October 2017

Fundamental IT Engineer Examination (Afternoon)

Questions must be answered in accordance with the following:

Question Nos.	Q1 – Q6	Q7 , Q8
Question Selection	Compulsory	Select 1 of 2
Examination Time	13:30 - 16:00 (150 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) **Question Selection**
   For **Q7** and **Q8**, mark the ⓢ of the question you select to answer in the "Selection Column" on your answer sheet.

   (4) **Answers**
   Mark your answers as shown in the following sample question.

[Sample Question]
In which month is the autumn Fundamental IT Engineer Examination conducted?

Answer group
a) September b) October c) November d) December

Since the correct answer is " b) October ", mark your answer sheet as follows:

[Sample Answer]

Sample	ⓐ ● ⓒ ⓓ ⓔ ⓕ ⓖ ⓗ ⓘ ⓙ

Do not open the exam booklet until instructed to do so.
Inquiries about the exam questions will not be answered.

Notations used for pseudo-language

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

[Declaration, comment, and process]

Notation		Description
○		Declares names, types, etc., of procedures, variables, etc.
/* text */		Describes comments in the text.
Process	• variable ← expression	Assigns the value of the expression to the variable.
	• procedure(argument, ...)	Calls the procedure and passes / receives the argument.
	▲ conditional expression process ▼	Indicates a one-way selection process. If the conditional expression is `true`, then the process is executed.
	▲ conditional expression process 1 ─ process 2 ▼	Indicates a two-way selection process. If the conditional expression is `true`, then process 1 is executed. If it is `false`, then process 2 is executed.
	■ conditional expression process ■	Indicates a pre-test iteration process. While the conditional expression is `true`, the process is executed repeatedly.
	■ process ■ conditional expression	Indicates a post-test iteration process. The process is executed, and then while the conditional expression is `true`, the process is executed repeatedly.
	■ variable: init, cond, incr process ■	Indicates an iteration process. The initial value init (given by an expression) is stored in the variable at the start of the iteration process, and then while the conditional expression cond is `true`, the process is executed repeatedly. The increment incr (given by an expression) is added to the variable in each iteration.

[Logical constants]

`true`, `false`

(continued on next page)

[Operators and their priorities]

Type of operation	Operator	Priority
Unary operation	`+, −, not`	High
Multiplication, division	`×, ÷, %`	↑
Addition, subtraction	`+, −`	
Relational operation	`>, <, ≥, ≤, =, ≠`	
Logical product	`and`	↓
Logical sum	`or`	Low

Note: With division of integers, an integer quotient is returned as a result.
The “%” operator indicates a remainder operation.

Questions **Q1** through **Q6** are all **compulsory**. Answer every question.

Q1. Read the following description of authenticity and confidentiality of e-mail data, and then answer Subquestions 1 through 3.

Electronic mail, i.e. e-mail, is the most extensively used network-based distributed application. E-mails may be sent and received to and from various system architectures, host operating systems, vendor platforms and communication suites, rendering it an effective tool for communication. E-mails are likely to contain critical information that is to be secured. The security of an e-mail is maintained by ensuring its authenticity and confidentiality.

PGP (Pretty Good Privacy) is a remarkable technique to check the authenticity of e-mail data. This technique provides authentication services to an e-mail by incorporating a digital signature in it.

At the sender side (Figure 1), PGP generates the digital signature from the e-mail body using SHA-1 (Secure Hash Algorithm-1) according to the following steps:

(1) SHA-1 generates a 160-bit hash code of the message.

(2) The hash code is encrypted with the RSA algorithm using the sender's private key.

(3) The result is appended to the message.

The message is then compressed at a ratio of approximately 2.0 using the ZIP algorithm. The compressed message enables the effective use of transmission and storage resources.

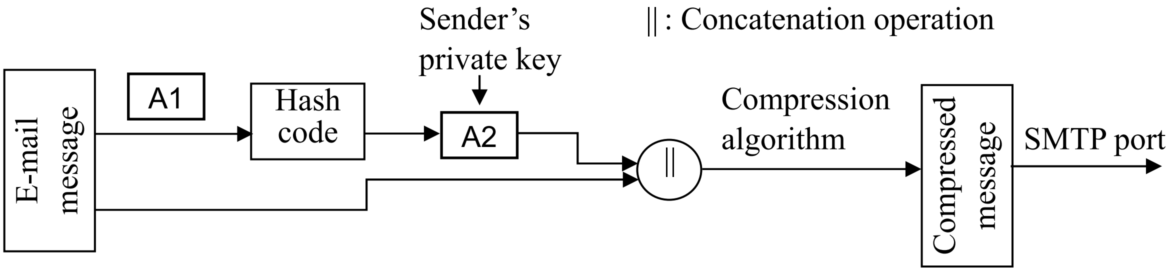

Figure 1 Authentication operations at the sender side

At the receiver side (Figure 2), the digital signature is separated from the incoming e-mail message after the message is decompressed. Further operations proceed according to the following steps:

(1) Decrypt the hash code from the digital signature by using the sender's public key.

(2) Generates a new hash code from the received e-mail message.

(3) If the two hash codes match, the message is verified to be authentic.

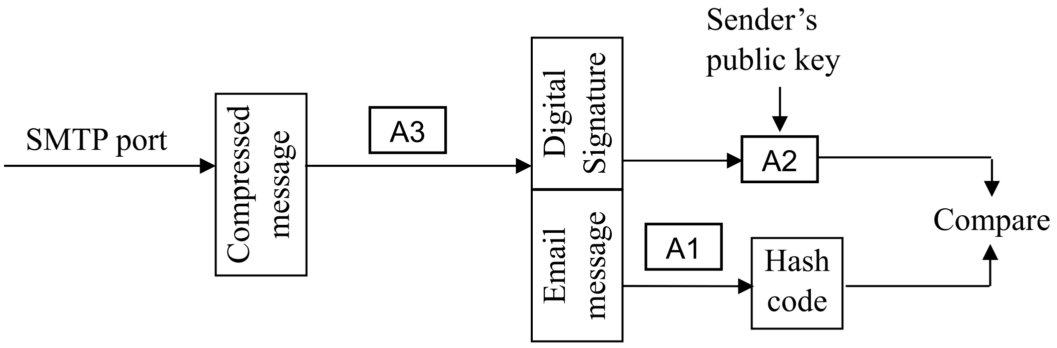

Figure 2 Authentication operations at the receiver side

Subquestion 1

From the answer group below, select the correct answer to be inserted in each blank □ in Figure 1 and Figure 2. Here, the answers to be inserted in A1, A2 and A3 should be selected as the correct combination from the answer group for A.

Answer group for A

	A1	A2	A3
a)	RSA algorithm	Compression algorithm	ZIP algorithm
b)	RSA algorithm	Decompression algorithm	ZIP algorithm
c)	SHA-1	RSA algorithm	Compression algorithm
d)	SHA-1	RSA algorithm	Decompression algorithm
e)	ZIP algorithm	SHA-1	RSA algorithm
f)	ZIP algorithm	SHA-1	RSA algorithm

Subquestion 2

From the answer group below, select the correct answer to be inserted in each blank □ in Figure 3.

In PGP, a symmetric 128-bit session key is generated and used each time a new e-mail is sent. The key is bound to the message and transmitted with it. The effectiveness of this method is limited to ensuring the possessor's authenticity; it does not ensure the confidentiality of the e-mail.

