October 2019

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 in the pseudo-language

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

[Declaration, comment, and process]

Notation		Description
type: *var1*, …, *array1*[], …		Declares variables *var1*, … , and/or arrays *array1*[], … , by data *type* such as INT and CHAR.
FUNCTION: *function*(*type*: *arg1*, …)		Declares a *function* and its arguments *arg1*, … .
/* comment */		Describes a comment.
Process	*variable* ← *expression* ;	Assigns the value of the *expression* to the *variable*.
	function (*arg1*, …) ;	Calls the *function* by passing / receiving the arguments *arg1*, … .

	IF (*condition*) {
	process1
	}
	ELSE {
	process2
	}
		Indicates the selection process.
		If the *condition* is true, then *process1* is executed.
		If the *condition* is false, then *process2* is executed, when the optional ELSE clause is present.

	WHILE (*condition*) {
	process
	}
		Indicates the "WHILE" iteration process.
		While the *condition* is true, the *process* is executed repeatedly.

	DO { *process* } WHILE (*condition*);	Indicates the "DO-WHILE" iteration process. The *process* is executed once, and then while the *condition* is true, the *process* is executed repeatedly.

	FOR (*init* ; *condition* ; *incr*) {
	process
	}
		Indicates the "FOR" iteration process.
		While the *condition* is true, the *process* is executed repeatedly.
		At the start of the first iteration, the process *init* is executed before testing the *condition*.
		At the end of each iteration, the process *incr* is executed before testing the *condition*.

[Logical constants]

true, false

[Operators and their precedence]

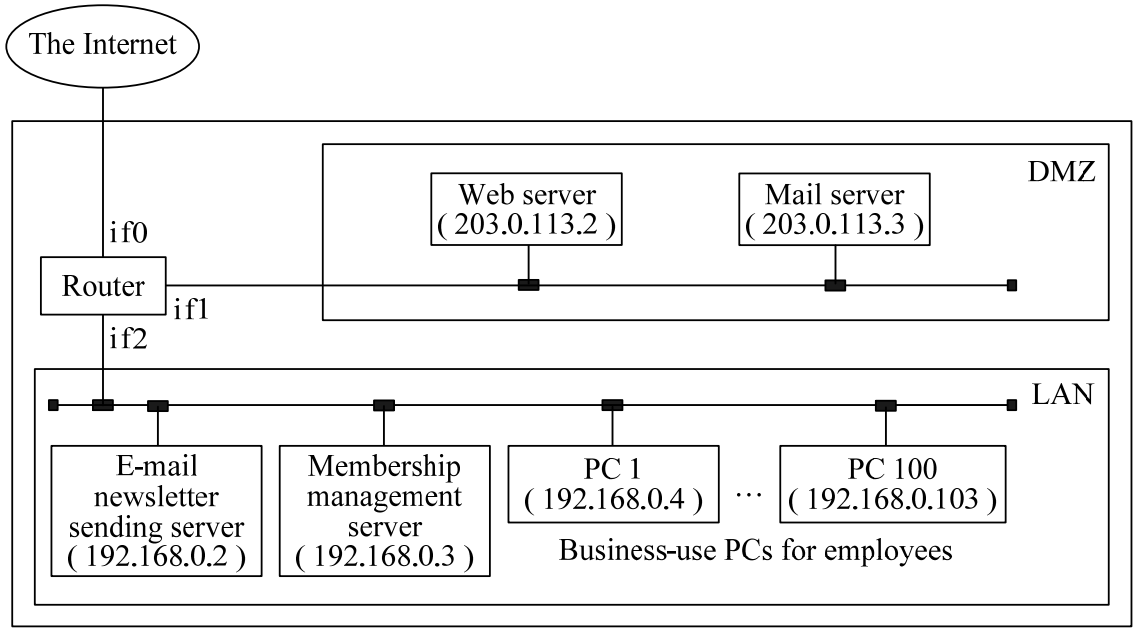
Type of operation	Unary	Arithmetic		Relational	Logical	
Operators	+, −, not	×, ÷, %	+, −	>, <, ≥, ≤, =, ≠	and	or
Precedence	High ◄					► Low

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 network security, and then answer Subquestions 1 through 4.

Company *A* constructs and operates a product introduction website on an in-house Web server. Company *A* accepts membership registrations on this website, and creates and sends e-mail newsletters to the registered members.
The network configuration of Company *A* is shown in Figure 1.



Note 1: The IP address of each device is shown in parentheses.
Note 2: if0, if1, and if2 indicate the router's network interfaces.

Figure 1 Network configuration of Company *A*

The workflow of membership registration is as follows:

(1) An applicant who wants to request membership accesses the membership application Web page via the Internet by using HTTP over TLS (hereinafter, HTTPS), and then enters an e-mail address. The Web page is managed by the Web server.

(2) For each applicant, the Web server generates a Web page specific to the applicant for entering member information, and sends an e-mail message that contains the URL of the page to the entered e-mail address.

(3) The applicant accesses a Web page dedicated to the applicant by using HTTPS, and enters member information such as personal name and occupation (other than e-mail address). The Web page is indicated by the URL in the e-mail that is sent in (2).

(4) The Web server registers the e-mail address from (1) and the member information from (3) into a database. The database runs on the membership management server (hereinafter, member information database).
(5) The Web server sends an e-mail message to the e-mail address from (1) and notifies the applicant that registration is completed.

The workflow of creating and sending an e-mail newsletter is as follows:
(1) Newsletter entry Web pages are on the e-mail newsletter sending server.
(2) The person in charge of the e-mail newsletter accesses a newsletter entry Web page with a Web browser that runs on a business-use PC by using HTTP, and enters the text of the e-mail newsletter.
(3) The e-mail newsletter sending server obtains the e-mail addresses of all members from the member information database, and sends the e-mail newsletter to the e-mail addresses.

The e-mail is sent via a mail transmission service which runs on the mail server. For purposes of this question, name resolution such as URLs or e-mail addresses needs not be considered.

Subquestion 1

The membership application Web page, first accessed by an applicant when he or she registers for membership, employs the method of requesting input of the applicant's e-mail address only, then separately requesting input of member information. From the answer group below, select the appropriate main objective for using this two-step procedure.

Answer group

a) In order to encrypt communication and prevent the leak of applicants' member information to third parties.
b) In order to prevent access by the applicant to the member information database.
c) In order to prevent registration of another person's e-mail address or an incorrect e-mail address.
d) In order to prevent the registration of incorrect member information (other than e-mail address).

Subquestion 2

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

A router is equipped with the firewall feature of a dynamic packet filter-type and only permits packets that are allowed by the router's settings to pass through.
The setting is listed in order of source, destination, and communication port, and separated by commas. The settings allow packets from the source to the communications port at the destination to pass through. The settings also allow the packets in response to the allowed packets to pass through.
Source and destination are specified by an IP address or a network interface. When an IP address is specified, packets from that IP address (when specified as the source) or to that IP address (when specified as the destination) are allowed. When a network interface is specified, packets coming in (when specified as the source) or going out (when specified as the destination) through that network interface are allowed.
Communication port is specified by a port number by which each service listens for packets. The protocols and listening port numbers used by each service running on each server in Company *A* are shown in Table 1.

