT_CODE = P.PRODUCT_CODE
OR P.WHOLESALE_CODE = W.WHOLESALE_CODE
- c) WHERE O.PRODUCT_CODE = P.PRODUCT_CODE
AND P.WHOLESALE_CODE = W.WHOLESALE_CODE
- d) WHERE O.PRODUCT_CODE = P.PRODUCT_CODE
OR P.WHOLESALE_CODE = W.WHOLESALE_CODE

4 Communication Network

[Scope of Questions]

Data transmission,
Transmission control, TCP/IP, LAN, WAN,
Internet, Email, Web, etc.

Question 1

Q1. Read the following description about a communication network and then answer Subquestions 1, 2 and 3.

Company A plans to provide internal information service by installing an information-providing server which the head office and the factory can commonly use.

As shown in Figure 1, the LAN in the head office and that in the factory are connected via routers to the backbone network of Company A, and the backbone network is connected via a firewall to the Internet. TCP and IP are used at the transport layer and the network layer, respectively.

At the head office and the factory, each client uses a displaying program called a browser. The browser allows the client to communicate with a myriad of servers on the Internet with an application-layer protocol called HTTP. All computers on the network are set up in such a way that communication with the outside via routers always passes through proxy servers.

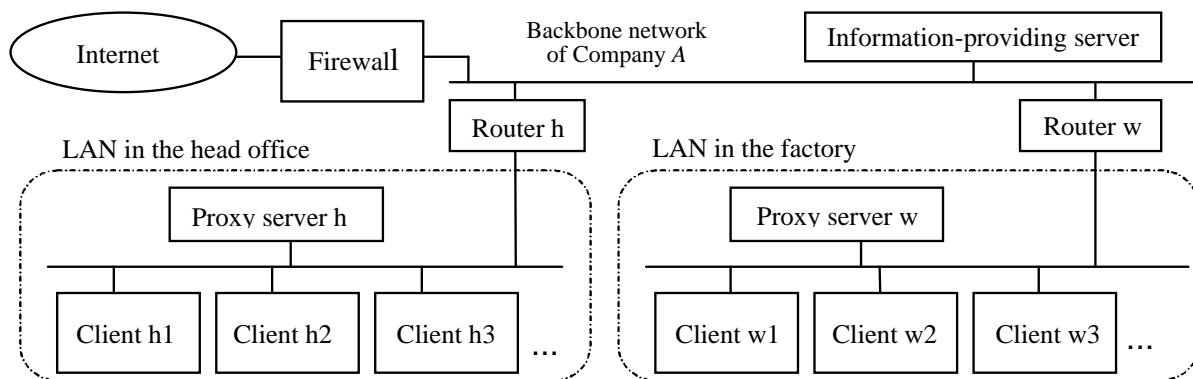


Fig. 1 Internal network of Company A

Figure 2 shows the communication path for the internal information-providing service. If HTTP, TCP and IP are used, the application layer directly receives the service from the transport layer.

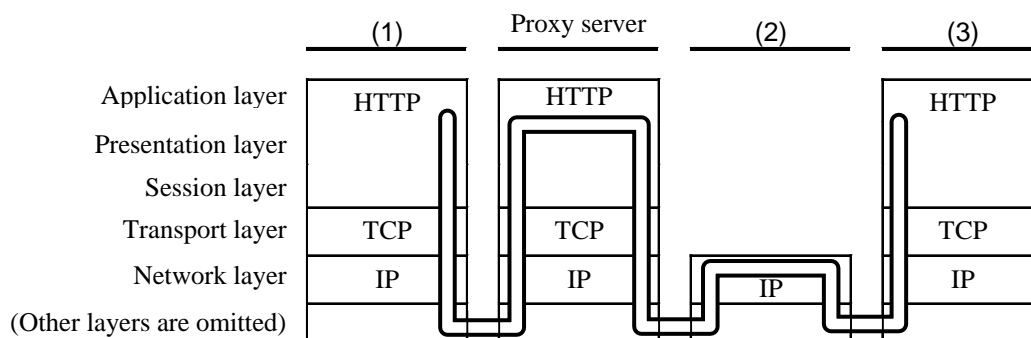


Fig. 2 Communication path for the internal information-providing service

Subquestion 1

In Figure 1, the client h1 must be authorized to pass (filter) through the router h if it communicates with the information-providing server. To be authorized, a correct source-to-destination setting must be established. From the answer group below, select the two correct source-to-destination settings.

Answer group

	Source	Destination
a)	Client h1	Information-providing server
b)	Client h1	Proxy server h
c)	Client h1	Firewall
d)	Information-providing server	Client h1
e)	Information-providing server	Proxy server h
f)	Proxy server h	Client h1
g)	Proxy server h	Information-providing server
h)	Proxy server h	Firewall
i)	Firewall	Client h1
j)	Firewall	Proxy server h

Subquestion 2

In Figure 2, a certain combination of different equipments is used in the computer network. From the answer group below, select the one correct combination that is used in Figure 2.

Answer group

	(1)	(2)	(3)
a)	Client	Information-providing server	Router
b)	Client	Router	Information-providing server
c)	Information-providing server	Client	Router
d)	Information-providing server	Router	Client
e)	Router	Client	Information-providing server
f)	Router	Information-providing server	Client

Subquestion 3

A problem occurred as described in the following. From the answer group below, select the possible cause of the problem.

All clients connecting to the head office LAN and to the factory LAN were able to communicate with the information-providing server and the servers on the Internet.

One client was added to the LAN at the factory. This new client was able to communicate directly with the other clients on the factory LAN but was unable to communicate with the information-providing server or the servers on the Internet.

All clients including the newly added client on the factory LAN used the same equipment, the same software and the same browser setting.

Answer group

- a) A wrong IP address was set on the information-providing server, thus causing incorrect access control.
- b) A wrong IP address was set on the proxy server "w", thus causing incorrect access control.
- c) A wrong IP address was set on the router "w", thus causing incorrect access control.
- d) The setting on the router "w" was wrong, causing communication using the HTTP protocol to be disabled.

5 Information Processing Technology

[Scope of Questions]

System performance, System reliability,
Risk management, Security, Standardization,
Operations research, etc.

Question 1

Q1. Read the following description about transaction processing performance, and then answer Subquestion.

As shown in the figure below, there is a server which consists of one CPU and two disks. As shown, databases DB1 and DB2 are allocated to different disks. In the case of this server, different transactions for each disk can be executed completely independently. Table 1 gives the breakdown for the average time of the use of system resources per transaction based on the result of the transaction-processing analysis.

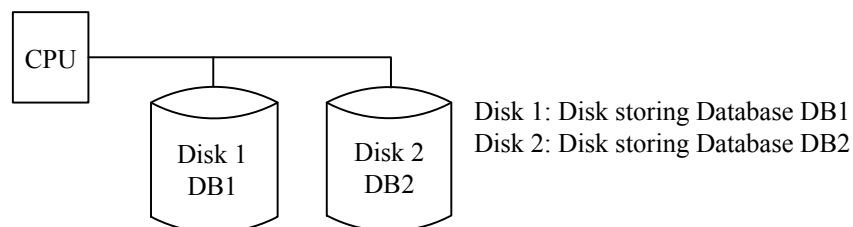


Fig. Server Design

Table 1 Average Time of Use of System Resources per Transaction

System resource	Average time of use per transaction
CPU	20 ms
Disk 1	80 ms
Disk 2	40 ms

The number of transactions processed by the server per second at any given point in time is called the TPS (Transactions Per Second) of that server. While operating, the TPS of the server, the usage rate of a given resource, and the average time of use of system resources per transaction (abbreviated hereinafter merely as "average time of use") are related as follows:

$$\text{Usage Rate} = \text{TPS} \times \text{Average Time of Use} \dots\dots [1]$$

For example, the TPS of the server is given by the following formula when the average time of use of a given system resource is 50 ms and the usage rate for that resource is 80%.

$$\text{TPS} = \frac{0.8}{0.05} = 16$$

Subquestion

From the answer groups below, select the correct answers to be inserted in the blanks in the following descriptions.

- (1) When the upper limit on the usage rate of each system resource shown in the figure is placed at 40%, the upper limit for the TPS of this system is A .
- (2) When we continue to assume that the upper limit on the usage rate of each system resource is 40%, in order to quadruple the upper limit on the TPS of the system, it is necessary to reduce the access time for the disks by at least B . Here, it is assumed that when the system resource is made n times as large as before, the system resource access time in the transaction processing is reduced to 1/n.
- (3) When the TPS of this server is 10, the average response time to process a transaction is approximately C times as long as the transaction processing time. Here, it is assumed that the average response time for each system resource is as follows, and that the average response time for transaction processing is the sum of the average response time for each system resource.

$$\text{Average Response Time} = \frac{\text{Average Time of Use}}{1 - \text{Usage Rate}} \dots\dots\dots [2]$$

- (4) Based on the above formulas [1] and [2], one can see that, as the TPS of the server goes up, the usage rate of system resources increases and D .

Answer group for A:

- | | | |
|----------|---------|---------|
| a) 0.005 | b) 0.01 | c) 0.02 |
| d) 5 | e) 10 | f) 20 |

Answer group for B:

- a) increasing the number of Disk 1 to four.
- b) increasing the number of Disk 2 to four.
- c) increasing the number of Disk 1 to two, and the number of Disk 2 to two.
- d) increasing the number of Disk 1 to two, and the number of Disk 2 to four.
- e) increasing the number of Disk 1 to four, and the number of Disk 2 to two.

Answer group for C:

- | | | |
|---------|---------|---------|
| a) 0.48 | b) 0.95 | c) 1.66 |
| d) 2.86 | e) 3.51 | |

Answer group for D:

- a) the average response time increases
- b) the average response time decreases
- c) the average time of use increases
- d) the average time of use decreases

Question 2

Q2. Read the following description about scheduling, and then answer Subquestion.

A subcontractor is selected in order to complete a certain project in the minimum number of days at the lowest possible cost. This project can be divided into eight tasks, *A* through *H*, and each of these tasks is given a specific order of precedence, as shown in Table 1. If a certain task has a precedent task, it cannot be started until the precedent tasks are completed.

Table 2 shows the number of working days and costs if these tasks are entrusted to subcontractor *X* or *Y*. Subcontractor *X* can complete each task at less cost than subcontractor *Y*, though the former requires more working days than the latter.