For ensuring the confidentiality of e-mail contents, the symmetric encryption algorithm may be used to encrypt an e-mail message in 64-bit cipher feedback mode. The session key is encrypted by using the receiver's public key. Figure 3 shows the operations for ensuring both authenticity and confidentiality of the e-mail body.

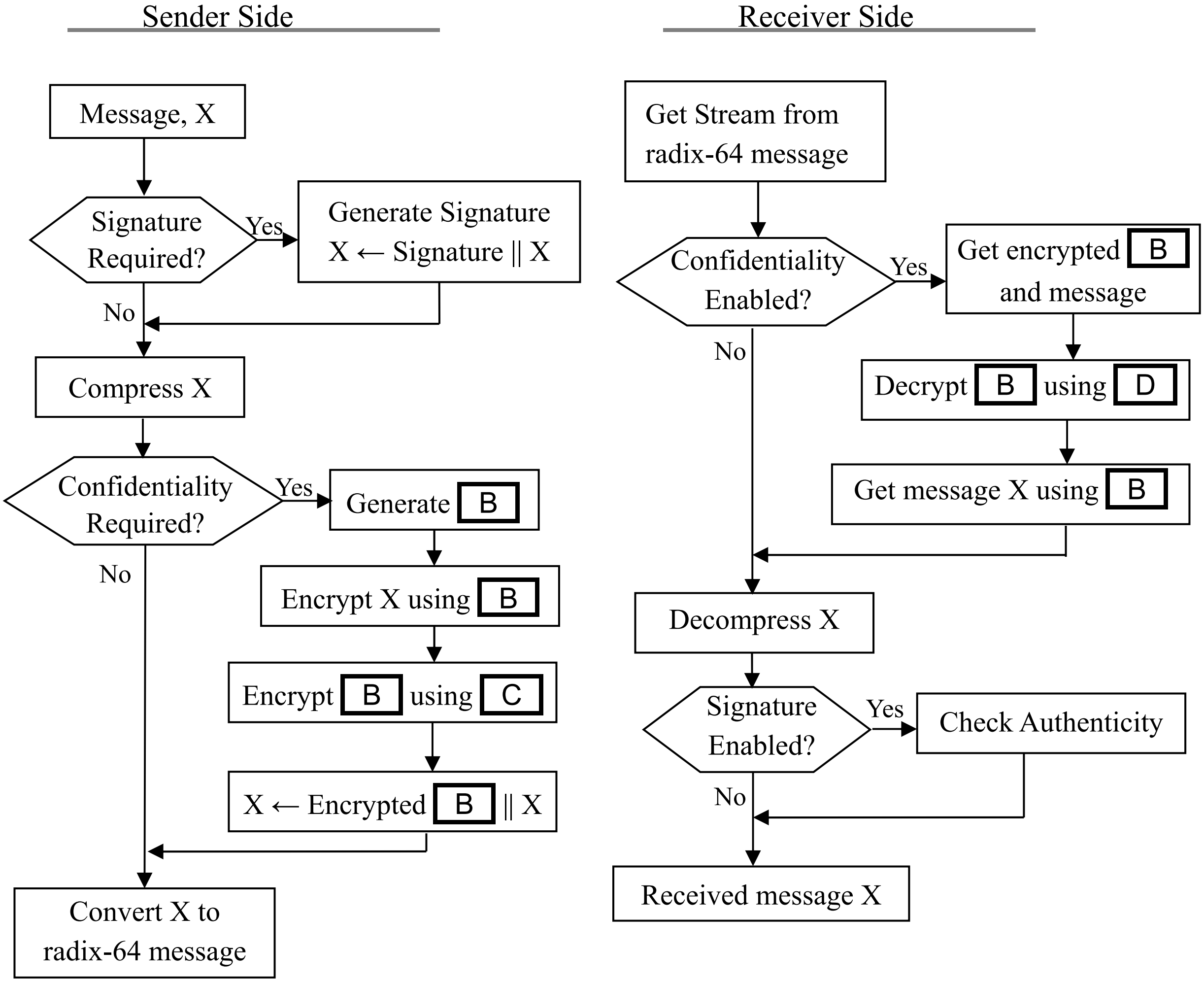

Figure 3 Operations for ensuring both authenticity and confidentiality of the e-mail body

Answer group for B through D

a) Compression key
b) Hash key
c) Receiver's private key
d) Receiver's public key
e) Sender's private key
f) Sender's public key
g) Session key

Subquestion 3

From the answer group below, select the correct answer to be inserted in the blank $\boxed{\quad\quad}$ in the following description.

The digital signature is generated from the plaintext rather than the encrypted or compressed message. This ensures that for the purpose of signature verification, a third-party need not be concerned with the symmetric key or compression algorithm. Message encryption is applied after compression in order to strengthen cryptography security because the compressed message exhibits less redundancy than the original plaintext. Less redundancy renders it more challenging for a hacker to perform cryptanalysis process.

The resulting encrypted message at the sender side is a stream of 8-bit binary codes. However, numerous e-mail systems permit the use of only ASCII characters. To accommodate this restriction, PGP provides the service of converting the 8-bit binary stream to a stream of ASCII characters using radix-64 conversion. Each group of three 8-bit binary codes is mapped into four ASCII characters, expanding the message by 33%.

For example, when a sender sends an e-mail consisting of 9k ASCII characters, it is transmitted over the Internet as an 8-bit binary stream of size $\boxed{\quad\text{E}\quad}$ kB. Assume that the ZIP algorithm exhibits a compression ratio of 2.0 and that the sizes of the digital signature and encrypted session key are negligible.

Answer group for E

a) 4.5
b) 6
c) 8
d) 9
e) 12
f) 16

Q2. Read the following description of shared resource access, and then answer Subquestion.

In a multi-programming operating system, there is a possibility of accessing the same resource by two or more processes simultaneously. The mutually exclusive control ensures that two or more concurrent processes do not enter their critical sections simultaneously. A critical section is a segment of a process that cannot be executed concurrently by two or more processes.

It is necessary for the operating system to ensure that:

(1) if one process is accessing a shared resource such as modifiable data, the other processes are excluded from using that resource,

(2) each process receives enough processor resource to function effectively, and

(3) the processor makes an effort to minimize its idle status.

[Mutually exclusive control - Case 1]

In a processor, two processes P0 and P1 are running concurrently. Both P0 and P1 attempt to update the shared data. The processor maintains data consistency by permitting only one process to update the shared data, and preventing the other process from accessing the shared data while the permitted process is updating that data.
A global variable `turn`, shared by P0 and P1, is used for mutually exclusive control. The initial value of `turn` is 0.
Figure 1 shows the mutually exclusive control for P0 and P1.

```
PROCESS P0 {
   ...
   WHILE (turn is not 0) {
      WAIT for a predefined time unit
   }
   UPDATE shared data
   SET turn ← 1
   ...
}
```

```
PROCESS P1 {
   ...
   WHILE (turn is not 1) {
      WAIT for a predefined time unit
   }
   UPDATE shared data
   SET turn ← 0
   ...
}
```

Figure 1 Mutually exclusive control for P0 and P1

Table 1 shows, as an example, the functioning of this mutually exclusive control.