Table 1 Protocols and listening port numbers, by service

Service	Protocol	Listening port number
Web	HTTP	80
	HTTPS	443
Member information database	Proprietary protocol	4194
E-mail (receiving)	POP3	110
E-mail (transmission)	SMTP	25

The current router settings are shown in Figure 2.
The setting in line 1 allows access to the Web server from the Internet by using HTTP. The setting in line 2 allows transmission of e-mail that comes in from the Internet to the mail server.

```
if0,203.0.113.2,80
if0,203.0.113.3,25
203.0.113.3,if0,25
if2,203.0.113.2,80
if2,203.0.113.2,443
if2,203.0.113.3,25
if2,203.0.113.3,110
if0, [ A ]
203.0.113.2, [ B ]
```

Figure 2 Current router settings

Answer group for A and B

a) 192.168.0.2,443
b) 192.168.0.2,4194
c) 192.168.0.3,4194
d) 203.0.113.2,443
e) 203.0.113.2,4194
f) if0,443

Subquestion 3

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

Company *A* outsources its Web server maintenance, and the outsourcing contractor is allowed to operate Company *A*'s Web server over the Internet from their specified PCs. For that reason, Company *A* runs an SSH service on listening port 22 on the Web server, and adds a new line "[C]" to the router settings.
When a packet sent from this PC arrives at the router, the packet's sender IP address is 198.51.100.2.

Answer group for C

a) 198.51.100.2,203.0.113.2,22
b) 198.51.100.2,if0,22
c) 203.0.113.2,198.51.100.2,22
d) if0,198.51.100.2,22

Subquestion 4

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

The methods by which SSH services authenticate clients include password authentication method and public key authentication method. Company *A* adopts public key authentication method.

Public key authentication method verifies signatures that are created with private keys by using the corresponding public keys, and performs authentication of clients as follows:

(1) The client sends the signature that is created with the private key, along with the public key corresponding to that private key, to the server.

(2) The server confirms that the public key in (1) is registered with the server, and uses the public key to verify the signature from (1).

(3) Successful verification means that the client holds the private key that corresponds to the public key that is registered with the server, and thus the client is authenticated.

In this way, when clients use SSH services on servers with public key authentication method, there is no need to transmit information such as passwords and [D] over the network.

Answer group for D

a) private keys

b) private keys and public keys

c) public keys

Q2. Read the following description of logical expressions, and then answer Subquestion.

Consider a system that accepts a 4-bit integer number of 0000(0) to 1111(15) from input lines I3, I2, I1 and I0, and checks whether the number meets one or more of the three types of numbers shown in Table 1. The system then outputs the results to output lines Sq, Pr and Mp according to the truth table shown in Table 2.

Table 1 Three types of numbers checked by the system

Type (Output line)	Description	Applicable numbers
The square of a number (Sq)	An integer number that is the square of a number	0000(0), 0001(1), 0100(4), 1001(9)
A prime number (Pr)	A natural number greater than 1 that cannot be formed by multiplying two smaller natural numbers	0010(2), 0011(3), 0101(5), 0111(7), 1011(11), 1101(13)
A multiple of 3 (Mp)	An integer number that is a multiple of 3	0000(0), 0011(3), 0110(6), 1001(9), 1100(12), 1111(15)

Table 2 Truth table

	Inputs				Outputs		
	I3	I2	I1	I0	Sq	Pr	Mp
0	0	0	0	0	1	0	1
1	0	0	0	1	1	0	0
2	0	0	1	0	0	1	0
3	0	0	1	1	0	1	1
4	0	1	0	0	1	0	0
5	0	1	0	1	0	1	0
6	0	1	1	0	0	0	1
7	0	1	1	1	0	1	0
8	1	0	0	0	0	0	0
9	1	0	0	1	1	0	1
10	1	0	1	0	0	0	0
11	1	0	1	1	0	1	0
12	1	1	0	0	0	0	1
13	1	1	0	1	0	1	0
14	1	1	1	0	0	0	0
15	1	1	1	1	0	0	1

In the expressions shown below, the symbols •, + and $\overline{\ }$ are used to indicate the logical operators AND, OR and NOT respectively.

First, consider a logical expression that checks if a number is a square of a number and outputs the result to the output line Sq. The expression Sq-1 shown below outputs 1 on output line Sq when the number is 0000(0), 0001(1), 0100(4), or 1001(9); otherwise it outputs 0. Each term in expression Sq-1 corresponds to a number.
The first two terms in expression Sq-1, that correspond to numbers 0000(0) and 0001(1), can be combined into a simpler term as shown in expression Sq-2.

Expression Sq-1: $\overline{I3} \bullet \overline{I2} \bullet \overline{I1} \bullet \overline{I0} + \overline{I3} \bullet \overline{I2} \bullet \overline{I1} \bullet I0 + \overline{I3} \bullet I2 \bullet \overline{I1} \bullet \overline{I0} +$ [A]

Expression Sq-2: [B] $+ \overline{I3} \bullet I2 \bullet \overline{I1} \bullet \overline{I0} +$ [A]

Next, consider a logical expression that checks if a number is a prime number and outputs the result to the output line Pr.
The truth table for Pr in Table 2 is transformed into a Karnaugh map. In this case, it is a 4 × 4 matrix as shown in Figure 1. The rows indicate inputs I3 and I2, and the columns indicate inputs I1 and I0. Each cell value represents the corresponding output value Pr that is 1 or 0 (0 is customarily omitted).

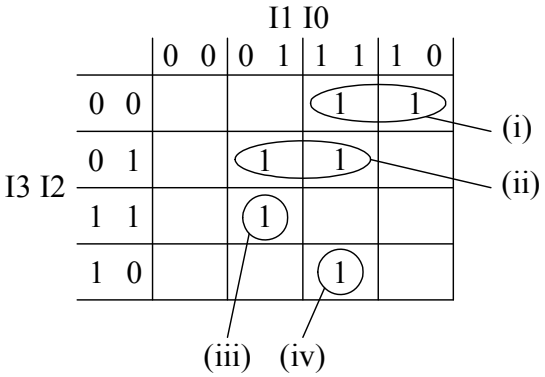


Figure 1 Karnaugh map for Pr

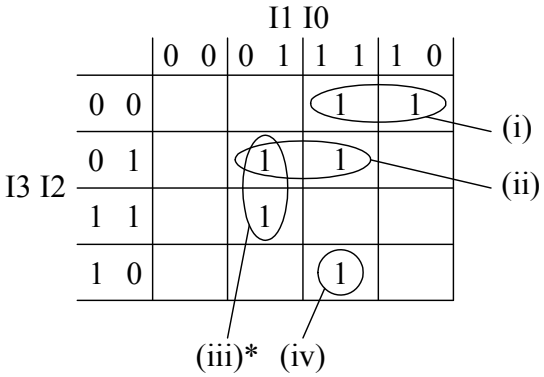


Figure 2 Karnaugh map for Pr

A Karnaugh map is used to simplify a logical expression. When a pair of adjacent 1s exists vertically or horizontally (ref. pairs marked by an ellipse in Figures 1 and 2), its output can be expressed by a simpler term, such as a transformation from expression Sq-1 into Sq-2.
From Figure 1, expression Pr-1 that outputs the result to the output line Pr is obtained.