Table 1 List of tasks

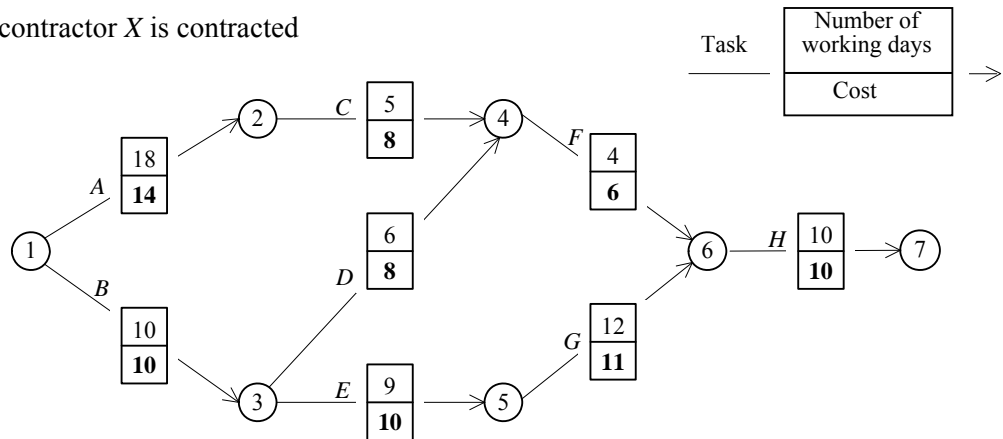
Task	Precedent task
<i>A</i>	–
<i>B</i>	–
<i>C</i>	<i>A</i>
<i>D</i>	<i>B</i>
<i>E</i>	<i>B</i>
<i>F</i>	<i>C, D</i>
<i>G</i>	<i>E</i>
<i>H</i>	<i>F, G</i>

Table 2 Number of working days and costs by subcontractor

Task	Subcontractor <i>X</i>		Subcontractor <i>Y</i>	
	Number of working days	Cost	Number of working days	Cost
<i>A</i>	18	14	16	15
<i>B</i>	10	10	8	13
<i>C</i>	5	8	4	10
<i>D</i>	6	8	5	10
<i>E</i>	9	10	8	13
<i>F</i>	4	6	3	9
<i>G</i>	12	11	10	12
<i>H</i>	10	10	9	12

Each of the arrow diagrams on the next page shows the case in which subcontractor *X* or *Y* is contracted to perform all the tasks.

If subcontractor X is contracted



If subcontractor Y is contracted

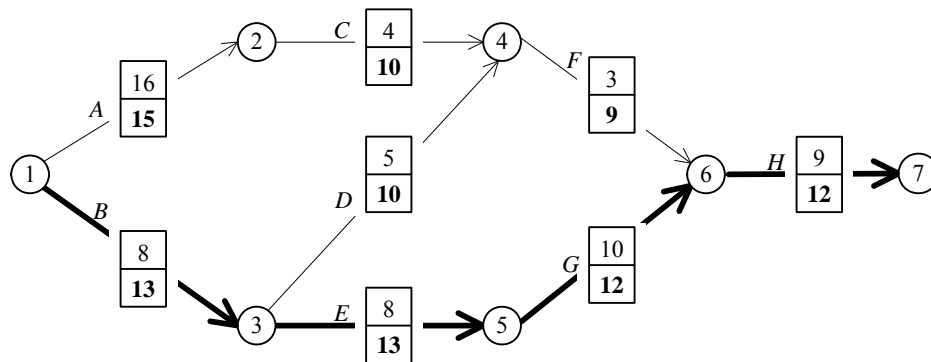


Figure Arrow diagram of the project

Subquestion

From the answer group below, select the correct answers to be inserted in the blanks in the following description.

- (1) When subcontractor X is contracted to perform all the tasks, the cost needed to complete the project is A and the number of days needed is B.
- (2) When subcontractor Y is contracted to perform all the tasks, the cost needed to complete the project is 94 and the number of days needed is 35. The critical path for this schedule is *BEGH*, shown by the thick line in the diagram for subcontractor Y. If subcontractor X is contracted to perform part of the tasks, the number of working days remains the same (35 days) and the cost can be decreased; specifically, the lowest cost will be C.

Answer group

- | | | | | |
|-------|-------|-------|-------|-------|
| a) 35 | b) 41 | c) 63 | d) 74 | e) 77 |
| f) 86 | g) 87 | h) 88 | i) 89 | j) 90 |

Question 3

Q3. Read the following description about the floating point number representation, and then answer Subquestions 1 through 3.

Represent a number in the 32-bit floating point representation as shown in Figure 1.

The meaning of each bit is as follows:

(1) Bit 0: Sign

This bit contains the sign of the mantissa. "0" indicates a positive value, and "1" indicates a negative value.

(2) Bits 1 through 7: Exponent

These bits contain the binary expression of the exponent in base two. Negative values are expressed in two's complement.

(3) Bits 8 through 31: Mantissa

These bits contain the binary expression of the absolute value of the mantissa. Bit 8 contains the first bit in the fractional part.

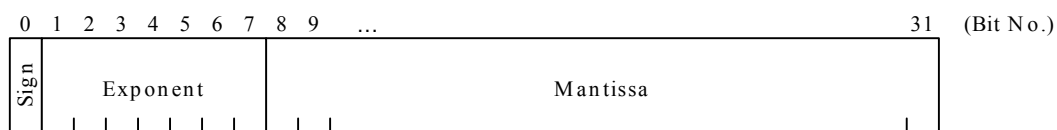


Fig. 1 Format for Floating Point Representation

Subquestion 1

Figure 2 illustrates a method for converting a binary value in floating point representation to a decimal value. Select the decimal values to be inserted in the blanks A and B from the answer groups below.

Here, " $(0.11101)_2$ " in Figure 2 indicates that "0.11101" is a binary number.

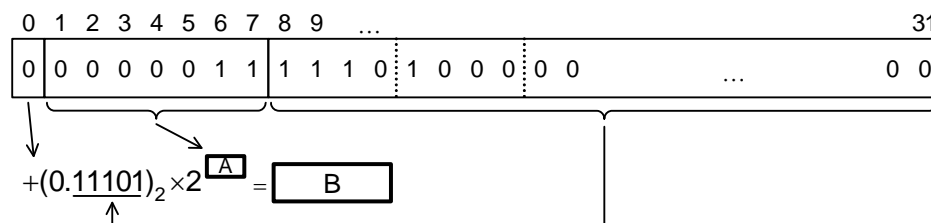


Fig. 2 Method for Converting a Binary Number in Floating Point Representation to a Decimal Value

Subquestion 3

Select the normalized representation of the floating point number in Subquestion 2 from the answer group below. Here, "normalizing" means manipulating the exponent and the mantissa to set the leftmost bit of the mantissa (Bit 8) to "1".

Answer group:

a)

0	1	1	1	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	...	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---

b)

0	1	1	1	1	1	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	...	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---

c)

0	1	1	1	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	...	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---

d)

0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	...	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---

e)

0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	...	0	0
---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	---	-----	---	---

6 Algorithms

[Scope of Questions]

Sort, Search, Character string processing,
File processing, Diagram, Graph, Numeric
calculation, etc.

Question 1

Q1. Read the following program description and the program itself, and then answer Subquestion.

[Program Description]

Function `RadixConv` is a program that converts a digit string in base M ($2 \leq M \leq 16$) to a digit string in base N ($2 \leq N \leq 16$).

- (1) A base M digit string consists of base M digits alone without space characters. In the case of base 11~16, alphabetic characters "A" ~ "F" are used to denote decimal values 10 ~ 15.
- (2) `RadixConv` first converts a base M digit string to an integer and then converts it to a base N digit string. Function `MToInt` converts a base M digit string to an integer and function `IntToN` converts the integer to a base N digit string.
- (3) Function `MToInt` and function `IntToN` use the following functions:
 - (i) Function `ToInt` that converts a numerical character P ("0", "1", ..., or "F") to an integer
 - (ii) Function `ToStr` that converts an integer Q ($0 \leq Q \leq 15$) to a numerical character ("0", "1", ..., or "F")
 - (iii) Predefined function `Length` that returns the length of the string
 - (iv) Predefined function `Substr` that extracts part of the string
- (4) Tables 1 through 5 below list the specification of arguments and return values of the functions.

Table 1 RadixConv

Argument/ Return value	Data type	Meaning
<code>FrDx</code>	Integer	Radix of digit string before conversion ($2 \leq FrDx \leq 16$)
<code>Fnum</code>	Char	Digit string before conversion
<code>TrDx</code>	Integer	Radix of converted digit string ($2 \leq TrDx \leq 16$)
Return value	Char	Converted digit string in base <code>TrDx</code>

Table 2 MToInt

Argument/ Return value	Data type	Meaning
<code>Rdx</code>	Integer	Radix of digit string before conversion ($2 \leq Rdx \leq 16$)
<code>Num</code>	Char	Digit string before conversion
Return value	Integer	Converted integer

Table 3 IntToN

Argument/ Return value	Data type	Meaning
Val	Integer	Integer before conversion
Rdx	Integer	Radix of converted digit string ($2 \leq \text{Rdx} \leq 16$)
Return value	Char	Converted digit string in base Rdx

Table 4 ToInt

Argument/ Return value	Data type	Meaning
P	Char	A numerical character before conversion ("0", "1", ..., or "F")
Return value	Integer	Converted integer

Table 5 ToStr

Argument/ Return value	Data type	Meaning
Q	Integer	Integer before conversion ($0 \leq Q \leq 15$)
Return value	Char	Converted numerical character

[Program]

```

○ char_type: RadixConv (int_type: Frdx, char_type: Fnum, int_type: Trdx)
• return IntToN(MToInt(Frdx, Fnum), Trdx)
/* Takes IntToN value as return value of function */

○ int_type: MToInt(int_type: Rdx, char_type: Num)
○ int_type: Idx, Val
• Val ← 0
■ Idx: 1, Idx ≤ Length(Num), 1 /* Length returns string length of Num */
  • Val ← A + ToInt(Substr(Num, Idx, 1))
■ /* Substr returns the Idxth (>= 1) character from the beginning of Num */
• return Val /* Takes Val as the return value of function */

○ char_type: IntToN(int_type: Val, int_type: Rdx)
○ int_type: Quo /* Quotient */
○ int_type: Rem /* Remainder */
○ char_type: Tmp
• B
• Tmp ← ""
■ Quo ≥ Rdx
  • Rem ← Quo % Rdx
  • Tmp ← ToString(Rem) + Tmp /* + is an operator to concatenate strings */
  • C
■
• D
• return Tmp /* Takes Tmp as return value of function */

○ int_type: ToInt(char_type: P)
○ int_type: Idx
○ char_type: Code[16] /* Index begins at 0 */
/* Code stores initial values "0", "1", "2", "3", "4", "5", "6", "7", /*
/* "8", "9", "A", "B", "C", "D", "E", "F" in this order */
/* Character values are incremented in this order */
• Idx ← 0
■ E /* Compare strings */
  • Idx ← Idx + 1
■
• return Idx /* Takes Idx as the return value of function */

○ char_type: ToString(int_type: Q)
○ char_type: Code[16] /* Index begins at 0 */
/* Code stores initial values "0", "1", "2", "3", "4", "5", "6", "7", /*
/* "8", "9", "A", "B", "C", "D", "E", "F" in this order */
/* Character values are incremented in this order */
• return Code[Q] /* Takes Code[Q] as the return value of the function */

```

Subquestion

From the answer groups below, select the correct answers to be inserted in the blanks in the above program.