Table 1 Execution status of P0 and P1 in time sequence

Time	Execution status of P0 and P1
t	• P0 and P1 are loaded and started. The initial value of `turn` is 0. • Both P0 and P1 attempt to access the shared data. • Then A enters into its critical section and accesses the shared data.
t+1	• P1 continues attempt to access the shared data. • P1 continues to wait in the loop because the value of `turn` is B .
t+2	• P0 completes updating the shared data.
t+3	• P1 continues attempt to access the shared data. • Subsequently, P1 enters into its critical section and accesses the shared data.
t+4	• P0 attempts to access the shared data again. • P0 continues to wait in the loop.
t+5	• P1 completes updating the shared data.
t+6	• P0 continues attempt to access the shared data. • Subsequently, P0 enters into its critical section and accesses the shared data.
t+7	• P0 completes updating the shared data. • At this point, the value of `turn` is C .
t+8	• ...

This mutually exclusive control for P0 and P1 is likely to involve a problem. For example, at the time *t*+8 in Table 1, a problem occurs under the situation where D .

[Mutually exclusive control - Case 2]

To solve the problem in Case 1, an alternative mutually exclusive control is proposed by introducing two global variables `flag0` and `flag1`, instead of using a single variable `turn`. The initial value of both `flag0` and `flag1` is set to 0.

Figure 2 shows the alternative mutually exclusive control for P0 and P1.

```
PROCESS P0 {
    ...
    SET flag0 ← 1
    WHILE (flag1 is 1) {
        WAIT for a predefined time unit
    }
    UPDATE shared data
    SET flag0 ← 0
    ...
}
```

```
PROCESS P1 {
    ...
    SET flag1 ← 1
    WHILE (flag0 is 1) {
        WAIT for a predefined time unit
    }
    UPDATE shared data
    SET flag1 ← 0
    ...
}
```

Figure 2 Alternative mutually exclusive control for P0 and P1

However, this mutually exclusive control for P0 and P1 is also likely to involve a problem. For example, a problem occurs under the situation where [E].

Subquestion

From the answer groups below, select the correct answer to be inserted in each blank [ ] in the description.

Answer group for A through C

a) 0   b) 1   c) P0   d) P1

Answer group for D

a) Both P0 and P1 attempt to access the shared data.
b) Both P0 and P1 have no more need to access the shared data.
c) P0 attempts to access the shared data; however, P1 has no more need to access it.
d) P1 attempts to access the shared data; however, P0 has no more need to access it.

Answer group for E

a) Both P0 and P1 update the shared data exactly once.
b) P0 (or P1) updates the shared data continuously.
c) P0 sets `flag0` to 1, and P1 sets `flag1` to 1 simultaneously.
d) The number of times the shared data is updated by P0 differs from that by P1.

Q3. Read the following description of a relational database, and then answer Subquestions 1 through 3.

An IT training center in a city is developing a relational database to manage its training courses and students.

The relational database is composed of three tables: COURSE table, BATCH table, and STUDENT table. Each course has numerous batches. Moreover, each batch of a course has a number of students.

For each table, the table structure with its sample data are shown below. Here, a solid underline indicates a primary key, and a dotted underline indicates a foreign key.

(1) COURSE table

COURSE table contains the information about the course code, course name, and description.

CourseCode	CourseName	Description
10	Database	For database engineers
20	Programming	For C and Java programmers
30	Network	For network engineers

(2) BATCH table

BATCH table contains the information about the batch code, starting date, duration, course fee, net income, expected income, and course code. A course has numerous batches.

BatchCode	StartingDate	Duration	CourseFee	NetIncome	ExpectedIncome	CourseCode
101709	2017-09-04	2	1,000	8,000	10,000	10
101710	2017-10-02	2	1,000	16,000	10,000	10
201709	2017-09-11	4	2,000	16,000	20,000	20
301709	2017-09-25	3	1,500	18,000	15,000	30

(3) STUDENT table

STUDENT table contains the information about the student ID, student name, age, address, and email address. A student is allowed to attend any number of courses. When a student makes an application for a course, he/she specifies the batch he/she wants to attend.

StudentID	StudentName	Age	Address	EmailAddress
1	Ms. Alice	20	East 11, City S	alice@example.com
2	Mr. Bobby	28	North 22, City T	bobby@example.com
3	Mr. Charles	24	South 33, City U	charles@example.com

Subquestion 1

From the answer groups below, select the correct answer to be inserted in each blank □ in the following SQL statement.

In BATCH table, the ExpectedIncome column shows the planned income at the break-even point of profit and loss, and the NetIncome column shows the actual income obtained from the batch. Therefore, if NetIncome > ExpectedIncome, the batch gains a profit, and if NetIncome < ExpectedIncome, the batch suffers a loss. For the batch that is not completed, the NetIncome column has the value 0.

The following SQL statement outputs the course code, course name, and net profit of each course in descending order of net profit.

```
SELECT T.CourseCode, C.CourseName, T.Profit AS NetProfit
FROM (SELECT B.CourseCode, [      A      ] AS Profit
      FROM BATCH B
      GROUP BY B.CourseCode) T, COURSE C
WHERE [      B      ]
ORDER BY NetProfit DESC
```

The SQL statement creates the following output from the sample data of COURSE table and BATCH table shown in the description. For example, for course code 10, the batch code 101709 suffers a loss of 2,000, and the batch code 101710 gains a profit of 6,000. Therefore, the course code 10 gains a total profit of 4,000.

CourseCode	CourseName	NetProfit
10	Database	4,000
30	Network	3,000
20	Programming	-4,000

Answer group for A

a) `SUM(B.ExpectedIncome) - SUM(B.NetIncome)`
b) `SUM(B.NetIncome) - SUM(B.ExpectedIncome)`
c) `SUM(T.ExpectedIncome) - SUM(T.NetIncome)`
d) `SUM(T.NetIncome) - SUM(T.ExpectedIncome)`

Answer group for B

a) `B.BatchCode = T.BatchCode`
b) `B.CourseCode = C.CourseCode`
c) `B.CourseCode = T.CourseCode`
d) `C.CourseCode = T.CourseCode`

Subquestion 2

From the answer groups below, select the correct answer to be inserted in each blank □ in the following SQL statement.

Training administrators wish to determine the average net income for each course.
The following SQL statement outputs the course code and average net income for each course that have an average net income greater than 15,000.

```
SELECT CourseCode, AVG(NetIncome) AS AveNetIncome
FROM BATCH
      C       CourseCode
      D       AVG(NetIncome) > 15000
```

The SQL statement creates the following output from the sample data of BATCH table shown in the description.

CourseCode	AveNetIncome
20	16,000
30	18,000

Answer group for C and D

a) `GROUP BY`   b) `HAVING`

c) `ORDER BY`   d) `WHERE`

Subquestion 3

A lecturer of a training course wishes to have a list of the students of the forthcoming batch. A database engineer intends to provide the list as shown below.

```
Course code: 10          Batch code: 101710
Course name: Database    Starting date: 2017-10-02

Student ID  Student name  Age
       1    Ms. Alice      20
       3    Mr. Charles    24
      ...     ...         ...
```

In order to provide the list, the database engineer is required to modify the structure of the tables.

From the answer group below, select the most appropriate action to modify the structure of the tables considering data redundancy and data integrity.