Expression Pr-1: [C] $+ \overline{I3} \bullet I2 \bullet I0 + I3 \bullet I2 \bullet \overline{I1} \bullet I0 + I3 \bullet \overline{I2} \bullet I1 \bullet I0$

(i) (ii) (iii) (iv)

In Figure 1, the output value 1 of (iii) is evaluated as a single 1; however, this is in fact a part of vertical adjacent 1s as (iii)* shown in Figure 2. Therefore, the term corresponding to (iii) in expression Pr-1 can be expressed through a simpler term (iii)* in expression Pr-2 shown below. In this case, the output value 1 corresponds to the number 0101(5) being evaluated twice; however, this does not affect the final result.

Expression Pr-2: [C] + $\overline{I3} \cdot I2 \cdot I0$ + [D] + $I3 \cdot \overline{I2} \cdot I1 \cdot I0$

(i) (ii) (iii)* (iv)

Finally, consider a logical expression that checks if a number is a multiple of 3 and outputs the result to the output line Mp. When the Karnaugh map for Mp shown in Figure 3 is completed, there is(are) [E] pair(s) of adjacent 1s.

I3 I2 \ I1 I0	0 0	0 1	1 1	1 0
0 0	1		1	
0 1				
1 1				
1 0				

Note: shaded parts are incomplete

Figure 3 Karnaugh map for Mp

Subquestion

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

Answer group for A

a) $I3 \cdot \overline{I2} \cdot I1 \cdot \overline{I0}$ b) $I3 \cdot \overline{I2} \cdot \overline{I1} \cdot I0$ c) $\overline{I3} \cdot I2 \cdot I1 \cdot \overline{I0}$ d) $\overline{I3} \cdot I2 \cdot \overline{I1} \cdot I0$

Answer group for B through D

a) $I2 \cdot \overline{I1} \cdot I0$ b) $\overline{I2} \cdot I1 \cdot \overline{I0}$ c) $I3 \cdot I2 \cdot I1$ d) $I3 \cdot I2 \cdot \overline{I1}$

e) $I3 \cdot \overline{I2} \cdot I1$ f) $\overline{I3} \cdot I2 \cdot \overline{I1}$ g) $\overline{I3} \cdot \overline{I2} \cdot I1$ h) $\overline{I3} \cdot \overline{I2} \cdot \overline{I1}$

Answer group for E

a) no b) one c) two d) three

Q3. Read the following description of an entity relationship model, and then answer Subquestions 1 and 2.

A fruit shop C sells fruits and fruit-related products such as dried fruit and fruit juices. The shop has installed a bill management system that manages the customer information and the billing information.

Figure 1 shows the table structures of the database used in the bill management system and examples of data storage. The underlined items are the primary keys.

Product table

ProductID	Category	ProductName	UnitPrice
FMG005	Fruit	Mango 5 kg pack	10.00
FWM010	Fruit	Watermelon 10 kg pack	25.00
JPA0100	Juice	Pineapple juice 1000 ml	2.00

Customer table

CustomerID	Type	Name	Phone	Address
CH005	Company	East Hotel	223-3333	58 East town, Central city
CR052	Company	West Restaurant	224-4444	20 West town, Central city
P0082	Person	Emily Flippo	225-6789	974 South hill, Central city

Bill table

BillID	Date	CustomerID	BillAmount
08170001	2019-08-17	P0082	4.00
08170002	2019-08-17	CH005	125.00

BillDetail table

BillID	ProductID	Quantity
08170001	JPA0100	2
08170002	FMG005	5
08170002	FWM010	3

Figure 1 Table structures of the database and examples of data storage

A row of the Bill table corresponds to a bill for a customer. The details of each bill are recorded in the BillDetail table. A row of the BillDetail table is referred to as a purchase.

Subquestion 1

The shop is planning to offer a dish of assorted sliced fruit as a new product. For a promotion of this product to the customers who are likely to buy it, the shop decides to extract the name, phone, and address of customers who purchased products categorized as "Fruit" 30 times or more in which the total amount of such purchases was $300.00 or higher, during the three month period from April 1, 2019 to June 30, 2019.

From the answer group below, select the correct answer to be inserted in the blank [ ] in the following SQL statement.

```
SELECT Customer.Name, Customer.Phone, Customer.Address
   FROM Customer
   WHERE Customer.CustomerID IN
      (SELECT Bill.CustomerID
          FROM Bill, BillDetail, Product
          WHERE Bill.BillID = BillDetail.BillID
                AND Product.ProductID = BillDetail.ProductID
                [ A ])
```

Answer group for A

a)
```
AND Bill.Date >= '2019-04-01' AND Bill.Date <= '2019-06-30'
AND Product.Category = 'Fruit'
AND Product.UnitPrice * BillDetail.Quantity >= 300.00
GROUP BY Bill.CustomerID
HAVING COUNT(*) >= 30
```

b)
```
AND Bill.Date >= '2019-04-01' AND Bill.Date <= '2019-06-30'
AND Product.Category = 'Fruit'
GROUP BY Bill.CustomerID
HAVING SUM(Product.UnitPrice * BillDetail.Quantity) >= 300.00
   AND COUNT(*) >= 30
```

c)
```
AND Bill.Date >= '2019-04-01' AND Bill.Date <= '2019-06-30'
GROUP BY Bill.CustomerID
HAVING Product.Category = 'Fruit'
   AND SUM(Product.UnitPrice * BillDetail.Quantity) >= 300.00
   AND COUNT(*) >= 30
```

d)
```
GROUP BY Bill.CustomerID
HAVING Bill.Date >= '2019-04-01' AND Bill.Date <= '2019-06-30'
   AND Product.Category = 'Fruit'
   AND SUM(Bill.BillAmount) >= 300.00
   AND COUNT(*) >= 30
```

Subquestion 2

From the answer groups below, select the appropriate answer to be inserted in each blank [ ] in the following description and the SQL statement.

The shop wants to be able to change its unit prices in the Product table any number of times. Accordingly, the shop decides to add the sold unit price as an item in the BillDetail table allowing the unit prices at the time of billing to be identified. Figure 2 shows the table structure of the revised BillDetail table. The unit prices in the Product table are changed as needed prior to the operating hours of that day.