Answer group for A:

- | | |
|--------------|--------------|
| a) Rdx | b) Val |
| c) Val * Rdx | d) Val / Rdx |

Answer group for B, C, and D:

- | | |
|--------------------------------------|--------------------------------------|
| a) Quo \leftarrow Quo / Rdx | b) Quo \leftarrow Quo / Rem |
| c) Quo \leftarrow Rdx | d) Quo \leftarrow Rem / Rdx |
| e) Quo \leftarrow Val | f) Rem \leftarrow Rdx |
| g) Rem \leftarrow Val | h) Tmp \leftarrow ToStr(Quo) + Tmp |
| i) Tmp \leftarrow ToStr(Rem) + Tmp | |

Answer group for E:

- | | |
|-------------------|-------------------|
| a) P < Code[Idx] | b) P > Code[Idx] |
| c) P <= Code[Idx] | d) P >= Code[Idx] |

Question 2

Q2. Read the following program description, pseudo-language syntax explanation, and the program itself, and then answer Subquestions 1 and 2.

[Program Description]

This is a Sub-program `QuickSort` which sorts the integers from `A[Min]` to `A[Max]` ($0 \leq \text{Min} < \text{Max}$) in one-dimensional array `A`.

(1) Sorting procedures are as follows.

- (i) The program searches the array from `A[Min+1]` to `A[Max]` sequentially for an element whose value is different from the value of `A[Min]`, take the first such element found, compare it with `A[Min]`, and select whichever is the larger as the reference value (`Pivot`). If all of the elements in the array are the same, sort processing ends. For selecting the reference value, a sub-program `FindPivot` is used.
- (ii) Elements are rearranged so that all elements less than the `pivot` are `A[Min]`, ..., `A[i-1]` ($\text{Min} < i \leq \text{Max}$) and all elements equal to or greater than the `pivot` are `A[i]`, ..., `A[Max]`. A sub-program `Arrange` performs this processing.
- (iii) The rearranged elements (`A[Min]`, ..., `A[i-1]` and `A[i]`, ..., `A[Max]`) are treated as two new arrays and sorted by recursively applying `QuickSort`.

(2) Argument specifications for the sub-programs are given in the following tables.

Table 1 QuickSort Arguments

Variable name	Input/Output	Meaning
A	Input/Output	One-dimensional array to be sorted
Min	Input	Index of first element in sort range
Max	Input	Index of last element in sort range

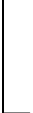
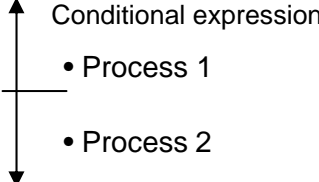
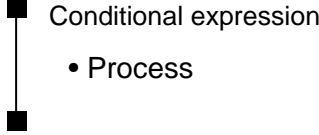
Table 2 FindPivot Arguments

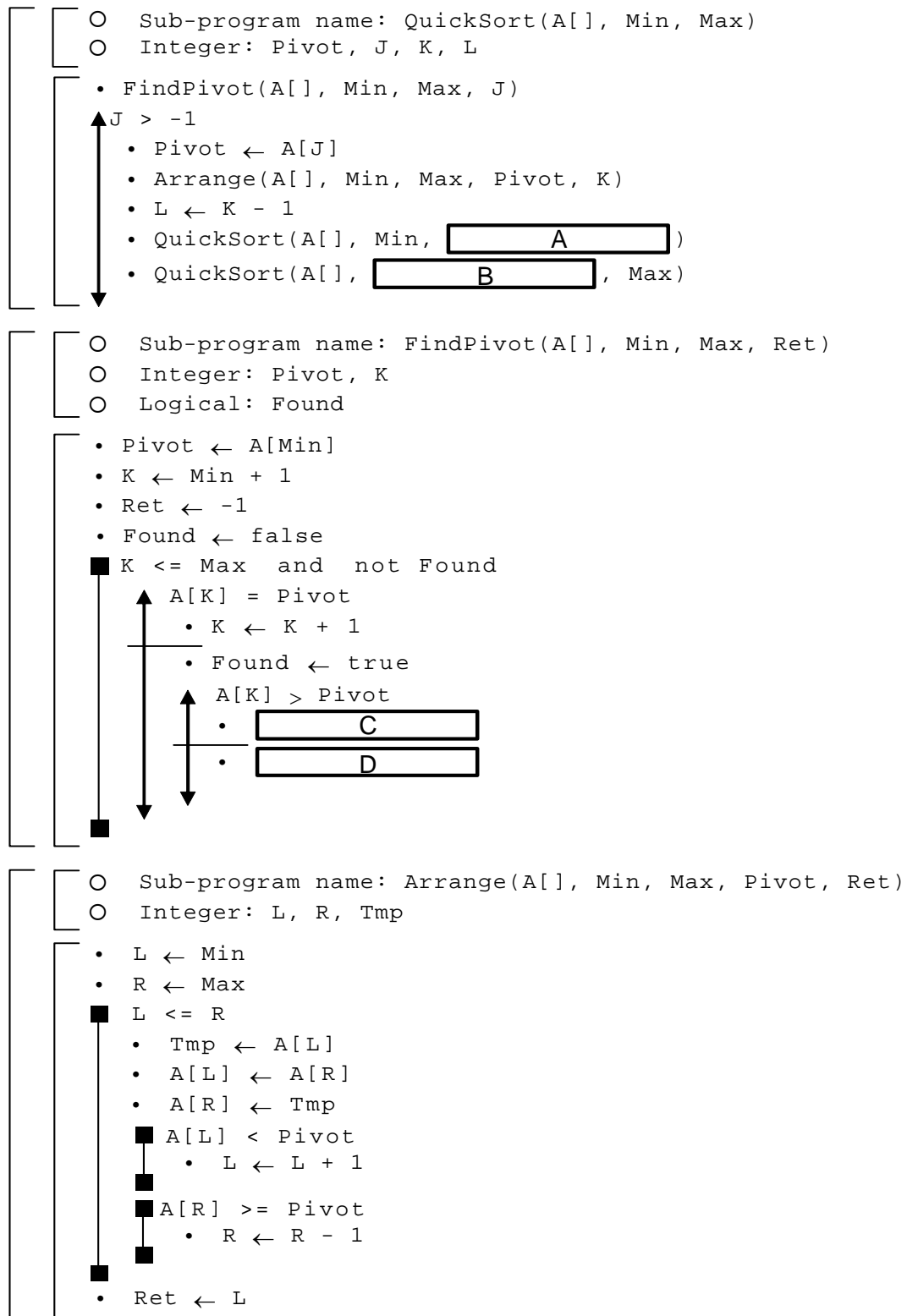
Variable name	Input/Output	Meaning
A	Input	One-dimensional array to be sorted
Min	Input	Index of first element in sort range
Max	Input	Index of last element in sort range
Ret	Output	Returns index of element storing pivot value. However, returns "-1" if A [Min], ..., A [Max] are all the same value.

Table 3 Arrange Argument

Variable name	Input/Output	Meaning
A	Input/Output	One-dimensional array to be sorted
Min	Input	Index of first element in sort range
Max	Input	Index of last element in sort range
Pivot	Input	Reference value
Ret	Output	Rearranges elements so values A[Min], ..., A[i-1] are less than Pivot and values A[i], ... A[Max] are equal to or greater than Pivot, and returns the value of i.

[Pseudo-Language Syntax Explanation]

Syntax	Explanation
	A continuous area where declarations and processes are described.
○	Declares name, type and other attributes of procedures, variables, etc.
• Variable ← Expression	Substitutes expression value for variable.
{statement}	Writes comment.
	<p>Indicates selection.</p> <p>When the conditional expression is true, process 1 is executed. When false, process 2 is executed.</p>
	<p>Indicates repetition with termination condition at the top.</p> <p>While the conditional expression is true, the process is executed.</p>

[Program]

Subquestion 1

From the answer groups below, select the correct answers to be inserted in the blanks in the above program.

Answer group for A and B:

- a) K b) L c) Max d) Min

Answer group for C and D:

- a) Ret \leftarrow A[K] b) Ret \leftarrow A[Max] c) Ret \leftarrow A[Min]
d) Ret \leftarrow K e) Ret \leftarrow Max f) Ret \leftarrow Min

Subquestion 2

The contents of arguments have been summarized in Table 4 after using QuickSort to sort elements $A[0]$ to $A[9]$ of an integer-type one-dimensional array. From the answer groups below, select the correct answers to be inserted in the blanks in the following table.

Table 4 QuickSort Call Count and Content of arguments

Call count cycle	A	Min	Max
1 st cycle	<div style="display: flex; justify-content: space-between;"> $A[0]$ $A[9]$ </div> <div style="border: 1px solid black; padding: 2px; text-align: center;"> 3 5 8 4 0 6 9 1 2 7 </div>	0	9
2 nd cycle	<div style="border: 1px solid black; padding: 2px; text-align: center;"> 3 2 1 4 0 6 9 8 5 7 </div>	0	4
3 rd cycle	<div style="border: 2px solid black; padding: 5px; display: inline-block;">E</div>	<div style="border: 2px solid black; padding: 5px; display: inline-block;">F</div>	<div style="border: 2px solid black; padding: 5px; display: inline-block;">G</div>
⋮	⋮	⋮	⋮
n th cycle	<div style="border: 1px solid black; padding: 2px; text-align: center;"> 0 1 2 3 4 5 6 7 8 9 </div>	9	9

Answer group for E:

- a)

0 | 1 | 2 | 4 | 3 | 6 | 9 | 8 | 5 | 7
- b)

0 | 2 | 1 | 3 | 4 | 6 | 9 | 8 | 5 | 7
- c)

0 | 2 | 1 | 4 | 3 | 6 | 9 | 8 | 5 | 7
- d)

0 | 4 | 1 | 2 | 3 | 6 | 9 | 8 | 5 | 7

Answer group for F and G:

- | | | |
|------|------|------|
| a) 0 | b) 1 | c) 2 |
| d) 3 | e) 4 | f) 5 |

7 Program Design

[Scope of Questions]

System development process,
Program design process, Structured design,
Module design,
Program design document, etc.

Question 1

Q1. Read the following description about program design, and then answer the Subquestions 1 through 4.

A system is designed that provides domestic news based on users' preferences. The outline of this system is as follows.

[System Outline]

- (1) News to be provided is saved in a news file in the following format. The news file is a sequential file consisting of records each of which contains a news item, and the news items are recorded in the order they were registered.

Record Format of News File

Date of registration	Time of registration	Date of occurrence	Time of occurrence	Category	Head-line	Out-line	Details	Address of image file
----------------------	----------------------	--------------------	--------------------	----------	-----------	----------	---------	-----------------------

The date of registration is an eight-digit character string representing the year, month and day when the news item was registered. For example, "20080401" represents April 1, 2008. The time of registration is a four-digit character string representing hours and minutes when the news item was registered. For example, "1830" represents 6:30 p.m. The date of occurrence is the year, month and day when the news event occurred and has the same format as that of the date of registration. The time of occurrence is the time when the news event occurred and has the same format as that of the time of registration. The category is a category of news, and stores one of the following: "Healthcare," "Education," "Economy," "Entertainment," "Science," "Society," "Sports," and "Politics." In addition, the headline for the news item, its outline, its details, and the address of a related image file are included.

- (2) User information is registered in the users file in the following format. The users file is an indexed file using the user ID as a key.