Answer group

a) Add the BatchCode column to STUDENT table.

b) Add the StudentID column to BATCH table.

c) Add the StudentID column to COURSE table.

d) Create a new table that consists of BatchCode and StudentID columns.

e) Create a new table that consists of BatchCode, CourseCode and StudentID columns.

Q4. Read the following description of an installation of a small-size business network, and then answer Subquestions 1 through 3.

Company V intends to install a small-size business network (hereinafter, internal network). IP addresses 10.1.1.0/24 are used for all the hosts and devices in the internal network excluding the hosts in the development department. IP addresses 10.1.2.0/25 are used for all the hosts in the development department.

There are several network segments. Variable length subnet masks are used for various network segments. They are connected to each other through the Central Router (CR), Branch 1 Router (BR1), and Branch 2 Router (BR2). To access the Internet from the internal network, NAT function is configured at CR. A packet filtering policy is also configured at CR to filter the TCP/IP traffic between the internal network and the Internet.

Figure 1 shows the configuration of the internal network, and Table 1 shows the routing table of BR1, including administrative distance (AD). An AD value is used by routers to determine the optimum route that is to be used when multiple paths to a destination exist. An AD is an integer from 0 to 255, where 0 is the most trusted, and 255 implies no traffic is to be passed via this route. A routing protocol with a lower AD value is considered as a "more preferred" path.

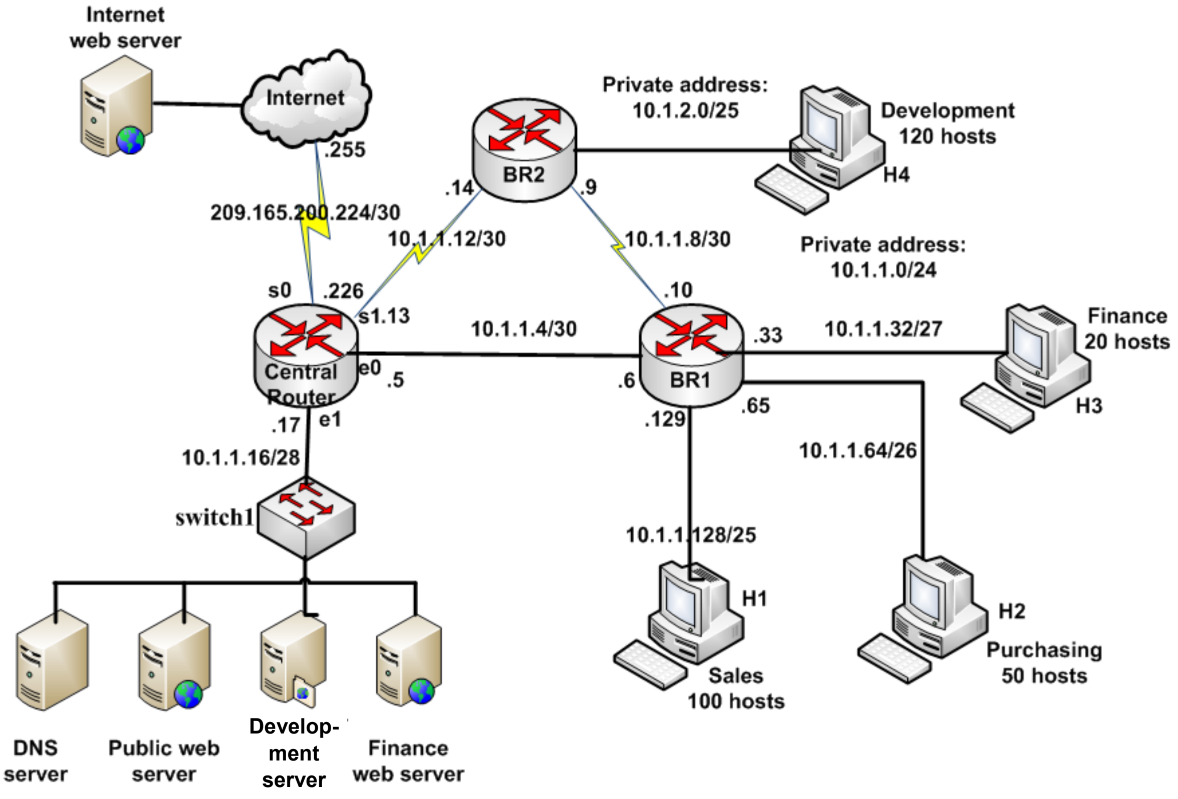

Figure 1 Configuration of the internal network

In Figure 1, the following symbols are used:

- **s*n***: Serial interface *n*, where *n* is an interface identification number
- **e*n***: Ethernet interface *n*, where *n* is an interface identification number
- **.*n***: Rightmost part of the IP address of the interface

For example, **s1 .13** implies that the IP address of serial interface **1** of CR is 10.1.1**.13** on the segment 10.1.1.12/30, and **e0 .5** implies that the IP address of Ethernet interface **0** of CR is 10.1.1**.5** on the segment 10.1.1.4/30.

Table 1 Routing table of BR1

Network	AD	Interface	Next-hop
10.1.1.4/30	0	e0	directly connected
10.1.1.32/27	0	e1	directly connected
10.1.1.64/26	0	e2	directly connected
10.1.1.128/25	0	e3	directly connected
10.1.1.8/30	0	s0	directly connected
10.1.1.16/28	1	e0	10.1.1.5
10.1.1.12/30	1	s0	10.1.1.9
10.1.2.0/25	1	s0	10.1.1.9
any/any	1	e0	10.1.1.5
any/any	50	s0	10.1.1.9

Subquestion 1

From the answer group below, select the correct answer to be inserted in the blank □ in the following description.

Concerning the hosts H2 in the purchasing department, a host creates a request to the Internet Web server to access the Internet Web services. BR1 receives this request packet via interface e2. According to the routing table, BR1 sends the packet via interface [A] for the Internet Web services.

Answer group for A

a) e0	b) e1	c) e2
d) e3	e) s0	

Subquestion 2

From the answer groups below, select the correct answer to be inserted in each blank □ in the following description.

A network administrator intends to install new servers in the server farm on the subnet 10.1.1.16/28. The maximum number of IP addresses (excluding the address of the router) that can be assigned to the servers in the server farm is $\boxed{\quad\text{B}\quad}$.

During the network test, a problem occurred. None of the hosts in the internal network could access the DNS server. The network administrator followed the trouble-shooting steps by using tools and issuing the ping commands, and determined that the hosts could communicate with the local network of BR1 (10.1.1.4/30, 10.1.1.128/25, 10.1.1.8/30) although not with the remote network address of the DNS server. By referring to the server information in Table 2 and the test results, the network administrator concluded that the cause of the problem was $\boxed{\quad\text{C}\quad}$.

Table 2 Server information

Server name	IP address/netmask	Default gateway
DNS server	10.1.1.31/28	10.1.1.17
Public web server	10.1.1.30/28	10.1.1.17
Development server	10.1.1.29/28	10.1.1.17
Finance web server	10.1.1.28/28	10.1.1.17

Answer group for B

a) 4 b) 5 c) 13
d) 14 e) 15

Answer group for C

a) TCP/IP has not been correctly installed on the hosts.
b) The default gateway on BR1 is incorrect.
c) The DNS server has been configured with the broadcast address of the subnet.
d) The network interface card installed in the hosts is not functioning.