BillDetail table

BillID	ProductID	Quantity	SoldUnitPrice

Figure 2 Table structure of revised BillDetail table

By adding the sold unit price as an item in the BillDetail table, the shop can obtain information such as [B].

To track the sales status after making this revision, the shop creates the following SQL statement that outputs the total bill amount during a period of three months from July 1, 2019 to September 30, 2019 by category in the Product table for each type shown in the Customer table.

```
SELECT Type, Category, [ C ] AS TotalBillAmount
   FROM (SELECT Customer.Type, Product.Category, [ D ]
            FROM Customer, Bill, BillDetail, Product
            WHERE Customer.CustomerID = Bill.CustomerID
                  AND Bill.BillID = BillDetail.BillID
                  AND BillDetail.ProductID = Product.ProductID
                  AND Bill.Date >= '2019-07-01'
                  AND Bill.Date <= '2019-09-30'
         ) BILL3Q
   GROUP BY [ E ]
ORDER BY Type ASC, TotalBillAmount DESC
```

Answer group for B

a) the difference between the unit price at which a certain customer bought a certain product at a certain time and the unit price changed immediately thereafter
b) the transition of the sold unit price of a product that is purchased
c) the transition of the unit prices of all products
d) the unit prices on the previous day of the latest unit price change date for all products

Answer group for C through E

a) `Bill.BillAmount`
b) `BillDetail.SoldUnitPrice, BillDetail.Quantity`
c) `Product.UnitPrice, BillDetail.Quantity`
d) `SUM(BillAmount)`
e) `SUM(SoldUnitPrice * Quantity)`
f) `SUM(UnitPrice * Quantity)`
g) `Type, Category`
h) `Type, Category, TotalBillAmount`

Q4. Read the following description of a network installation for a sale company, and then answer Subquestions 1 through 3.

Company D designs a network infrastructure with variable length subnet masks (VLSM). The use of VLSM is a technique dividing an IP address space into a hierarchy of subnets of different sizes to reduce wasted IP addresses.

Company D uses a private IP address space 192.168.1.0/24 for its internal network. According to VLSM, Company D divides 192.168.1.0/24 into eight subnet masks 192.168.1.0/30, 192.168.1.4/30, 192.168.1.8/29, 192.168.1.16/28, 192.168.1.32/27, 192.168.1.64/27, 192.168.1.96/27, and 192.168.1.128/25.

Figure 1 shows the network configuration of Company D. Table 1 shows the IP address plan of the routers and PCs in the Admin Section. All departments of Company D need access to the Internet.

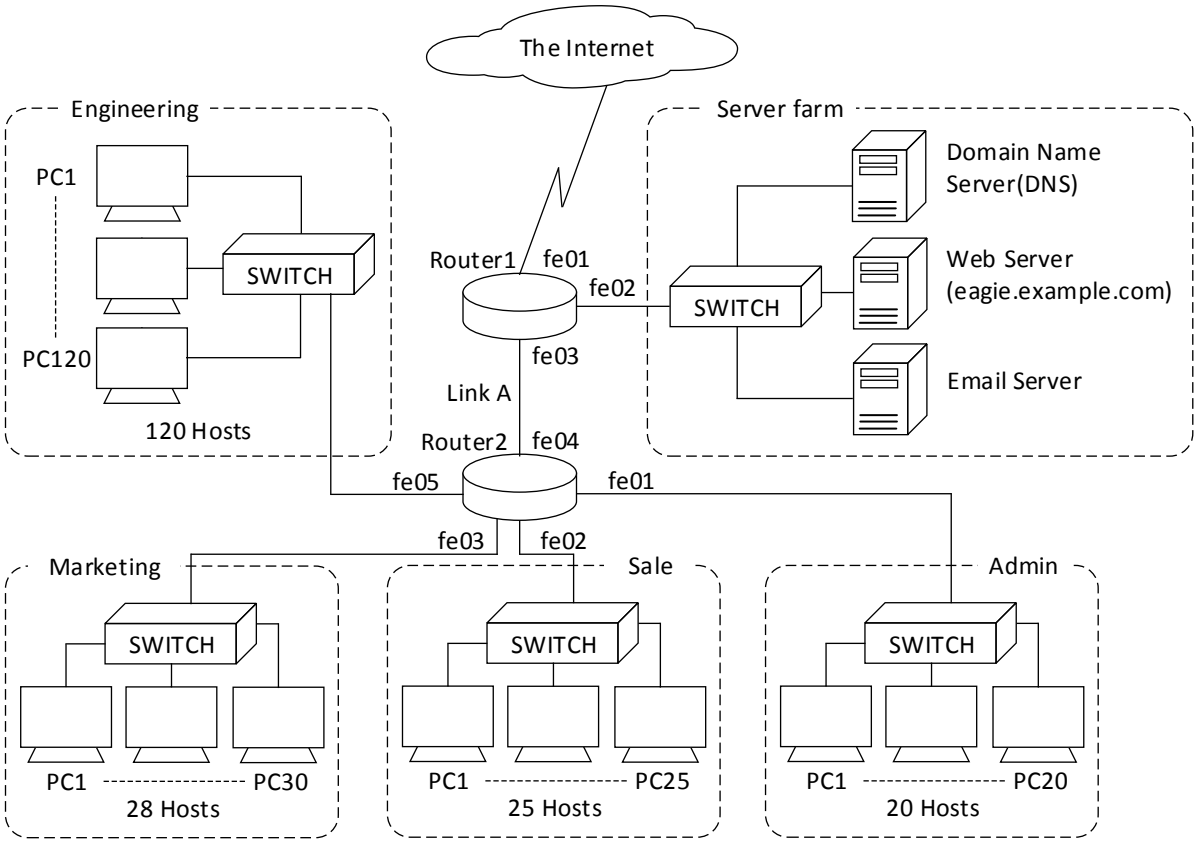


Figure 1 Network configuration of Company D

Table 1 IP address plan of routers and PCs in Admin Section (partial)

Device	Interface	IP address/mask	Gateway	Section
Router 1	fe01	202.20.120.95/30	N/A	
	fe02	192.168.1.9/29	N/A	
Router 2	fe01	192.168.1.33/27	N/A	
	fe02	192.168.1.65/27	N/A	
	fe03	192.168.1.97/27	N/A	
PC1 … PC20		192.168.1.34/27 … 192.168.1.53/27	192.168.1.33	Admin

Subquestion 1

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

Based on VLSM design, Company D assigns subnets for each section according to the required number of hosts. The lowest usable IP addresses are assigned to the hosts on the section Link A that is the connection between Router 1 and Router 2.
Table 2 shows the partial information of the IP address assignment list.