Record Format of Users File

User ID	Category 1	Category 2	Category 3	Category 4	Category 5	Date of last use	Time of last use
---------	------------	------------	------------	------------	------------	------------------	------------------

The users file registers up to 5 categories of each user's preferred news in category fields 1 through 5. If less than five categories, "NIL" (indicates empty) is stored in the remaining category field(s). The system provides at a maximum of 10 news items backwards by date and time for each registered category.

The date of last use and time of last use are the date and time when the user used the system last. The date of last use and the time of last use are character strings in the same

formats as the date of registration and the time of the registration respectively.

- (3) If more than 24 hours have elapsed since the date and time of the last use, all of the news items registered within the past 24 hours are subject to extraction. If the previous use of the system by a user is within 24 hours, only news items which were registered after the time of the last use are subject to extraction.
- (4) Based on the type of the user's terminal, the system changes the format of extraction results to be provided. If the user's terminal is a cell-phone, the applicable terminal type is "Simple," and the system provides an outline of each news item. If the user's terminal is a PC, the applicable terminal type is "Detail," and the system provides details of each news item and associated images.

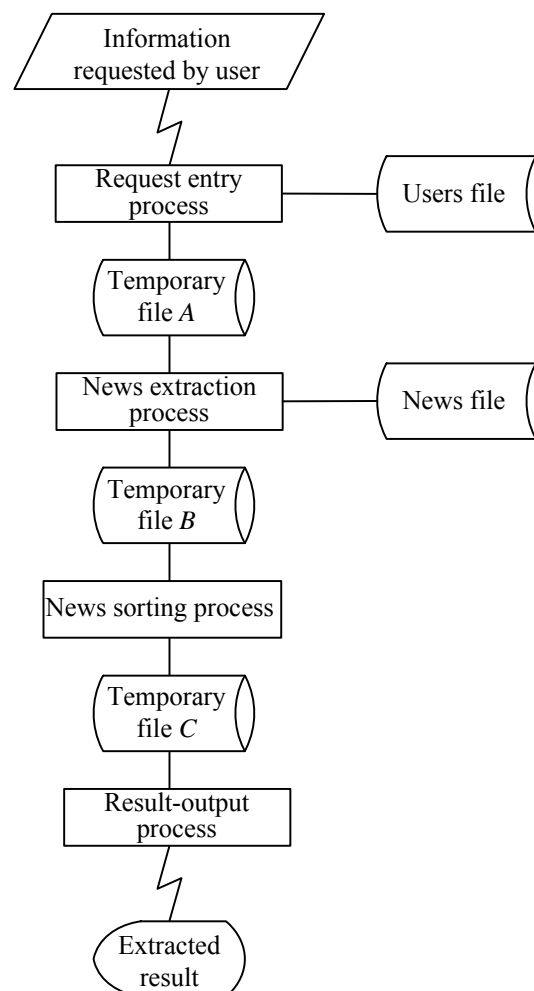


Fig. Process Flow

[Process Description]

- (1) In the request entry process, a record whose user ID is the same as the one included in the requested information is selected from the users file. The requested information from the user has the following format.

The relevant category, "Simple" or "Detail," is stored in the terminal type field.

Format of Requested Information

User ID	Terminal type
---------	---------------

Records in the following format are written into temporary file *A* from the requested information and the selected record.

Record Format of Temporary File A

Category 1	Category 2	Category 3	Category 4	Category 5	Date of extraction start	Time of extraction start	Terminal type
------------	------------	------------	------------	------------	--------------------------	--------------------------	---------------

Finally, the date of last use and the time of last use in the relevant user record in the users file are updated to the current date and time.

- (2) In the news extraction process, based on temporary file *A*, the record satisfying the following two conditions is simultaneously extracted from the news file.
- (i) The date and time of registration of the news record in the news file are more recent than the date and time of starting extraction from temporary file *A*.
 - (ii) The category of news file records meets one of category fields 1 through 5 in temporary file *A*. Necessary information is added to the record satisfying these conditions, and then the record is written to temporary file *B*.
- (3) In the news sorting process, temporary file *B* is sorted and then written to temporary file *C*.
- (4) In the result-output process, up to 10 items are extracted for each category from temporary file *C*, and the results are written in the following format in accordance with the terminal type.

Format used when the terminal type is "Simple":

Date of occurrence	Time of occurrence	Category	Headline	Outline
--------------------	--------------------	----------	----------	---------

Format used when the terminal type is "Detail":

Date of occurrence	Time of occurrence	Category	Headline	Detail	Address of image file
--------------------	--------------------	----------	----------	--------	-----------------------

- (5) Here, information between processes is transferred only using the files shown in the figure.

Subquestion 1

From the answer group below, select the correct answers to be inserted in the blanks in the following description.

A user's record in the users file is shown below:

XR205	Economy	Entertainment	NIL	NIL	NIL	20080401	2038
-------	---------	---------------	-----	-----	-----	----------	------

If this user used the system at 17:00 on April 3, 2008, A is stored in the Time-of-extraction- start field in the temporary file A by the request entry process, and B is stored in the Time-of-extraction-start field.

Answer group:

- | | | |
|---------------|---------------|---------------|
| a) "0000" | b) "1700" | c) "2038" |
| d) "20080401" | e) "20080402" | f) "20080403" |

Subquestion 2

The record format of temporary file B written by the news extraction process is shown below. From the answer group below, select the correct answers to be inserted in the blanks in this format.

Date of occurrence	Time of occurrence	<input type="text"/> C	Headline	Outline	Detail	Address of image file	<input type="text"/> D
--------------------	--------------------	------------------------	----------	---------	--------	-----------------------	------------------------

Answer group:

- | | |
|-----------------------------|-----------------------------|
| a) Terminal type | b) Time of extraction start |
| c) Date of extraction start | d) Time of registration |
| e) Date of registration | f) Category |
| g) User ID | |

Subquestion 3

From the answer group below, select the correct answers to be inserted in the blanks in the following description.

In the news-sorting process, records obtained in the news extraction process are subject to sort. In this case, the first sort key is , and the records are sorted in ascending order of the character codes. The second sort key is , and the records are sorted in descending order. The third sort key is , and the records are sorted in descending order.

Answer group:

- | | |
|-------------------------|-----------------------------|
| a) Outline | b) Address of image file |
| c) Detail | d) Time of extraction start |
| e) Time of registration | f) Date of registration |
| g) Time of occurrence | h) Date of occurrence |
| i) Category | |

Subquestion 4

From the answer group below, select the correct answers to be inserted in the blanks in the following description.

A user's record in the users file is shown as follows:

DT512	Politics	Economy	Sports	NIL	NIL	20080401	0400
-------	----------	---------	--------	-----	-----	----------	------

The number of registered news items for each category at 11:00 on April 1, 2008, is as shown below. If this user uses the system at the same time on the same day, the number of records written to temporary file C by the news sort process is , and the number of news items displayed by the result-output process is .

Table Number of Registered News Items

Category	Number of registered items per hour										
	00:00	01:00	02:00	03:00	04:00	05:00	06:00	07:00	08:00	09:00	10:00
	to 00:59	to 01:59	to 02:59	to 03:59	to 04:59	to 05:59	to 06:59	to 07:59	to 08:59	to 09:59	to 10:59
Healthcare	0	0	0	0	0	0	2	0	1	0	0
Education	0	0	0	0	0	1	0	1	0	0	0
Economy	0	0	0	1	0	2	2	6	8	2	4
Entertainment	1	0	1	0	1	2	0	1	2	1	1
Science	1	0	1	0	1	1	3	2	3	2	5
Society	0	1	2	0	1	0	2	3	5	4	3
Sports	0	0	0	1	0	0	1	1	0	3	2
Politics	0	1	0	0	0	1	4	2	3	6	3

Answer group:

- a) 27 b) 28 c) 50 d) 53
 e) 60 f) 98 g) 108

Question 2

Q2. Read the following description about program design, and then answer the Subquestions 1 through 3.

A program is developed for inventory query as a part of an inventory management system.

This inventory inquiry program handles two types of transactions.

Transaction type "A" is a query about movement(in/out) of parts during a specified period. A parts movement table is the output from this type of transaction.

Transaction type "S" is a query about current inventory. An inventory status table is the output from this type of transaction. If there are any parts in the inventory which are below the minimum stock quantity, then those parts are indicated in red.

Each transaction has the format as shown below. The number of digits, the data type, etc. in each field are checked. In addition, it must have a use authorization number in order to use the inventory-related tables. The existence of the use authorization number of the transaction in the use authorization number table is also checked. Transactions that do not pass these checks are processed as errors.

[Transaction format]

(1) Transaction type "A"

A	Transaction type	Period start (year, month, date)	Period end (year, month, date)
Number of part number	Part number	Part number	...
Part number	Part number	Part number	Part number

(2) Transaction type "S"

A	Transaction type
Number of part number	Part number
Part number	Part number
Part number	Part number

[Part of database formats related to inventory inquiries]

(1) Parts table

Part number	Part name	Ordering unit	Minimum stock quantity
-------------	-----------	---------------	------------------------

(2) Inventory table

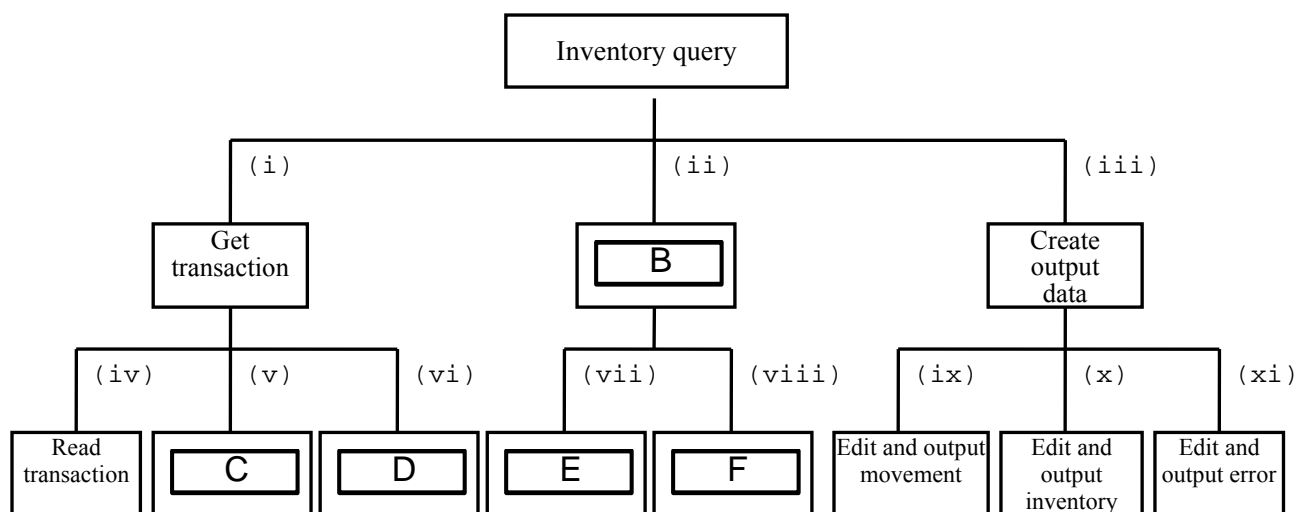
Part number	Stock quantity
-------------	----------------

(3) Use authorization number table

Use authorization number

(4) Movement table

Part number	In/Out	Quantity	Time stamp (year, month, date, hour, minute)
-------------	--------	----------	---

[Module structure chart]

[Inter-module interfaces]

Interface number	Input	Output
(i)	–	Transaction, format check flag, use authorization check flag
(ii)	Transaction	Matching records, low inventory flag, part number missing flag
(iii)	Matching records, G , low inventory flag, part number missing flag, format check flag, use authorization check flag	–
(iv)	–	Transaction
(v)	Use authorization number	Use authorization check flag
(vi)	Transaction	Format check flag
(vii)	Transaction	Matching records (parts table, movement table), part number missing flag
(viii)	Transaction	Matching records (parts table, inventory table), low inventory flag, part number missing flag
(ix)	Matching records (parts table, movement table)	–
(x)	Matching records (parts table, inventory table), low inventory flag	–
(xi)	Transaction, H	–

[Flag format]

- (1) Format check flag, use authorization check flag

Flag

(Passed: 0, Failed: 1)

- (2) Low inventory flag

Number of part number	Flag	Flag	...	Flag
-----------------------	------	------	-----	------

(In the order of part numbers in transaction; more than or equal to minimum stock quantity existent: 0, low inventory: 1)

- (3) Part number missing flag

Number of part number	Flag	Flag	...	Flag
-----------------------	------	------	-----	------

(In the order of part numbers in transaction; part number existent: 0, part number missing: 1)

Subquestion 1

From the answer group below, select the correct answers to be inserted in the blanks in the above transaction format.