Subquestion 3

From the answer groups below, select the correct answer to be inserted in each blank $\boxed{\qquad\quad}$ in the following description.

On the development server in the server farm, Web services are running. These Web services are accessed only from Web browsers running on the hosts H4 in the development department. The network administrator is considering security enhancement to limit the accesses to these Web services by adding the packet filtering policy on CR. The packet filtering policy is described as a set of access lists.

The format of the access list is as follows:

```
access-list   permit|deny protocol {source-address source-mask|any}
              {destination-address destination-mask|any} {eq port-number|any}
```

Example 1:

```
access-list permit tcp 10.1.1.64 0.0.0.255 10.1.2.128 0.0.0.255 any
```

This access-list permits TCP traffic from IP addresses 10.1.1.*x* to IP addresses 10.1.2.*y* for any port-number. Here, *x* is any value from 0 to 255 because the source-mask exhibits the value 0.0.0.255, indicating that "don't care" the right-most 8 bits of the source-address. Similarly, *y* is also any value from 0 to 255.

Example 2:

```
access-list deny tcp any any eq 23
```

This access-list denies TCP traffic from any IP address to any IP address for port-number 23.

Examples of port numbers are: 23 (telnet), 25 (smtp), 80 (http), 110 (pop3).

The network administrator creates the following two access lists:

```
access-list permit tcp [   D   ] 10.1.1.29 0.0.0.0 eq [   E   ]
access-list deny tcp any 10.1.1.29 0.0.0.0 [   E   ]
```

When a packet arrives at CR, the access lists are tested from the top. If the source, destination, and port-number of the packet match those in the first access list, the action (permit or deny) on the first access list is applied. Otherwise, the second access list is tested.

The first access list permits Web accesses from the hosts H4 in the development department to the Web services on the development server. The second access list denies Web accesses from any hosts to the Web services on the development server.

These access lists are to be applied at [F   ] interface (outbound direction).

Answer group for D

a) `10.1.1.0 0.0.0.0` b) `10.1.1.0 0.0.0.255` c) `10.1.2.0 0.0.0.127`
d) `10.1.2.0 0.0.0.255` e) `any`

Answer group for E

a) 23 b) 25 c) 80 d) 110

Answer group for F

a) e0 b) e1 c) s0 d) s1

Q5. Read the following description of a bus ticket reservation system, and then answer Subquestions 1 and 2.

Company W is a bus company that operates long-distance bus services. It develops a bus ticket reservation system. This system uses the Bus file, Booking file, and Seat file.
When a user wishes to book bus seats, he/she first selects the bus name, route name, departure date, and departure time on the screen. Then, the system displays detailed information of the selected bus with a seat map that shows the booking status of individual seats.

There are two types of booking processes: “buy” and “hold”.
If the user decides to “buy” the bus tickets, the system receives 100% of the fares, issues the bus tickets, and sets the relevant seat status to “sold”.
If the user wishes to “hold” the seats at the moment, the system receives 10% of the fares, and sets the relevant seat status to “held”. The user can specify “hold” by 3 days before the departure date. When the user decides to purchase the bus tickets for the held seats until 3 days before the departure date, the system receives the remaining 90% of the fares, issues the bus tickets, and alters the relevant seat status to “sold”. When the user wishes to cancel the held seats by 3 days before the departure date, the system returns the received 10% of the fares. When the user takes no action for the held seats by 3 days before the departure date, the system does not return the received 10% of the fares, and alters the relevant seat status of Booking file to “canceled” and the relevant seat status of Seat file to “available”.

Figures 1, 2 and 3 show the record format of Bus file, Booking file and Seat file respectively, with sample data. The underlined fields are primary keys.

<u>BusNo</u>	BusName	RouteName	DepartureDate	DepartureTime	SeatingCapacity	Fare
1021118	YM18	City Y → City M	2017-10-21	18:00	50	5,000
1021119	YM19	City Y → City M	2017-10-21	19:00	50	5,000
1021121	YM21	City Y → City M	2017-10-21	21:00	50	6,000
1021318	YB18	City Y → City B	2017-10-21	18:00	30	3,000

Figure 1 Record format of Bus file

<u>BookingID</u>	BusNo	UserName	PhoneNo	NumberOfSeats	TotalFare	Paid	SeatStatus
101402	1021118	Mr. A	09-1234567	2	10,000	10,000	sold
101403	1021118	Ms. B	09-2345678	3	15,000	1,500	held

Note: SeatStatus holds one of the three values: “canceled”, “held”, or “sold”.

Figure 2 Record format of Booking file

BusNo	SeatID	BookingID	SeatStatus
1021118	1A	101403	held
1021118	1B	101403	held
1021118	1C	101403	held
1021118	1D	(null)	available
1021118	2A	101402	sold
1021118	2B	101402	sold
…	…	…	…

Note: SeatStatus holds one of the three values: “available”, “held”, or “sold”.

Figure 3 Record format of Seat file

In Program 1 in Subquestion 1 and Program 2 in Subquestion 2, a `READ` statement

- `READ` *filename* `file WHERE` *condition*

reads a record that satisfies the *condition* from the *filename* file.

If the record is read successfully, the host variable `&ReadStatus` is set to “success”, and each field of the record can be referred to by specifying *filename*`.`*fieldname*, such as `Seat.BusNo`. If there are multiple records that satisfy the *condition*, the records can be read in one by one by executing the `READ` statement repeatedly.

Subquestion 1

From the answer groups below, select the correct answer to be inserted in each blank □ in Program 1.

Program 1 is a small-size module that performs a part of the cancellation process. It is called when a user requests cancellation of held seats.

[Program 1]

```
○ String type: WebInput
• DISPLAY "Enter [    A    ]:"
• GET the value entered from keyboard into WebInput
• READ Booking file WHERE Booking.[    A    ] = WebInput
▲ (&ReadStatus = "success") AND (Booking.SeatStatus = "held")
│ • READ Seat file WHERE Seat.BookingID = Booking.BookingID
│ ■ &ReadStatus = "success"     /* the next record exists */
│ │ • Seat.BookingID ← "(null)"
│ │ • [    B    ]
│ │ • WRITE the updated record into Seat file
│ │ • READ Seat file WHERE Seat.BookingID = Booking.BookingID
│ ■
│ • [    C    ]
│ • WRITE the updated record into Booking file
▼
```

Answer group for A

a)	`BookingID`	b)	`BusNo`
c)	`SeatStatus`	d)	`UserName`

Answer group for B and C

a) `Booking.SeatStatus ← "canceled"`

b) `Booking.SeatStatus ← Seat.SeatStatus`

c) `Booking.UserName ← "(null)"`

d) `Seat.SeatID ← "(null)"`

e) `Seat.SeatStatus ← "available"`

f) `Seat.SeatStatus ← Booking.SeatStatus`

Subquestion 2

From the answer group below, select the correct answer to be inserted in each blank [   ] in Program 2.

Program 2 is a small-size module that also performs a part of the cancellation process. It starts at 1:00 daily as a batch job subsequent to the file backup operation.
Here, the date of the day after tomorrow is provided by the host variable &TodayPlus2.