Table 2 IP address assignment list (partial)

Section	Number of hosts	Subnet
Server farm	6	192.168.1.8/29
Link A	2	A
Admin	20	192.168.1.32/27
Sale	25	192.168.1.64/27
Marketing	28	192.168.1.96/27
Engineering	120	B

Answer group for A and B

a) 192.168.1.0/30 b) 192.168.1.8/29 c) 192.168.1.16/28
d) 192.168.1.32/27 e) 192.168.1.128/25

Subquestion 2

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

All sections access the Internet, local Email, and local Web services with local DNS servers. An administrator must assign static IP addresses to the servers in Server farm. For the network 192.168.1.8/29, the fe02 of Router 1 is assigned the first usable host address, whereas the DNS server is assigned the last usable host address.
After the network configuration is completed, a network test is conducted. Testers from all sections being attempting to access the Internet, Email, and Web services. However, none of the testers can access these services. In addition, it is found that they can access shared resources by their IP addresses instead of the domain names. Thus, the administrator decides to test the connectivity by using a ping command. Table 3 shows the current network settings of the servers.

Table 3 Current network settings of the servers

Server	IP address	Subnet mask	Gateway
Web	192.168.1. 10	255.255.255.248	192.168.1. 9
Email	192.168.1. 11	255.255.255.248	192.168.1. 9
DNS	192.168.1. 15	255.255.255.240	192.168.1. 9

The test reveals that, except for the DNS server, the administrator successfully connects to all servers from the testers' PCs by using ping commands. In all PCs, the IP address of the DNS server is configured as shown in Table 3.
Based on these facts, it is found that [C] and [D]. Therefore, the problem can be resolved by entering [E] into the IP properties box of the DNS server.

Answer group for C and D

a) the Gateway of the DNS server is incorrect
b) the IP address of the DNS server conflicts with the broadcast address of the subnet
c) the IP address of the DNS server conflicts with the IP address of the Email server
d) the IP address of the DNS server conflicts with the IP address of the Web server
e) the subnet mask of the DNS server is incorrect

Answer group for E

a) IP address: 192.168.1.8, subnet mask: 255.255.255. 240
b) IP address: 192.168.1.9, subnet mask: 255.255.255. 240
c) IP address: 192.168.1.9, subnet mask: 255.255.255. 248
d) IP address: 192.168.1.14, subnet mask: 255.255.255. 248

Subquestion 3

From the answer group below, select the appropriate answer to be inserted in the blank [ ] in the following description.

For a security enhancement of Server farm section, Company D decides to configure static MAC addresses of specific servers on a switch. The switch allows a transmission of frames with the registered MAC addresses of the specific servers. If it detects a MAC address that does not match its registered address, the server cannot connect to the port of the switch. After this port security is configured, the port connected to the Email server no longer blinks. The administrator then checks the configuration of the switch and the MAC addresses of the related servers. Table 4 shows the MAC address information checked by the administrator. From Table 4, the administrator determines the cause of the problem and solves the problem by [F] after checking whether the correct Email server is connected.

Table 4 MAC address information

Port of switch	Registered MAC	Connected server	Detected MAC
fe02	00-40-0B-20-D9-43	DNS	00-40-0B-20-D9-43
fe03	00-0D-BD-48-33-3E	Web	00-0D-BD-48-33-3E
fe04	00-D0-BA-73-38-E5	Email	00-D0-BA-73-8B-E5

Answer group for F

a) changing the default gateway of the Email server to 192.168.1.9
b) changing the IP address of the Email server to 192.168.1.13
c) changing the MAC address of the Email server to 00-D0-BA-73-38-E5
d) changing the registered MAC address on port fe04 to 00-D0-BA-73-8B-E5

Q5. Read the following description of a school award system, and then answer Subquestions 1 through 3.

School E provides formal education for the primary and secondary level students (grades 1 through 12). With students normally ranging from approximately 6 to 18 years of age, School E aims for a balanced personal growth of their students with focus on the following main areas of development:

(1) Conduct
- Refers to the students' behaviors and attitudes toward other people, and their compliance or conformity to agreed-upon policies and good practices.
- Possible rating for conduct (arranged in order of highest to lowest):
  Excellent > Good > Satisfactory > Poor

(2) Extra-Curricular Activities
- Non-academic activities, usually conducted outside of the classroom, such as sports, journalism, competitions, study trips, or leadership training.
- Each time a student participates in extra-curricular activities, the corresponding points are given to that student. At the end of the school year, the total number of all points given to a student will be the extra-curricular points (ECP) of that student. Below are examples of points given for certain activities:
  - 4 points given for being a member of the swimming team in School E
  - 10 points given for winning first place in an inter-school math competition

(3) Academics
- Learning activities usually conducted inside a classroom or school laboratory, and deals with subjects such as science, math, English, or history.

[Qualification criteria]

As an added incentive for good performance, the following awards are given at the end of the school year to deserving students who meet these specific criteria for qualification:

(1) Good Conduct Award
- attain a final conduct rating of at least "Good"

(2) Best in Extra-Curricular Activities Award
- attain the highest ECP among all students within the same grade level

(3) Excellence in Academics Award
- attain a final conduct rating of at least "Good"
- the student should have no score of lower than 85 in any academic subjects
- the student should attain an average academic score of 95 or higher

(4) Distinctive Graduate Award

- attain a final conduct rating of at least "Good"
- the student should have no score of lower than 85 in any academic subjects
- should attain an overall performance score (OPS) of 95 or higher
- only applicable to graduating (currently at grade 12) students

[Steps to Determine Distinctive Graduate Award]

(1) Create a list of students who are currently at grade 12 level.
(2) Remove from the list the following students:
   (i) students who achieved a conduct rating of lower than "Good"
   (ii) students with a score of lower than 85 in any academic subjects
(3) Calculate and record the ECP for each student, and then determine ECPMin and ECPMax where ECPMin is the lowest ECP value among the students and ECPMax is the highest ECP value among the students.
(4) Calculate the ECScore for each student using the following condition and formula:

$$ECScore = 100 \times ((ECP - ECPMin) \div (ECPMax - ECPMin))$$

Here, assuming that ECPMax > ECPMin.

(5) Calculate the OPS for each student using the following formula:

$$OPS = 0.85 \times \text{average academic score} + 0.15 \times ECScore$$

(6) All students who attain an OPS of 95 or higher are qualified for the Distinctive Graduate Award.

Subquestion 1

Based on the description in [Qualification criteria], from the answer group below, select the statement that is true for any student who is qualified for the Distinctive Graduate Award.

Answer group

a) The student attains an average academic score of no lower than 95.
b) The student is also qualified for the Best in Extra-Curricular Activities Award.
c) The student is also qualified for the Excellence in Academics Award.
d) The student is also qualified for the Good Conduct Award.

Subquestion 2

From the answer groups below, select the correct answer to be inserted in each blank [ ] in Table 1.

To increase the efficiency in evaluating the students' performance, school E developed its own school award system. For a consistent implementation of the qualification criteria for each award, the decision table shown in Figure 1 was created.