Answer group for A:

- a) "A"
- b) "AS"
- c) "S"
- d) Number of transactions
- e) Movement
- f) Time stamp (year, month, date, hour, minute)
- g) Flag
- h) Use authorization number

Subquestion 2

From the answer group below, select the correct answers to be inserted in the blanks in the above module structure chart.

Answer group for B through F:

- a) Get matching records
- b) Process records related to inventory status
- c) Update inventory table
- d) Check transaction format
- e) Process records related to movement activities
- f) Update the entry and dispatch table
- g) Create an order placement message
- h) Check use authorization number

Subquestion 3

From the answer group below, select the correct answers to be inserted in the blanks in the above inter-module interfaces table.

Answer group for G:

- a) Transaction
- b) Transaction type
- c) Number of transactions
- d) Time stamp (year, month, date, hour, minute)
- e) Part number
- f) Number of part number
- g) Part name
- h) Use authorization number
- i) Use authorization number table

Answer group for H:

- a) Low inventory flag, part number missing flag
- b) Low inventory flag, part number missing flag, format check flag
- c) Low inventory flag, part number missing flag, use authorization check flag
- d) Low inventory flag, format check flag
- e) Low inventory flag, format check flag, use authorization check flag
- f) Low inventory flag, use authorization check flag
- g) Part number missing flag, format check flag
- h) Part number missing flag, format check flag, use authorization check flag
- i) Part number missing flag, use authorization check flag
- j) Format check flag, use authorization check flag

8 Program Development

[Scope of Questions]

Programming languages (C, JavaTM),
Coding, Development environment,
Test methods, etc.

Question 1**C Program**

Q1. Read the following description of a C program and the program itself, and then answer Subquestion.

[Program Description]

This is a seat reservation program for a concert hall. The program gets the number of consecutive seats to be reserved as input, then allocates the seats, and returns the result.

(1) The seating arrangement in the concert hall is as shown in the figure below:

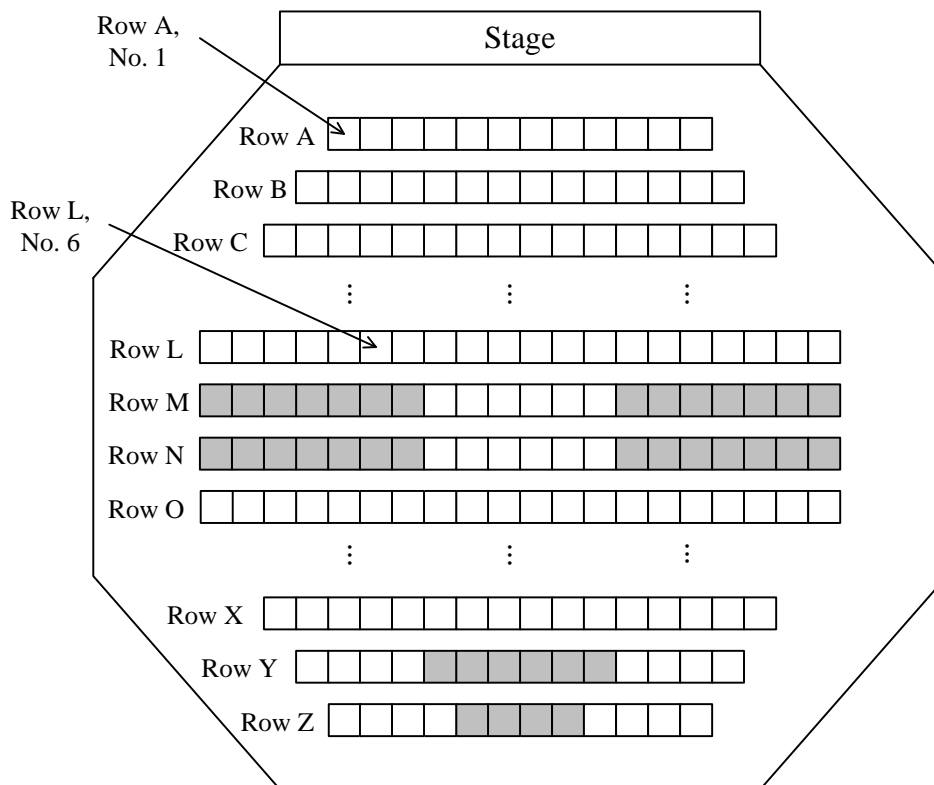


Fig. Seating Arrangement in the Concert Hall

- (i) There are 26 rows of seats coded A to Z. The number of seats per row varies from row to row. The number of seats in each row is stored in the global array `cnum` in the order of rows A to Z.
- (ii) A seat number consists of a row code and a number. The number indicates the position of the seat starting from the left side in each row when facing the stage. For example, seat "Row L, No. 6" is the 6th seat from the left when facing the stage in the 12th row (row L) up from the stage.
- (iii) The seats in the shaded area in the figure are already reserved.

- (2) The number of consecutive seats that are to be secured is stored in argument `number` and passed to the program.
- (3) The program starts searching for seats starting from the leftmost seat in the first row (row A, No. 1) as seen when facing the stage, seeking the desired number of consecutive seats in the same row, and secures the first available group of consecutive seats it finds. If the seats cannot be secured in row A, it sequentially searches from the leftmost seat in row B, C and so forth until finding the first available group of consecutive seats. If the desired number of consecutive seats cannot be found, it determines that the seats cannot be secured.
- (4) If the desired number of consecutive seats are secured, the program returns, as a return value, a pointer to the seat number structure (SEAT) which stores the row code where the seats are located, as well as the leftmost seat number (as viewed while facing the stage) among the secured seats. If not found, it returns a null pointer (NULL) as the returned value.
- (5) The declaration of the SEAT structure is as follows.

```
typedef struct {
    char row;    /* Row code */
    int no;      /* Number */
} SEAT;
```

- (6) The status of each seat in the concert hall is stored in the global variable `status`. If the seat is secured, that seat is marked as reserved.

$$\text{Status}[i][j] = \begin{cases} ' ': \text{Seat is available.} \\ 'R': \text{Seat is reserved} \end{cases}$$

The value of the index `i` corresponding to rows A through Z is 0 through 25. The value of the index `j` corresponding to numbers 1 through `n` is 0 through `n-1`.

[Program]

```

#define MAXNUM 30    /* Maximum number of seats in row */
#define ROWNUM 26    /* Number of rows */

typedef struct {
    char row;        /* Row code */
    int no;          /* Number */
} SEAT;

static char rname[] = {"ABCDEFGHIJKLMNOPQRSTUVWXYZ"},
            status[ROWNUM][MAXNUM];

static int cnum[ROWNUM] = {12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 30,
30, 30, 30, 30, 30, 30, 28, 26, 24, 22, 20, 18, 16, 14, 12};
static SEAT empty;

SEAT *book_seat(int);

SEAT *book_seat(int number)
{
    int ridx, cidx, eidx, flg = 0;

    for (ridx = 0; ridx < ROWNUM; ridx++) {
        for (cidx = 0; cidx <= cnum[ridx] - number; cidx++) {
            if (status[ridx][cidx] == ' ') {
                A;
                for (eidx = B; cidx < eidx; eidx--)
                    if (status[ridx][eidx] == 'R') {
                        flg = 0;
                        break;
                    }
                if (flg == 1) break;
            }
        }
        if (flg == 1) break;
    }

    if (flg == 0)
        return NULL;
    for (eidx = cidx + number - 1; cidx <= eidx; eidx--)
        status[ridx][eidx] = 'R';
    empty.row = C;
    empty.no = D;
    return &empty;
}

```

Subquestion

From the answer groups below, select the correct answers to be inserted in the blanks in the above program.

Answer group for A:

- | | |
|--------------------------------|------------------------------------|
| a) <code>cidx++</code> | b) <code>cidx--</code> |
| c) <code>cidx += number</code> | d) <code>cidx += number - 1</code> |
| e) <code>flg = 0</code> | f) <code>flg = 1</code> |

Answer group for B:

- | | |
|-----------------------------------|-------------------------------|
| a) <code>0</code> | b) <code>cidx</code> |
| c) <code>cidx + 1</code> | d) <code>cidx + number</code> |
| e) <code>cidx + number - 1</code> | f) <code>number</code> |
| g) <code>number - 1</code> | |

Answer group for C:

- | | |
|---------------------------------|---------------------------------|
| a) <code>ridx</code> | b) <code>ridx + 1</code> |
| c) <code>ridx - 1</code> | d) <code>rname[ridx]</code> |
| e) <code>rname[ridx + 1]</code> | f) <code>rname[ridx - 1]</code> |

Answer group for D:

- | | | |
|-----------------------------|-----------------------------|--------------------------|
| a) <code>cidx</code> | b) <code>cidx + 1</code> | c) <code>cidx - 1</code> |
| d) <code>ridx + cidx</code> | e) <code>ridx - cidx</code> | |

Question 2**C Program**

Q2. Read the following description of a C program and the program itself, and then answer Subquestions 1 and 2.

[Program 1 Description]

This program reads a nonempty source program written in C language from standard input, removes comments, and then outputs it to standard output.

(1) Description on the notation of source programs

- (i) "Comments" handled by this program are character strings that start with `/*` and end with `*/`, excluding those included in character constants, character string literals, and comments.
- (ii) The type of usable characters is as follows.

Space	0	@	P	`	p
!	1	A	Q	a	q
"	2	B	R	b	r
#	3	C	S	c	s
\$	4	D	T	d	t
%	5	E	U	e	u
&	6	F	V	f	v
'	7	G	W	g	w
(8	H	X	h	x
)	9	I	Y	i	y
*	:	J	Z	j	z
+	;	K	[k	{
,	<	L	\	l	
-	=	M]	m	}
.	>	N	^	n	~
/	?	O	_	o	

(iii) Description as shown below is not used.