[Program 2]

```
• READ Bus file WHERE [  D  ]
■ &ReadStatus = "success"
│ • READ Booking file WHERE [  E  ] AND [  F  ]
│ ■ &ReadStatus = "success"
│ │ • READ Seat file WHERE Seat.BookingID = Booking.BookingID
│ │ ■ &ReadStatus = "success"
│ │ │ • Seat.BookingID ← "(null)"
│ │ │ • Seat.SeatStatus ← "available"
│ │ │ • WRITE the updated record into Seat file
│ │ │ • READ Seat file WHERE Seat.BookingID = Booking.BookingID
│ │ ■
│ │ • Booking.SeatStatus ← "canceled"
│ │ • WRITE the updated record into Booking file
│ │ • READ Booking file WHERE [  E  ] AND [  F  ]
│ ■
│ • READ Bus file WHERE [  D  ]
■
```

Answer group for D through F

a) `Booking.BookingID = Seat.BookingID`

b) `Booking.BusNo = Bus.BusNo`

c) `Booking.BusNo = Seat.BusNo`

d) `Booking.SeatStatus = "held"`

e) `Booking.SeatStatus = Seat.SeatStatus`

f) `Bus.DepartureDate ≤ &TodayPlus2`

g) `Bus.DepartureDate = &TodayPlus2`

Q6. Read the following description of programs and the programs themselves, and then answer Subquestions 1 through 3.

[Program Description]

(1) For enhancing the environment to attract higher number of tourists, it is decided to carry out waste collection along the coast. The entire coastline of City A resort is divided into `N` portions from 1 to `N`. Analysis reveals that in the `I`-th portion, there are `X[I]` tons of garbage; `I` is in the range from 1 to `N`.

(2) There is a garbage truck for use in trial operation. During a trip, the truck can collect and process a maximum of `T` tons of garbage. Because it is a trial run, the engineers are highly cautious, wishing to select a certain shoreline that includes sequential portions of coastline for convenient tracking and evaluation.

(3) The algorithm determines the number of feasible trips that can be selected for the truck to clean the coastline, considering that each trip covers sequential portions and that the total garbage in these portions is less than or equal to the truck capacity.
Program 1, named `TrashGen`, calculates and outputs the number of feasible trips. The argument specification for the program `TrashGen` is given in Table 1.

Table 1 Argument specification for the program `TrashGen`

Variable	Input/Output	Description
`N`	Input	The number of coastline portions.
`T`	Input	The maximum quantity of garbage (tons) that the collector can process. Here, `T` is a positive integer.
`X[]`	Input	An array of `N` elements where `X[I]` is the quantity of garbage (tons) at portion `I`.
`Seg`	Output	The number of feasible trips.

(4) For example, consider a sample case of waste collection shown in Figure 1.

`N`	9
`T`	10

	1	2	3	4	5	6	7	8	9
`X[]`	11	1	2	1	1	5	10	2	3

Figure 1 Sample case of waste collection

In the case shown in Figure 1, there are 19 feasible trips for the truck to clean the coastline, namely, {2}, {2,3}, {2,3,4}, {2,3,4,5}, {2,3,4,5,6}, {3}, {3,4}, {3,4,5}, {3,4,5,6}, {4}, {4,5}, {4,5,6}, {5}, {5,6}, {6}, {7}, {8}, {8, 9}, and {9}. Therefore, the program `TrashGen` returns 19.

[Program 1]

```
○ Program: TrashGen(Integer type: N,
                    Integer type: T,
                    Integer type: X[N])
  ○ Integer type: I, J, Seg, Sum

  • Seg ← 0
  • Sum ← 0
  ■ I: 1, [   A   ], 1
  │ • Sum ← 0
  │ ■ J: I, [   B   ], 1
  │ │ • Sum ← Sum + X[J]
  │ │ ▲ Sum ≤ T
  │ │ │ • [   C   ]
  │ │ ▼
  │ ■
  ■
  • Return Seg
```

Subquestion 1

From the answer groups below, select the correct answer to be inserted in each blank [   ] in the above program.

Answer group for A

a) `I < N`

b) `I ≤ N`

c) `I < N and Sum < T`

d) `I ≤ N and Sum < T`

e) `I < N and Sum ≤ T`

f) `I ≤ N and Sum ≤ T`

Answer group for B

a) `J < N and Sum ≤ T`

b) `J ≤ N and Sum ≤ T`

c) `J < N and Sum ≤ X[I]`

d) `J ≤ N and Sum ≤ X[I]`

e) `J < N or Sum ≤ T`

f) `J ≤ N or Sum ≤ T`

Answer group for C

a) `Seg ← Seg + 1`

b) `Seg ← Seg + J − I`

c) `Seg ← Seg + J − I - 1`

d) `Seg ← Seg + J − I + 1`

e) `Seg ← Seg + X[I]`

f) `Seg ← Seg + X[J]`

Subquestion 2

Program 2, named `TrashGen2`, is implemented by modifying the program `TrashGen` to output certain information.

From the answer group below, select the correct answer to be inserted in each blank [   ] in the following description. Here, assume that the correct answers are inserted in blanks [A] through [C] in Program 2.

When the program `TrashGen2` is executed for the case shown in Figure 1, the following statement will be outputted. Here, the shaded part ▨ is not shown intentionally.

```
Sum of garbage from portion ▨ to portion [ D ] is [ E ].
```

[Program 2]

```
○ Program: TrashGen2(Integer type: N,
                     Integer type: T,
                     Integer type: X[N])
  ○ Integer type: I, J, M, From, To, Sum, Seg

  • M ← 0
  • Seg ← 0
  • From ← 0
  • To ← 0
  ■ I: 1, [ A ], 1
  │ • Sum ← 0
  │ ■ J: I, [ B ], 1
  │ │ • Sum ← Sum + X[J]                ← α
  │ │ ▲ Sum ≤ T
  │ │ │ ▲ Sum > M
  │ │ │ │ • M ← Sum
  │ │ │ │ • From ← I
  │ │ │ │ • To ← J
  │ │ │ ▼
  │ │ │ • [ C ]
  │ │ ▼
  │ ■
  ■
  /* Output the statement. Here, "+" means the string concatenation operator. */
  • Print ("Sum of garbage from portion " + From +
           " to portion " + To + " is " + M + ".")
  • Return Seg
```

Answer group for D and E

a) 0 b) 1 c) 2 d) 3 e) 4

f) 6 g) 7 h) 8 i) 9 j) 10

Subquestion 3

From the answer group below, select the correct answer to be inserted in the blank $\boxed{\qquad\quad}$ in the following description.

Until the program `TrashGen2` is terminated, the statement surrounded by the dashed rectangle that is pointed by "← α" will be executed $\boxed{\quad\text{F}\quad}$ times.

Answer group for F

a) 18 b) 19 c) 24 d) 26 e) 28

f) 39 g) 40 h) 52 i) 60 j) 61

Concerning questions **Q7** and **Q8**, **select one** of the two questions.
Then, mark the ⓢ in the selection area on the answer sheet, and answer the question.
If two questions are selected, only the first question will be graded.

Q7. Read the following description of a C program and the program itself, and then answer Subquestions 1 and 2.

The body mass index (BMI) is a measurement that categorizes a person as underweight, normal, overweight, or obese based on their weight and height (See Table 1). The BMI is defined as the body weight (kg) divided by the square of the body height (m).