Table 1 Decision table for determining awards to be given to students

Determining awards to be given to students																	
Conditions part	Conduct rating of at least "Good"	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	Y	N	N
	No score of lower than 85 in any academic subject	–	–	Y	Y	Y	Y	Y	Y	Y	A		–	N	N	–	–
	OPS of 95 or higher	N	N	Y	Y	Y	Y	N	N	–			–	–	–	–	–
	Currently in grade 12	Y	Y	Y	Y	Y	Y	–	–	N			N	–	–	–	–
	Average academic score of 95 or higher	N	N	N	N	Y	Y	Y	Y	Y			N	–	–	–	–
	Highest ECP within the same grade level	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N	Y	N
Actions part	Give Good Conduct Award	X	X	X	X	X	X	X	X	X	X	X	X	X	X	–	–
	Give Best in Extra-Curricular Activities Award	X	–	X	–	X	B				–	X	–	X	–	X	–
	Give Excellence in Academics Award	–	–	–	–	X					X	–	–	–	–	–	–
	Give Distinctive Graduate Award	–	–	X	X	X					–	–	–	–	–	–	–

Note: In the conditions part, "Y" indicates that the condition is true, "N" indicates that the condition is false, and "–" indicates that the condition has no relation to being true or false.

In the actions part, "X" indicates that the described action is executed when all conditions are met, and "–" indicates that the described action is not executed.

Answer group for A

a)

N	–
–	–
N	N
Y	N

b)

N	–
–	–
N	N
Y	Y

c)

Y	–
–	–
N	N
N	N

d)

Y	–
–	–
N	N
N	Y

e)

Y	–
–	–
N	N
Y	N

f)

Y	–
–	–
N	N
Y	Y

Answer group for B

a)

–	X	–	X
X	X	X	X
–	X	–	X

b)

–	X	–	X
X	X	X	X
X	–	–	–

c)

–	X	–	X
X	X	X	X
X	–	–	X

d)

X	–	X	–
X	X	X	X
–	X	–	X

e)

X	–	X	–
X	X	X	X
X	–	–	–

f)

X	–	X	–
X	X	X	X
X	–	–	X

Subquestion 3

Based on the description in [Steps to Determine the Distinctive Graduate Award], from the answer group below, select the statement that is true for any Grade 12 student.

Answer group

a) A student with a conduct rating of “Excellent” has a greater chance of being qualified for the Distinctive Graduate Award than a student with a conduct rating of “Good”, assuming that both students have the same average academic score and have no score of lower than 85 in any academic subjects.

b) A student with a higher ECP has a greater chance of qualifying for the Distinctive Graduate Award than a student with a lower ECP, assuming that both students have the same average academic score and both have a conduct rating of “Good”.

c) Any student whose ECP is equal to ECPMax, has a conduct rating of “Good”, and has an average academic score of 85 or higher, is qualified for the Distinctive Graduate Award.

d) Any student whose ECP is equal to ECPMin can still qualify for the Distinctive Graduate Award.

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

(See the top of this booklet for the notations used in the pseudo-language.)

This program solves the scheduling of N tasks. A task can be dependent on other task(s). The tasks and the dependency between them can be represented by a graph with N nodes and some directed edges. If task *t* depends on task *s*, there will be a directed edge from vertex *s* to vertex *t*. This program outputs orders of N tasks that can be scheduled by preserving the dependency among the tasks.

Figure 1 shows a graph with five vertices, Task 1 through Task 5, and six edges. For example, Task 2 depends on Task 3 and Task 5, and thus there are two direct edges to vertex 2 from vertex 3 and from vertex 5.

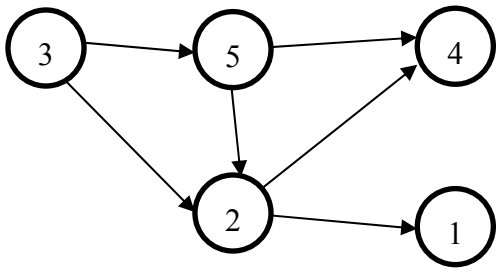


Figure 1 Example of a directed graph with five vertices

[Program Description]

The variables and functions used in the program are described in Table 1 and Table 2. The indexes of all arrays start with 1. The values of variable N and array adj[N][N] were set in advance to represent the directed graph shown in Figure 1. The value of adj[s][t] is set to 1 if there is a directed edge from vertex s to vertex t; otherwise, adj[s][t] is set to 0. Hence, adj[2][1], adj[2][4], adj[3][2], adj[3][5], adj[5][2] and adj[5][4] are 1; otherwise, they are 0.

Table 1 Variables used in the program

Variable	Description
`adj[N][N]`	Adjacency matrix, `adj[s][t]` is 1 if there is a directed edge from vertex s to vertex t, and is 0 otherwise.
`indeg[N]`	`indeg[j]` indicates the number of incoming edges to vertex `j`.
`queue_array[N]`	An array to maintain the queue. `insert_queue()` and `remove_queue()` are two functions used to enqueue and dequeue a vertex from the queue respectively.
`schedule[]`	A sequence of tasks.

Table 2 Functions used in the program

Function	Description
`Scheduling()`	To find a sequence of tasks and place it in `schedule[]` array.
`indegree(INT: vertex)`	To return the number of incoming edges to a vertex.
`insert_queue(INT: vertex)`	To insert a vertex in the queue.
`remove_queue()`	To remove a vertex from the queue.

The algorithm can be described as follows:

Step 1: Insert only those vertices in the queue for which the value of `indeg[]` is 0.

Step 2: Pop the vertices from the queue, such that every vertex `s` (`s` = 1, 2, ... , `N`) with a 0 value of `indeg[s]` is popped. Vertex `s` is placed in the array `schedule[]`.

Step 3: If vertex `s` has been popped in step 2, remove all edges originating from vertex s. This can be achieved by decreasing the value of `indeg[t]` by 1 for every `t` (`t` = 1, 2, ... , `N`) where an edge from vertex `s` to vertex `t` exists. Then, insert the vertices in the queue for which the value of `indeg[t]` is 0.

Step 4: Repeat steps 2 and 3 until the queue becomes empty.

Step 5: If the number of vertices inserted into `schedule[]` is equal to `N`, we have a schedule for `N` vertices while preserves the dependency. Otherwise, there is no schedule that preserves the dependency.

Assume that the directed edges does not constitute any closed loop(s).