- Nested comments

Example `/* aaaaa /* bbbbbb */ ccccc */`

- Three-character notation for graphic characters

`??= ??(??' ??< ??> ??) ??! ??-`

(iv) There are no grammatical errors.

- (2) Program 1 removes comments in accordance with the following procedure. Since the program simply processes the analyses of character constants, character string literals and comments, they may not be recognized correctly depending on the coding in source programs, resulting in a malfunction.
- (i) When detecting a single quote or a double quote, the program interprets it as the beginning of a character constant or character string literal, and then uses function quote to read and output the character string as it is until the program detects the corresponding single quote or double quote.
 - (ii) When detecting "/*", the program interprets it as the beginning of a comment and skips characters before the first appearance of "*/".
- (3) An execution example of the comment removal by Program 1 is shown below.

Input source program

```
/* This program uses fgets to output
 * a line from a file on the screen. */
#include <stdio.h>
int main( void )
{
    FILE *stream;    /* file pointer */
    char line[100]; /* input stream */

    if( (stream = fopen( "crt_fgets.txt", "r" )) != NULL )
    {
        if( fgets( line, 100, stream ) == NULL)
            printf( "fgets error\n" ); /* error message */
        else
            printf( "%s", line);
        fclose( stream );
    }
}
```

Output results after the removal of comments

```
#include <stdio.h>
int main( void )
{
    FILE *stream;
    char line[100];

    if( (stream = fopen( "crt_fgets.txt", "r" )) != NULL )
    {
        if( fgets( line, 100, stream ) == NULL)
            printf( "fgets error\n" );
        else
            printf( "%s", line);
        fclose( stream );
    }
}
```

Fig. Execution Example of the Comment Removal

[Program 1]

```

#include <stdio.h>
void quote( char );

main()
{
    int c1, c2;

    while ( (c1 = getchar()) != EOF ) {
        /* detection of single quote */
        if ( c1 == '\'' ) quote( '\'' );
        /* detection of double quote */
        else if ( c1 == '\"' ) quote( '\"' );
        /* detection of slash */
        else if ( c1 == '/' ) {
            c2 = getchar();
            /* when the next character is an asterisk */
            if ( c2 == '*' ) {
                /* removal of comment character string */
                while ( 1 ) {
                    while ( (c1 = getchar()) != '*' );
                    c2 = getchar();
                    if ( c2 == '/' ) break;
                }
            }
            /* other cases */
            else {
                putchar(c1);
                putchar(c2);
            }
        }
        else putchar(c1); /* one character read is outputted as it is */
    }
}

void quote( char c )
{
    /* extraction of character constant and character string literal */
    char cc;

    putchar(c);
    while ( (cc = getchar()) != c ) putchar(cc);
    putchar(cc);
}

```

Subquestion 1

From the answer group below, select the code that causes wrong operation when it is entered into program 1.

Answer group:

- a) `/* "aaaaaaa" */`
- b) `/* aaa 'a' */`
- c) `if (c == '\\') {`
- d) `printf(" \'");`
- e) `printf("aaa /* comment */ \n");`

[Program 2 Description]

To solve the problem pointed out in (2) of **[Program 1 Description]**, Program 2 is then written as follows.

- (1) The process is divided into three modes: character constant, character string literal, and comment.
- (2) An appearance of a single quote switches the "character constant mode" between ON and OFF. However, this does not apply to a piece of code which is an expanded representation using a "\", to a piece of code within a character string literal, or to a piece of code within a comment.
- (3) An appearance of a double quote switches the "character string literal mode" between ON and OFF. However, this does not apply to a piece of code which is an expanded representation using a "\", to a piece of code within a character constant, or to a piece of code within a comment.
- (4) An appearance of "/*" and "*/" switches the "comment mode" between ON and OFF. However, this does not apply to a piece of code within a character constant or to a piece of code within a character string literal.

[Program 2]

```

#include <stdio.h>
main()
{
    int c1, c2;
    int c_mode = 0; /* initialize comment mode to off */
    int quotel = 0; /* initialize character constant mode to off */
    int quote2 = 0; /* initialize character string literal mode to off */

    for ( c1 = getchar(); ( c2 = getchar() ) != EOF; c1 = c2 ) {

        if ( !c_mode ) { /* when comment mode is off */
            /* detecting if \ is in a character constant or a character string literal.*/
            if ( A && c1 == '\\\' ) {
                putchar(c1);
                putchar(c2);
                c2 = getchar();
                continue;
            }
            /* detecting if single quote is not inside a character string literal */
            else if ( !quote2 && c1 == '\'' )
                B;
            /* detecting if double quote is not inside a character constant */
            else if ( !quotel && c1 == '\"' )
                C;
            /* detecting if / and * are not inside a character constant */
            /* and not inside character string literal */
            else if ( D && c1 == '/' && c2 == '*' ) {
                E;
                c2 = getchar();
                continue;
            }
            putchar(c1);
        }

        else {
            if ( c1 == '*' && c2 == '/' ) { /* end of comment? */
                E;
                c2 = getchar();
            }
        }
        putchar(c1);
    }
}

```

Subquestion 2

From the answer groups below, select the correct answers to be inserted into the blanks

in Program 2.

Answer group for A and D:

- | | |
|--------------------------------------|--|
| a) <code>!quote1</code> | b) <code>!quote2</code> |
| c) <code>(!quote1 !quote2)</code> | d) <code>(!quote1 && !quote2)</code> |
| e) <code>(quote1 quote2)</code> | f) <code>(quote1 && quote2)</code> |

Answer group for B, C and E:

- | | |
|----------------------------------|---|
| a) <code>c_mode = !c_mode</code> | b) <code>c_mode = quote1 && quote2</code> |
| c) <code>quote1 = !quote1</code> | d) <code>quote1 = !quote2</code> |
| e) <code>quote1 = quote2</code> | f) <code>quote2 = !quote1</code> |
| g) <code>quote2 = !quote2</code> | h) <code>quote2 = quote1</code> |

Question 3**C Program**

Q3. Read the following description of a C program and the program itself, and then answer Subquestion.

[Program Description]

Function `execute` draws lines using the marker displayed on the screen .

- (1) A bitmap screen has 800 pixels in the horizontal direction and 600 pixels in the vertical direction. Figure 1 shows the screen's coordinate system. A marker (• in Fig. 1) with positional and directional information is shown on the screen. The marker moves in four directions: up, down, left, or right. The locus of the marker is drawn when it is moved.

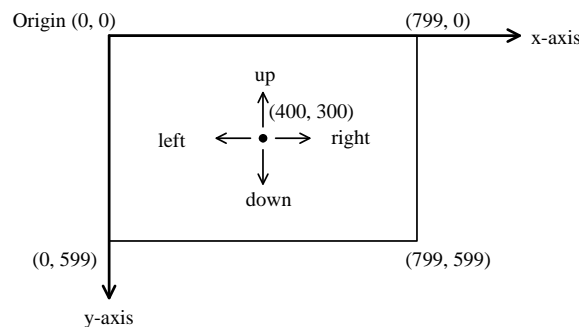


Fig. 1 Screen coordinate system and initial marker state

- (2) The marker is represented by `MARKER` type structure `mark`. When the program starts, the marker position is set to (400, 300) and its direction of movement is upward.

```
typedef struct {
    int x; /* x coordinate of marker */
    int y; /* y coordinate of marker */
    int dir; /* Marker direction 0:right, 1:up, 2:left, 3:down */
} MARKER;

MARKER mark = {400, /* Initial x coordinate of marker */
               300, /* Initial y coordinate of marker */
               1 /* Initial direction of marker (up) */
               };
```

- (3) Instructions for operating the marker are defined. Each instruction consists of an instruction code and a value, and is expressed by the structure `INST`.

```
typedef struct {
    char code; /* Instruction code */
    int val; /* value */
} INST;
```

The instructions are stored in the order of execution from the beginning of `insts`, which is an array of the structure `INST`.

- (4) The table below lists instruction codes and the descriptions.

Instruction code	Description
{	Repeat instruction execution <code>val</code> times from the next instruction to the last one before the instruction code <code>'}'</code> that forms a pair. <code>val</code> is an integer greater than 1.
t	Change the direction of movement of the marker by $90^\circ \times val$ only in the counterclockwise direction. <code>val</code> is a nonnegative integer.
f	Move the marker in the current direction by <code>val</code> pixels to draw a line from the source to the destination. <code>val</code> is any integer.
}	Indicates the end of a series of instructions to be repeated. <code>val</code> is not referenced.
\0	Indicates the end of the instructions. <code>val</code> is not referenced.

- (5) Figure 3 shows the output of function `execute` executed with the instructions stored in the structure array `insts`, as shown in Fig. 2. Note, however, that the coordinate values in Fig. 3 are added for the sake of explanation and are not actually outputted.

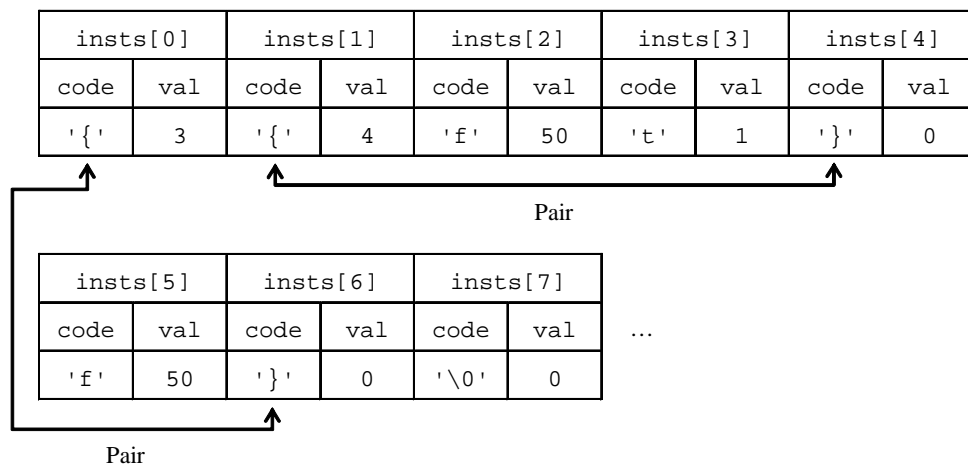


Fig. 2 Example of instructions stored in the structure array `insts`

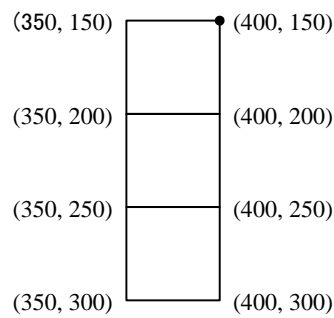


Fig. 3 Results of executing the example in Fig. 2

- (6) The following functions are available for drawing lines:

```
void drawLine( int x1, int y1, int x2, int y2 );
```

Features: Draws a line segment connecting coordinate (x1, y1) and coordinate (x2, y2) that is in the screen area.