Table 1 The BMI category

Category	BMI	
	from	to
Underweight	-	18.5
Normal	18.5	25
Overweight	25	30
Obese	30	-

[Program Description]

(1) The program reads a series of physical data from the standard input. The input for the program is provided in the following order. The first line contains the number of persons `n` (`n` is 30 or fewer). Each of the next `n` lines contain a string and two floating point numbers, indicating the person name, his/her weight in kilograms, and height in meters, respectively. The person name does not exceed in 9 characters, and no space character is inserted. The values are separated by one or more space characters. Figure 1 shows an example of input data.

```
10
Alice 78.95 1.54
Chuck 93.06 1.78
Bob 47.20 1.64
Heidi 83.19 1.80
David 72.30 1.91
Mallory 70.58 1.53
Eve 76.53 1.67
Frank 65.17 1.72
Oscar 49.91 1.45
Grace 37.15 1.60
```

Figure 1 Example of input data

(2) The program prints `n` lines as output. Each line contains three values: the name, BMI value, and category of the person. The output is sorted by category (from underweight to obese). Within each category, the output is sorted in ascending order of the ASCII codes of the name. Figure 2 shows the output for the example input data shown in Figure 1.

```
Bob 17.55 Underweight
Grace 14.51 Underweight
David 19.82 Normal
Frank 22.03 Normal
Oscar 23.74 Normal
Chuck 29.37 Overweight
Eve 27.44 Overweight
Heidi 25.68 Overweight
Alice 33.29 Obese
Mallory 30.15 Obese
```

Figure 2 Output for the input data shown in Figure 1

(3) The following four user-defined functions are used in the program:

(i) `float calcBMI(float weight, float height)`

This function calculates the BMI value.

(ii) `enum BMIcategory calcCategory(float bmi)`

This function returns the category to `bmi`.

(iii) `void sortbyBMI(struct person p[], int from, int to)`

This function sorts between the `from`-th and `to`-th element of `p[]` in ascending order of `bmi`.

(iv) `void sortbyName(struct person p[], int from, int to)`

This function sorts between the `from`-th and `to`-th element of `p[]` in ASCII code order of `name`.

(4) The following library function is used in the program:

(i) `int strcmp(const char* s1, const char* s2)`

This function compares the string pointed by `s1` and the string pointed by `s2`. When `s1` < `s2`, it returns a negative value; when `s1` = `s2`, it returns 0; when `s1` > `s2`, it returns a positive value.

[Program]

```
#include <stdio.h>
#include <string.h>

#define NAME_LEN 10
#define MAX_NUM_PERSON 30
#define NUM_CATEGORY 4

enum BMIcategory {
   underweight, normal, overweight, obese
};

struct person {
   char name[NAME_LEN];
   float bmi;
   enum BMIcategory category;
};

float calcBMI(float, float);
enum BMIcategory calcCategory(float);
void sortbyBMI(struct person p[], int from, int to);
void sortbyName(struct person p[], int from, int to);

char* BMICategoryName[] = {"Underweight", "Normal",
                           "Overweight", "Obese"};

int main() {
   int i, n;
   struct person p[MAX_NUM_PERSON];
   int numPerson[NUM_CATEGORY];
   float weight, height;
   int from = 0;

   for (i = 0; i < NUM_CATEGORY; i++) {
      numPerson[i] = 0;
   }
   scanf("%d", &n);
   for (i = 0; i < n; i++) {
      scanf("%s%f%f",      A      , &weight, &height);
      p[i].bmi = calcBMI(      B      );
      p[i].category = calcCategory(      C      );
      numPerson[p[i].category]++;
   }
   [      D      ];
   for (i = 0; i < NUM_CATEGORY; i++) {
      [      E      ];
      from += numPerson[i];
   }

   for (i = 0; i < n; i++) {
      printf("%s %3.2f %s\n",
             p[i].name, p[i].bmi, BMICategoryName[p[i].category]);
   }
}

float calcBMI(float weight, float height) {
   return weight / (height * height);
}

enum BMIcategory calcCategory(float bmi) {
   if (bmi < 18.5) {
      return underweight;
   } else if (bmi < 25) {
      return normal;
   } else if (bmi < 30) {
      return overweight;
   } else {
      return obese;
   }
}

void sortbyBMI(struct person p[], int from, int to) {
   struct person tmp;
   int i, j;

   for (i = from; i <= to - 1; i++) {
      for (int j = i + 1; j <= to; j++) {
         if (p[i].bmi > p[j].bmi) {
            tmp = p[i];
            p[i] = p[j];
            p[j] = tmp;
         }
      }
   }
}
```

```
void sortbyName(struct person p[], int from, int to) {
   struct person tmp;
   int i, j;

   for (i = from; i <= to - 1; i++) {
      for (j = i + 1; j <= to; j++) {
         if (strcmp(p[i].name, p[j].name) > 0) {    //  ← α
            tmp = p[i];                              //  ← β
            p[i] = p[j];
            p[j] = tmp;
         }
      }
   }
}
```

Subquestion 1

From the answer groups below, select the correct answer to be inserted in each blank □ in the above program.

Answer group for A

a) `&p[i].bmi`

b) `category`

c) `name`

d) `p[i].bmi`

e) `p[i].category`

f) `p[i].name`

Answer group for B

a) `&p[i].weight, &p[i].height`

b) `&weight, &height`

c) `p[i].weight, p[i].height`

d) `weight, height`

Answer group for C

a) `bmi`

b) `numPerson[i]`

c) `p[i]`

d) `p[i].bmi`

Answer group for D

a) `sortbyBMI(p, 0, n)`

b) `sortbyBMI(p, 0, n - 1)`

c) `sortbyBMI(p, 0, numPerson[0])`

d) `sortbyName(p, 0, n)`

e) `sortbyName(p, 0, n - 1)`

f) `sortbyName(p, 0, numPerson[0])`

Answer group for E

a) `sortbyBMI(p, from, from + numPerson[i])`

b) `sortbyBMI(p, from, from + numPerson[i] - 1)`

c) `sortbyBMI(p, from, numPerson[i])`

d) `sortbyBMI(p, from, numPerson[i] - 1)`

e) `sortbyName(p, from, from + numPerson[i])`

f) `sortbyName(p, from, from + numPerson[i] - 1)`

g) `sortbyName(p, from, numPerson[i])`

h) `sortbyName(p, from, numPerson[i] - 1)`

Subquestion 2

From the answer group below, select the correct answer to be inserted in each blank $\boxed{\quad\quad}$ in the following description.

The program is executed with the input data shown in Figure 1.
Until the program is terminated, the statement marked by "← α" is executed $\boxed{\quad\text{F}\quad}$ times, and the statement marked by "← β" is executed $\boxed{\quad\text{G}\quad}$ time(s).

Answer group for F

a) 4 b) 8 c) 18

d) 45 e) 55 f) 81

Answer group for G

a) 1 b) 3 c) 5

d) 7 e) 9 f) 11

Q8. Read the following description of Java programs and the programs themselves, and then answer Subquestions 1 and 2.

[Program Description]

(1) The `SpecialNumbers` class represents a set of unique integers. Its capacity can be specified at the time of creation and cannot be altered subsequently. It maintains the actual length of the set (i.e., the number of the unique integers).
It incorporates a method referred to as `conditionalAdd` that adds the specified new element to the set.

(i) If the new element is unique in the set, the new element is added and the length of the set is increased.

(ii) The sequence of the added numbers is internally maintained.