[Program]

```
GLOBAL: INT: N ← 5  /* Assign 5 to N to represent Fig.1 */
GLOBAL: INT: adj[N][N] ← {{0,0,0,0,0}, /* Assign values to array */
                          {1,0,0,1,0}, /* adj to represent Fig.1 */
                          {0,1,0,0,1},
                          {0,0,0,0,0},
                          {0,1,0,1,0}}
GLOBAL: INT: front ← 1, rear ← 1
GLOBAL: INT: indeg[N]
GLOBAL: INT: queue_array[N]
GLOBAL: INT: schedule[N]

FUNCTION: insert_queue(INT: vertex) {
    queue_array[rear] ← vertex;
    rear ← rear + 1;
}

INT: FUNCTION: remove_queue() {
   INT: item
   item ← queue_array[    A    ];
    B    ;
   return item;
}

INT: FUNCTION: indegree(INT: vertex) {
   INT: i, in_deg ← 0
   FOR (i ← 1; i ≤ N; i ← i + 1) {
      IF (adj[i][vertex] = 1) {
         in_deg ← in_deg + 1;
      }
   }
   return in_deg;
}

FUNCTION: Scheduling() {
   INT:  i, j ← 1, vertex
   FOR (i ← 1; i ≤ N; i ← i + 1) {
      indeg[i] ← indegree(i);
      IF (indeg[i] = 0) {
         insert_queue(i);              /* α */
      }
   }
```

```
    WHILE (front < rear) {
       vertex ← remove_queue();
       schedule[j] ← vertex;
       j ← j + 1;
       FOR (i ← 1; i ≤ N; i ← i + 1) {  /* γ */
          IF (adj[vertex][i] = 1) {
             adj[vertex][i] ← 0;         /* β */
             indeg[i] ← indeg[i] - 1;
             IF (indeg[i] = 0) {
                insert_queue(i);
             }
          }
       }
    }

    Print("The order of tasks is:");
    FOR (i ← 1; i ≤ N; i ← i + 1) {
       Print(" ", schedule[i]);
    }
}
```

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) `1` b) `N` c) `front`
d) `front - 1` e) `front + 1`

Answer group for B

a) `front ← front - 1` b) `front ← front + 1` c) `front ← rear - 1`
d) `front ← rear` e) `front ← rear + 1`

Subquestion 2

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

Function `Scheduling` is called with the list of tasks shown in Figure 1. The line pointed out by `/* α */` is executed [C] time(s), and the line pointed out by `/* β */` is executed [D] time(s). In addition, the order of tasks printed out from the program is as follows:

```
The order of tasks is: [ E ]
```

In addition, if the line pointed out by `/* γ */` is replaced with:

```
FOR (i ← N; i ≥ 1; i ← i - 1) {
```

Then the order of tasks printed out from the program changes as follows:

```
The order of tasks is: [ F ]
```

Answer group for C and D

a) 1 b) 2 c) 3
d) 4 e) 5 f) 6
g) 7 h) 8

Answer group for E and F

a) 1 2 4 5 3 b) 1 4 2 5 3 c) 3 2 1 4 5
d) 3 5 2 1 4 e) 3 5 2 4 1 f) 3 5 4 2 1
g) 4 1 2 5 3 h) 5 4 1 2 3

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.

A computer game was designed to simulate a race among N ($2 \leq N \leq 10$) players numbered 0, 1, …, N−1. The game is played on a board of 50 squares in line, which are numbered 1, 2, …, 50.

The rules of the game are as follows:

(1) All players are at square 1 at the beginning of the game. The player who reaches square 50 first wins the game.

(2) All players are given three different "Lucky Numbers" from 1 to 6 before starting the game. One of the numbers is called the "Primary Lucky Number" and the other two numbers are called "Secondary Lucky Numbers". These numbers are used in the next rule.

(3) Each player rolls a dice twice in turn, in ascending order of their number to determine how many squares they can move.

(i) The first roll of the dice determines how many squares a player can move. For example, if a player rolls a dice and it indicates 3, the player can move 3 squares during their turn.

(ii) The second roll of the dice determines a multiplier to be multiplied to the number of squares the player can move. The multiplier is determined based on Table 1.

Table 1 Multiplier definition table

Second roll of the dice	Multiplier
Primary Lucky Number	×3
Secondary Lucky Number	×2
Other	×1

(4) If a player moves to a square where another player has already landed, the incoming player occupies the square and the previous player moves back to square 1.

Figure 1 shows an example of a player's move during the game. There are two players in this example: Player 0 has stopped at square 2 and player 1 has stopped at square 6. Suppose player 0 was given a 1 as the Primary Lucky Number and 2 and 3 as the Secondary Lucky Numbers. Player 0 now rolls the dice twice and gets a 2 followed by a 3. This allows player 0 to move to square 6 and player 1, who was occupying square 6, moves back to square 1.

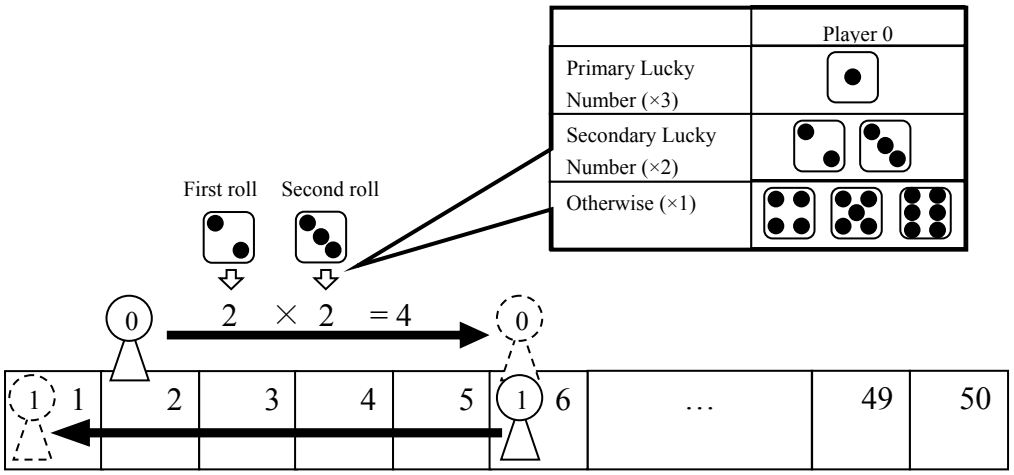


Figure 1 Example of a player moving during the game

There are different data structures and functions used to simulate the game as described below:

(1) The function `rand1` returns a random number from 1 (inclusive) to the number designated by the argument (inclusive).
(2) The function `init` places all players at square 1 and chooses their lucky numbers.
(3) The function `move` moves a player designated by the only argument and sets another player back to square 1 if needed.
(4) The function `print` prints the current status of the game to the standard output. The function outputs '`|`' for every 10th square and '`-`' otherwise, except for the following:
   (i) Squares 1 and 50, for which the function outputs '`S`' and '`G`' respectively.
   (ii) Squares at which any of the players has landed other than (i). The function outputs the player's number '`0`' to '`9`').
(5) An array `position` holds how many squares each player has moved from square 1. For example, if `position[0]` holds 1, it means that player 0 has stopped at square 2.
(6) An array `p_choice` holds the multiplier (3, 2 or 1) defined in Table 1. For example, when player *N* gets a *D* on the second roll of the dice, `p_choice[`*N*`][`*D* – 1`]` determines the multiplier for that turn.