- (7) The following functions related to the marker are available:

```
void eraseMarker( MARKER mark );
```

Feature: Do not display the marker when it is in the screen area.

```
void paintMarker( MARKER mark );
```

Feature: Display the marker when it is in the screen area.

- (8) Even if the marker has moved out of the screen area and is no longer visible, its position coordinates and direction of movement are retained.

[Program]

```
#define INSTSIZE 100 /* Upper limit of number of instructions */
#define STACKSIZE 50 /* Upper limit of nesting */

typedef struct {
    int x;          /* x coordinate of marker */
    int y;          /* y coordinate of marker */
    int dir;        /* Marker direction 0:right 1:up 2:left 3:down */
} MARKER;

typedef struct {
    char code;      /* Instruction code */
    int val;        /* value */
} INST;

typedef struct {
    int opno;       /* Element No. of array insts at which the start of loop is defined */
    int rest;       /* Remaining loop count */
} STACK;
```

```

void drawLine( int, int, int, int );
void eraseMarker( MARKER );
void paintMarker( MARKER );

INST insts[INSTSIZE];    /* Structure array for storing instructions */
MARKER mark = {400,      /* Initial x coordinate of marker */
               300,      /* Initial y coordinate of marker */
               1          /* Initial direction of marker (up) */
               };

void execute(){
    STACK stack[STACKSIZE];
    int opno = 0;    /* Element No. of array insts which contains the instruction to be executed */
    int spt = -1;    /* Stack pointer */
    int dx, dy;

    paintMarker( mark );
    while( insts[opno].code != '\0' ){
        switch( insts[stack].code ){
            case '{':
                stack[ A ].opno = opno;
                stack[spt].rest = insts[opno].val;
                break;
            case 't':
                mark.dir = B;
                break;
            case 'f':
                eraseMarker( mark );
                dx = ( mark.dir % 2 == 0 ) ? C;
                dy = ( mark.dir % 2 == 0 ) ? D;
                drawLine( mark.x, mark.y,
                           mark.x + dx, mark.y + dy );
                mark.x += dx;
                mark.y += dy;
                paintMarker( mark );
                break;
            case '}':
                if ( stack[spt].rest E ){
                    opno = stack[spt].opno;
                    stack[spt].rest--;
                } else {
                    F;
                }
                break;
        }
        G;
    }
}

```


Subquestion

From the answer groups below, select the correct answers to be inserted in the blanks in the above program.

Answer group for A:

- | | | |
|----------|----------|--------|
| a) ++spt | b) --spt | c) spt |
| d) spt++ | e) spt-- | |

Answer group for B:

- a) (mark.dir + insts[opno].val) % 2
- b) (mark.dir + insts[opno].val) % 3
- c) (mark.dir + insts[opno].val) % 4
- d) mark.dir + insts[opno].val
- e) mark.dir + insts[opno].val % 2
- f) mark.dir + insts[opno].val % 3
- g) mark.dir + insts[opno].val % 4

Answer group for C and D:

- a) (1 - mark.dir) * insts[opno].val : 0
- b) (2 - mark.dir) * insts[opno].val : 0
- c) (mark.dir - 1) * insts[opno].val : 0
- d) (mark.dir - 2) * insts[opno].val : 0
- e) 0 : (1 - mark.dir) * insts[opno].val
- f) 0 : (2 - mark.dir) * insts[opno].val
- g) 0 : (mark.dir - 1) * insts[opno].val
- h) 0 : (mark.dir - 2) * insts[opno].val
- i) 0 : mark.dir * insts[opno].val
- j) mark.dir * insts[opno].val : 0

Answer group for E:

- | | | |
|--------|--------|---------|
| a) < 0 | b) < 1 | c) == 0 |
| d) > 0 | e) > 1 | |

Answer group for F and G:

- | | | |
|---------------|---------------|-----------|
| a) mark.dir++ | b) mark.dir-- | c) opno++ |
| d) opno-- | e) spt++ | f) spt-- |

Question 4

Java Program

Q4. Read the following description of a Java program and the program itself, and then answer Subquestion.

[Program Description]

The program consists of the following: a) the class `Encoder`, which converts byte string of binary data according to a certain algorithm and obtains the result as a character string; b) a test class for the class `Encoder`. The conversion algorithm is as follows.

- (1) It extracts 6 bits at a time starting from the top of the byte sequence and converts it into a corresponding character in the conversion table. If the number of bytes is not a multiple of 3, then there will be 4 bits or 2 bits of zeros at the end, so that the number of bits will be a multiple of 6.
- (2) As many "=" as necessary are added to the end of the converted character string so that the number of characters is the smallest multiple of 4 equal to or greater than $\frac{4}{3}$ times of the number of bytes given.
- (3) A conversion example is shown in Figure 1 below.

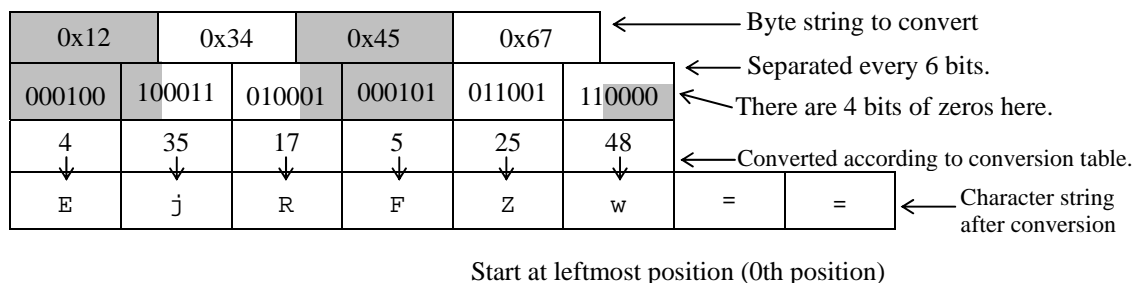


Fig. 1 Conversion Example

- (4) The class `Encoder` has two methods: `next` and `hasNext`. The `next` method returns the converted characters one by one in sequence, according to the conversion algorithm. If it is called when there are no characters to return, `java.util.NoSuchElementException` is thrown.

The `hasNext` method is true if a character can be returned without generating an exception the next time that the `next` method is called.

- (5) The variable `CHARS` is a table that converts 6-bit numeric values into a single character, and the size of its array is 64. The variable `n` holds information indicating the sequential position of the character to be returned the next time the `next` method is called. (position of the leftmost character is 0.)

The variable `bin` holds the binary data string to be converted.

- (6) The execution result of the test class (EncoderTest) for the byte string in Figure 1 is shown in Figure 2.

EjRFZw==

Fig. 2 Execution Results

[Program]

```
public class Encoder {
    static final char[] CHARS = ("ABCDEFGHIJKLMNOPQRSTUVWXYZ" +
        "abcdefghijklmnopqrstuvwxyz0123456789+/").toCharArray();
    private int n = 0;
    private byte[] bin;
    public Encoder(byte[] bin) {
        if (bin == null) this.bin = new byte[0];
        else this.bin = bin;
    }
    public boolean hasNext() {
        return n < A;
    }
    public char next() {
        char letter;
        int pos = (int)(n * 0.75);
        // Acquires 2 bytes of data containing the 6-bit data targeted for conversion
        if (pos < bin.length) {
            int cell = bin[pos++] << 8;
            if (pos < bin.length) cell += bin[pos] & 255;
            // obtains 6 bits to be converted and converts them into the corresponding character
            letter = CHARS[(cell >> (n + 3) % 4 * 2 + 4) & 63];
        } else {
            if (B)
                throw new java.util.NoSuchElementException();
            else letter = C;
        }
        n++;
        return letter;
    }
}

class EncoderTest {
    public static void main(String[] args) {
        byte[] bin = {0x12, 0x34, 0x45, 0x67};
        Encoder encoder = new Encoder(bin);
        while (encoder.hasNext()) {
            System.out.print(encoder.next());
        }
        System.out.println();
    }
}
```

Subquestion

From the answer groups below, select the correct answers to be inserted in the blanks in the above program.

Answer group for A:

- | | |
|--|--|
| a) <code>(bin.length + 2) * 4 / 3</code> | b) <code>(bin.length + 2) / 3 * 4</code> |
| c) <code>bin.length * 4 / 3</code> | d) <code>bin.length / 3 * 4</code> |

Answer group for B:

- | | |
|-----------------------------------|------------------------------------|
| a) <code>!hasNext()</code> | b) <code>hasNext()</code> |
| c) <code>n < bin.length</code> | d) <code>n >= bin.length</code> |

Answer group for C:

- | | |
|--------------------------|-------------------------|
| a) <code>' '</code> | b) <code>'='</code> |
| c) <code>CHARS[n]</code> | d) <code>next()</code> |

Question 5**Java Program**

Q5. Read the following description of a Java program and the program itself, and then answer Subquestion.

[Program Description]

Interface `CharIterator` defines an operation to extract characters (`char`) from its instance in order, independent of the data structure. In `CharIterator`, the following methods are defined:

```
public char next()
```

If the next character exists, it is returned. If it does not exist, `java.util.NoSuchElementException` is thrown. If, for example, the `CharIterator` has n characters, the first character is returned for the first call, and the second character is returned for the second call. In this way, characters are returned in order until the n -th call is made. For the $n+1$ -th or a subsequent call, `NoSuchElementException` is thrown.

Note that, `NoSuchElementException` is a subclass of `java.lang.RuntimeException`.

```
public boolean hasNext()
```

If the next character exists, `true` is returned. If it does not exist, `false` is returned.

Class `CharIteratorFactory` defines a method that returns `CharIterator` that matches the data type specified as the argument.

In `CharIteratorFactory`, the following class methods are defined:

```
public static CharIterator getCharIterator(String data)
```

`CharIterator`, which extracts characters in order from the `String` specified as an argument, is returned. If a `null` argument is specified, `NullPointerException` is thrown.

```
public static CharIterator getCharIterator(char[][] data)
```

`CharIterator`, which extracts characters in order from the array of arrays (two dimensional character array) whose elements' type is `char` specified as the argument, is returned. If a `null` argument is specified, `NullPointerException` is thrown.

Characters are to be extracted in ascending order of index values of character array and its array. That is, for m , whose type is `char[][]`, their elements $m[i][j]$ are extracted in ascending order of i , and for the same ' i ', in ascending order of j .