(iii) It throws an `IllegalStateException` if the new element to be added is already present or the set has already attained its maximum capacity.

(2) The `getLength` method tells how many elements are present.

(3) The `getCapacity` method tells how many elements in total can be stored.

(4) The `nth` method returns the `n`-th value after arranging all the elements in ascending (increasing) order.

(i) If `n` = 1, it returns the lowest number.
If `n` = 2, it returns the second lowest number.
If `n` = 3, it returns the third lowest number, etc.
If `n` = `length`, it returns the highest number.

(ii) The method corrects any out-of-boundary values of `n` to be as the boundary values. That is, if `n` < 1, it returns the lowest number, and if `n` > `length`, it returns the highest number.

(5) The `median` method calculates and returns the median value of the elements.

(i) If the total number of elements is odd (e.g., five elements), the median is the middle element value after sorting. For example, if there are five elements as shown below, the median is 30, that is the middle element value of the sorted elements.

Elements (original order)	20	40	10	50	30
Elements (sorted order)	10	20	30	40	50

(ii) If the total number of elements is even (e.g., six elements), the median is the average of the two middle elements after sorting. For example, if there are six elements as shown below, the median is 35, that is (30 + 40) / 2.

Elements (original order)	20	60	10	40	30	50
Elements (sorted order)	10	20	30	40	50	60

(6) The `getNumbers` method returns a new `int` array containing only the added elements, preserving the sequence.

(7) The `toString` method returns a `String` representing the added elements, preserving the sequence.

The following output is produced when the `Tester` class is executed with input values of 20, 40, 10, 50, and 30.

```
numbers: [20, 40, 10, 50, 30]
median: 30.0
nth(1): 10
nth(2): 20
nth(3): 30
nth(4): 40
nth(5): 50
nth(-5): 10
nth(100): 50
```

[Program 1]

```
import java.util.Arrays;
import java.util.NoSuchElementException;

public class SpecialNumbers {
   private final int[] numbers;
   private int length = 0;

   public SpecialNumbers(int capacity) {
      numbers = new int[capacity];
   }

   public void conditionalAdd(int newElement) {
      for (int i = 0; i < length; i++) {
         if (numbers[i] == newElement) {
            throw new IllegalStateException(newElement
                                            + " already exists");
         }
      }
      if (length == │     A     │) {
         throw new IllegalStateException("Capacity is full");
      }
      // Successful entry of unique number
      numbers[│     B     │] = newElement;
   }
```

```
public int getCapacity() {
   return     A     ;
}

public double median() {
   if (isEmpty()) {
      throw new NoSuchElementException("no numbers");
   }
   // Create a copy with mentioned quantity of elements from beginning
   int[] temp = Arrays.copyOf(numbers, length);
   Arrays.sort(temp);     // Sorts the array in ascending order
   if (temp.length % 2 == 0) {
      return (temp[     C     ] + temp[     D     ]) / 2.0;
   } else {
      return temp[     C     ];
   }
}

public int nth(int n) {
   if (isEmpty()) {
      throw new NoSuchElementException("no numbers");
   }
   if (n > length) {
      n = length;
   } else if (n < 1) {
      n = 1;
   }
   // Create a copy with mentioned quantity of elements from beginning
   int[] temp = Arrays.copyOf(numbers, length);
   // Partial sort up to n-th element
   for (int i = 1; i <= n; ++i) {
      int minLocation = i - 1;
      for (int j = minLocation + 1; j < temp.length; ++j) {
         if (temp[j] < temp[minLocation]) {
            minLocation =     E     ;
         }
      }
      if (i - 1 != minLocation) {
         int backup = temp[i - 1];
         temp[i - 1] = temp[minLocation];
         temp[minLocation] = backup;
      }
   }
   return temp[n - 1];
}
```

```
   public int[] getNumbers() {
      return [     F     ];
   }

   public int getLength() {
      return length;
   }

   private boolean isEmpty() {
      return getLength() == 0;
   }

   public String toString() {
      return Arrays.toString(getNumbers());
   }
}
```

[Program 2]

```
public class Tester {
   private static final int[] NUMBERS = { 20, 40, 10, 50, 30 };

   public static void main(String[] args) {
      SpecialNumbers sn = new SpecialNumbers(5);
      for (int e : NUMBERS) {
         sn.conditionalAdd(e);
      }
      System.out.println("numbers: " + sn);
      System.out.println("median: " + sn.median());
      for (int i = 1; i <= sn.getLength(); ++i) {
         System.out.println("nth(" + i + "): " + sn.nth(i));
      }
      System.out.println("nth(-5): " + sn.nth(-5));
      System.out.println("nth(100): " + sn.nth(100));
   }
}
```

Subquestion 1

From the answer groups below, select the correct answer to be inserted in each blank □ in Program 1.

Answer group for A and B

a) `capacity` b) `capacity + 1` c) `capacity++`
d) `length` e) `length + 1` f) `length - 1`
g) `length++` h) `numbers.length`

Answer group for C and D

a) `(temp.length + 1) / 2` b) `temp.length / 2`
c) `temp.length / 2 + 1` d) `temp.length / 2 - 1`

Answer group for E

a) `i` b) `i + 1` c) `i - 1`
d) `j` e) `j + 1` f) `j - 1`

Answer group for F

a) `Arrays.copyOf(numbers, length)`
b) `Arrays.copyOf(numbers, numbers.length)`
c) `new int[numbers.length]`
d) `new int[numbers]`
e) `numbers`
f) `numbers.clone()`

Subquestion 2

From the answer groups below, select the correct answer to be inserted in each blank □ in the following description.

In order to support a subsequence of the numbers, the following new constructor and method are added to the `SpecialNumbers` class.

(1) The new constructor creates a `SpecialNumbers` with the specified `int` array as `numbers`.

(2) The `subNumbers` method creates a `SpecialNumbers` containing a subsequence of this `SpecialNumbers`. Arguments `start` and `end` specify the first and last characters, respectively, of the subsequence. If `start` and/or `end` are out-of-range values, the

values are adjusted following the `nth` method convention. An `IllegalArgumentException` is thrown if `start` is greater than `end` after the adjustment.

The following is the source code of the additional constructor and method.

```
private SpecialNumbers(int[] numbers) {
   this.numbers = numbers;
}

public SpecialNumbers subNumbers(int start, int end) {
   if (start < 1) {
      start = 1;
   }
   if (end > getLength()) {
      end = getLength();
   }
   if (start > end) {
      throw new IllegalArgumentException();
   }
   int[] subarray = Arrays.copyOfRange(numbers, start - 1, end);
   SpecialNumbers subNumbers = new SpecialNumbers(subarray);
   subNumbers.length = subarray.length;
   return subNumbers;
}
```

In order to test the subsequence functionality, the following lines are added to the end of the `main` method in the `Tester` class.

```
   SpecialNumbers sub = sn.subNumbers(2, 5);
   System.out.println("sub: " + sub.median());
   SpecialNumbers subsub = sub.subNumbers(2, 4);
   System.out.println("subsub: " + subsub.median());
```

The following output is produced by the additional lines listed above.

```
sub: [   G   ]
subsub: [   H   ]
```

Answer group for G and H

a) `20.0`   b) `25.0`   c) `30.0`

d) `35.0`   e) `40.0`   f) `45.0`