Class CharIteratorTest is a program that tests methods defined in CharIteratorFactory. The execution result of method main is shown in the figure below:

'H'	'1'	'6'
'2'	'0'	'0' '4'

Fig. Execution Results of CharlteratorTest.main

[Program 1]

```
public interface CharIterator {
    public boolean hasNext();
    public char next();
}
```

[Program 2]

```
import java.util.NoSuchElementException;

public class CharIteratorFactory {
    public static CharIterator getCharIterator(String data) {
        if (data == null)
            throw new NullPointerException();
        // Generates and returns an instance of CharIterator that
        // returns characters in order from String data.
        return A;
    }

    public static CharIterator getCharIterator(char[][] data) {
        if (data == null)
            throw new NullPointerException();
        // Generates and returns an instance of CharIterator that
        // returns characters in order from two dimensional character array.
        return B;
    }
}
```

```

class StringCharIterator implements CharIterator {
    private String data;
    private int index = 0;

    StringCharIterator(String data) {
        this.data = data;
    }
    // Check whether or not the next characters exist in data.
    public boolean hasNext() {
        return C;
    }

    public char next() {
        // If the next characters do not exist, throw NoSuchElementException.
        if (index >= data.length())
            throw new NoSuchElementException();
        // Returns the character next to data and updates index value.
        return D;
    }
}

```

```

class Char2DArrayCharIterator implements CharIterator {
    private char[][] data;
    private int index1 = 0, index2 = 0;

    Char2DArrayCharIterator(char[][] data) {
        this.data = data;
    }
    public boolean hasNext() {
        // If an element of data[index1][index2] exists, true is returned.
        // If it does not exist, the next element defined is searched.
        // If the next element does not exist, false is returned.
        for (; index1 < data.length; index1++) {
            if (data[index1] != null
                && index2 < data[index1].length) {
                return true;
            }
            E;
        }
        return false;
    }
    public char next() {
        // Calls method hasNext to check for next element. If an element exists,
        // the element is returned, and the index value is updated.
        // If no element exists, NoSuchElementException is thrown.
        if (hasNext()) {
            return F;
        }
        throw new NoSuchElementException();
    }
}

```

[Program 3]

```

public class CharIteratorTest {
    public static void main(String[] args) {
        CharIterator itr =
            CharIteratorFactory.getCharIterator("H16");
        printIterator(itr);

        itr = CharIteratorFactory.getCharIterator(
            new char[][] { { '2' },
                          { '0' },
                          null,
                          { '0', '4' } });
        printIterator(itr);
    }
    private static void printIterator(CharIterator itr) {
        while (itr.hasNext()) {
            System.out.print("'" + itr.next() + "' ");
        }
        System.out.println();
    }
}

```

Subquestion

From the answer groups below, select the correct answers to be inserted in the blanks in the above program.

Answer group for A and B:

- a) getCharIterator((char[][]) data)
- b) getCharIterator((String) data)
- c) new Char2DArrayCharIterator()
- d) new Char2DArrayCharIterator(data)
- e) new StringCharIterator()
- f) new StringCharIterator(data)

Answer group for C:

- | | |
|-----------------------------|-----------------------------|
| a) index < data.length() | b) index <= data.length() |
| c) index >= data.length() | d) index++ < data.length() |
| e) index++ <= data.length() | f) index++ >= data.length() |

Answer group for D:

- | | |
|--|--------------------------------------|
| a) <code>data.charAt(++index)</code> | b) <code>data.charAt(--index)</code> |
| c) <code>data.charAt(index + 1)</code> | d) <code>data.charAt(index++)</code> |
| e) <code>data.charAt(index--)</code> | f) <code>data.charAt(index)</code> |

Answer group for E:

- | | |
|--------------------------------------|--------------------------------------|
| a) <code>index1 = 0</code> | b) <code>index1 = data.length</code> |
| c) <code>index1 = index2</code> | d) <code>index2 = 0</code> |
| e) <code>index2 = data.length</code> | f) <code>index2 = index1</code> |

Answer group for F:

- | | |
|--|--|
| a) <code>data[index1++][index2++]</code> | b) <code>data[index1++][index2]</code> |
| c) <code>data[index1][++index2]</code> | d) <code>data[index1][--index2]</code> |
| e) <code>data[index1][index2 + 1]</code> | f) <code>data[index1][index2++]</code> |

Question 6

Java Program

Q6. Read the following description of a Java program and the program itself, and then answer Subquestions 1 and 2.

[Program Description]

This program is a simulator that simulates the call center operation of Company A. Company A uses this simulator to estimate the time required for a user to connect to an operator when making a call to the call center. A call made to the call center is always responded by a single operator.

Assume for this simulator that as long as an operator is available, no caller has to wait. For instance, assume six calls were made to the call center at 60-second intervals when two operators were available. The duration of each call was 130 seconds, 100 seconds, 150 seconds, 90 seconds, 110 seconds, and 140 seconds respectively. This means that the wait time of the third user was 10 seconds, the fifth user 30 seconds, and the others 0 seconds. (See Figure 1.)

Note that this simulator simulates one second in the real world in 0.1 second.

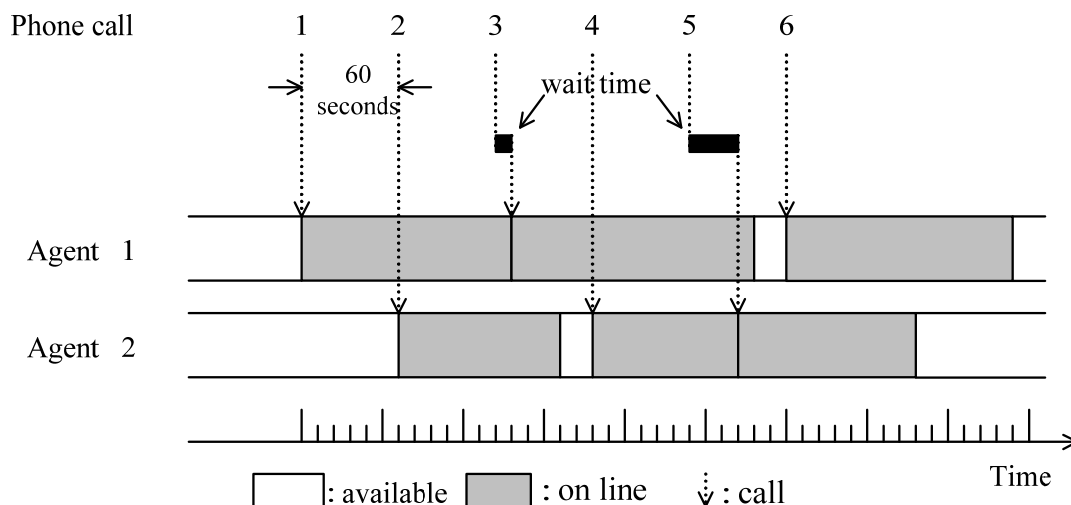


Fig. 1 Users' wait time

This program consists of the following classes:

(1) `CallCenter`

This class represents the call center. It performs simulation after specifying the number of operators, array of talk times, and intervals that generate calls in the arguments of the constructor. It consists of the methods and internal classes described below.

```
public static void main(String[] args)
```

This method starts the simulator for testing.

```
Call answer()
```

This method is called by an operator to respond to a call. It returns a `Call` object to the operator. This method makes the operator thread wait until a call can be allocated to the operator. If there remains no call to be allocated after all calls generated by this class have been allocated, it returns `null`.

```
Operator
```

This thread class simulates the operator. Each operator is processed in an individual thread. It responds to a call to the call center and converses. It repeats the processing until all the calls generated by `CallCenter` have been allocated.

(2) Call

This class represents a call from a caller. The duration of talk time is specified in the argument of the constructor. This class consists of the methods described below.

```
public void talk()
```

This method simulates the status during the talk between an operator and the caller. This method displays the wait time of a caller allocated to the operator in second units (rounded off), and then stops the operator thread only for the time specified as talk time.

Figure 2 shows the results of the program when the example of Figure 1 was simulated.

Note that (s) represents seconds.

0(s)
0(s)
10(s)
0(s)
30(s)
0(s)

Fig. 2 Results of simulation

`java.util.Vector` in the program is a class that represents a variable-length array, and this class consists of the following methods:

```
public boolean add(Object obj)
```

It appends element `obj` at the end of the array.

```
public Object remove(int index)
```

It deletes the element at the position specified by `index` from the array, and returns the deleted element. The position of the first element is 0. The elements after `index` are shifted to the front of the array.

```
public boolean isEmpty()
```

If there is no element, this method returns `true`. Otherwise, it returns `false`.

[Program 1]

```

import java.util.Vector;

public class CallCenter {
    private final Vector waitingList = new Vector();
    private boolean running; // true when call generated

    public static void main(String[] args) {
        int op = 2;          // Number of operators
        // Talk time (seconds)
        long[] duration = {130, 100, 150, 90, 110, 140};
        long interval = 60; // Interval for generating call (seconds)
        new CallCenter(op, duration, interval);
    }

    public CallCenter(int op, long[] duration, long interval) {
        running = true;
        // Generate an operator thread and start it.
        for (int i = 0; i < op; i++) new Operator().start();
        long nextCallTime = System.currentTimeMillis();
        for (int i = 0; i < duration.length; i++) {
            // Generate a call and add it to list.
            synchronized (waitingList) {
                waitingList.add(new Call(duration[i]));
                waitingList.notify();
            }
            // Wait until next call is generated.
            nextCallTime += interval * 100; // Operates 10 times faster.

            long sleeping = A;
            try {
                if (sleeping > 0) Thread.sleep(sleeping);
            } catch (InterruptedException ie) {}
        }
        // End all operator threads.
        running = false;
        B {
            waitingList.notifyAll();
        }
    }
}

```

```

Call answer() {
    synchronized (waitingList) {
        while (waitingList.isEmpty() C) {
            try {
                D;
            } catch (InterruptedException ie) {}
        }
        if (waitingList.isEmpty()) return null;
        return (Call)waitingList.remove(0);
    }
}

class Operator extends Thread {
    public void run() {
        Call call;
        E call.talk();
    }
}

```

[Program 2]

```

public class Call {
    private final long start, duration;
    public Call(long duration) {
        this.duration = duration;
        start = System.currentTimeMillis();
    }
    public void talk() {
        long elapsed = System.currentTimeMillis() - start;
        // Display elapsed time after rounding it off.
        System.out.println(F + "(s)");
        try {
            Thread.sleep(duration * 100); // Operates 10 times faster.
        } catch (InterruptedException ie) {}
    }
}

```

Subquestion 1

From the answer groups below, select the correct answers to be inserted in the blanks in the above programs.

Answer group for A:

- a) `nextCallTime`
- b) `nextCallTime - duration[i] * 100`
- c) `nextCallTime - System.currentTimeMillis()`
- d) `System.currentTimeMillis() - nextCallTime`

Answer group for B:

- a) `for (int i = 0; i < op; i++)`
- b) `if (waitingList != null)`
- c) `synchronized (this)`
- d) `synchronized (waitingList)`

Answer group for C:

- a) `&& !running`
- b) `&& running`
- c) `== !running`
- d) `|| running`

Answer group for D:

- a) `notify()`
- b) `wait()`
- c) `waitingList.notify()`
- d) `waitingList.wait()`

Answer group for E:

- a) `if ((call = answer()) != null)`
- b) `if (answer() != null)`
- c) `while ((call = answer()) != null)`
- d) `while (answer() != null)`

Answer group for F:

- a) `(elapsed + 50) / 100`
- b) `(elapsed + 500) / 100`
- c) `(elapsed - 50) / 100`
- d) `(elapsed - 500) / 100`

Subquestion 2

A `CallCenter` instance was generated, as shown below, from the answer group below, select the number of operators occupied after 80 seconds (8 seconds in real time) have elapsed in the simulator.

```
new CallCenter(3, new long[]{70, 90, 100, 110}, 30);
```

Answer group:

- a) 0 b) 1 c) 2 d) 3