[Program]

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int rand1(int);
void init_lucky_numbers(int);
void init(void);
int move(int);
void print(void);

#define N 4

int p_choice[N][6];
int position[N];

int rand1(int n) {
    return 1 + rand() / (((unsigned int)RAND_MAX + 1) / n);
}

void init_lucky_numbers(int playerNO) {
    int i, j, k;
    const int multipliers[] = { 1, 1, 1, 2, 2, 3 };
    for (i = 0; i < 6; i++) {
        p_choice[playerNO][i] = [ A ];
    }
    for (j = 6; j > 0; j--) {
        k = rand1( [ B ] );
        for (i = 0; i < 6; i++) {
            if (p_choice[playerNO][i] == 0) {
                k--;
                if (k == 0) {
                    p_choice[playerNO][i] = multipliers[j - 1];
                    break;
                }
            }
        }
    }
}
```

```
void init(void) {
    int i;
    /* srand resets the random number sequence */
    srand(time(NULL));
    for (i = 0; i < N; i++) {
        position[i] = 1;
        init_lucky_numbers(i);
    }
}

int move(int playerNO) {
    int base_steps = rand1(6);
    int multiplier = p_choice[playerNO][rand1(6) - 1];
    int steps = base_steps * multiplier;
    int i;
    position[playerNO] += steps;
    if (position[playerNO] >= 50) {
        position[playerNO] = 50;
        return 1;
    }
    for (i = 0; i < N; i++) {
        if (i != playerNO && position[i] == position[playerNO]) {
            position[i] = 1;
            break;
        }
    }
    return 0;
}

void print(void) {
    char status[]
             = "S--------|---------|---------|---------|---------G";
    int playerNO;
    int squareNO;

    for (playerNO = N-1; playerNO >= 0; playerNO--) {
        squareNO = position[playerNO] - 1;
        if (status[squareNO] == '-' || status[squareNO] == '|') {
            status[squareNO] = [ C ];
        }
    }
    printf("%s\n", [ D ]);
}
```

```
int main() {
    int playerNO = 0;
    init();                          <-- α
    while (!move(playerNO)) {
        print();
        playerNO = [    E    ];
    }                                    } β
    /* print the last status of the game */
    print();
    printf("Player NO. %d win!\n", playerNO);
    return 0;
}
```

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) `-1` b) `-N` c) `0`
d) `1` e) `N`

Answer Group for B

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

Answer Group for C

a) `playerNO - '0'` b) `playerNO + '0'`
c) `position[playerNO]` d) `position[playerNO] - 1`

Answer Group for D

a) `*status` b) `status` c) `status[0]`
d) `status[50]`

Answer Group for E

a) `(playerNO % N) + 1` b) `(playerNO + 1) % N`
c) `playerNO % N` d) `playerNO + 1`

Subquestion 2

The program is modified as shown below to print all players who reach square 50 with the same number of turns as winners.

(1) Insert the following line before the position designated by α.

```
unsigned int finished = 0;
```

(2) Replace the 7 lines designated by β with the following 9 lines.

```
while (!finished) {
    for (playerNO = 0; playerNO < N; playerNO++) {
        F ;
        print();
        if (position[playerNO] == 50) {
            printf("Player NO. %d win!\n", playerNO);
        }
    }
}
```

From the answer group below, select the correct answer to be inserted in the blank F in the above program.

Answer group for F

a) `finished &= move(playerNO)`
b) `finished = move(playerNO)`
c) `finished = ~move(playerNO)`
d) `finished ^= move(playerNO)`
e) `finished |= move(playerNO)`

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

[Program Description]

This program tracks the exam scores of the students. All students have a unique number (ID) identifying them. Some students may have no exam score values yet, and their score values may be updated later. The program also supports student groups where students in a single group can be sorted by a different attribute.

(1) The `Student` class represents a student having the following three attributes:
   (i) ID: The student's identification
   (ii) name: The student's name
   (iii) score: The exam score value. If no score value is available, a value of -1.0 is applied.

   The class has two constructors. One takes all attributes specified by the parameters. The other takes only the ID and name attributes specified by the parameters, and -1.0 is used as its score value indicating that this student has not yet taken the exam, or that the score value is not yet available.
   The `withScore` method returns a new `Student` instance created with the ID and name of this `Student` and the specified new score value.

(2) The `StudentGroup` class represents a group of students. The constructor creates an instance with the specified capacity that is the maximum number of students the instance can record. `Students` can be added to a `StudentGroup` by calling the `addStudents` method. The score value of a particular student can be updated by calling the `updateScore` method. The `sort` method sorts the students in the specified order.

(3) The `SortOrder` enum defines the constants to specify what `Student` attribute to use for sorting the students in a `StudentGroup`. The enum implements the `Comparator` interface so that the `sort` method of `StudentGroup` can invoke the `compare` method to collate two `Students`. The constants are as follows:
   (i) `BY_ID`: Specifies the ID attribute to be used for sorting in ascending order.
   (ii) `BY_NAME`: Specifies the name attribute to be used for sorting in ascending order. The case-insensitive comparison is performed.
   (iii) `BY_SCORE`: Specifies the score attribute to be used for sorting in descending order.

(4) The `StudentGroupTester` class tests the above classes and enum. The following output is produced by executing the `main` method of `StudentGroupTester`.

```
Sorted by SCORE:
(111, Abco, 100.0)
(444, Cdef, 70.0)
(222, Ghio, 0.0)
(333, Jokl, no score)

Sorted by NAME:
(111, Abco, 100.0)
(444, Cdef, 70.0)
(222, Ghio, 0.0)
(333, Jokl, no score)

Sorted by ID:
(111, Abco, 100.0)
(222, Ghio, 0.0)
(333, Jokl, no score)
(444, Cdef, 70.0)
```

[Program 1]

```
class Student {
   private final int id;
   private final String name;
   private final double score;

   Student(int id, String name) {
      A (id, name, -1);
   }

   Student(int id, String name, double score) {
      this.id = id;
      this.name = name;
      this.score = score;
   }

   int getId() { return id; }

   String getName() { return name; }

   double getScore() { return score; }
```

```
    Student withScore(double newScore) {
        return [ B ](id, name, newScore);
    }

    @Override
    public String toString() {
        return String.format("(%d, %s, %s)", id, name,
                             score [ C ] 0 ? "no score" : score);
    }
}
```

[Program 2]

```
class StudentGroup {
    private final Student[] students;
    private int size;

    StudentGroup(int capacity) {
        students = new Student[capacity];
    }

    void addStudents(Student... studentsToAdd) {
        for (Student student : studentsToAdd) {
            if (size == students.length) {
                throw new IllegalStateException("No more space");
            }
            students[ [ D ] ] = student;
        }
    }

    void updateScore(int id, double score) {
        for (int i = 0; i < size; i++) {
            if (students[i].getId() == id) {
                students[i] = students[i].withScore(score);
                return;
            }
        }
        throw new IllegalArgumentException("Student not found");
    }

    void sort(SortOrder order) {
        for (int i = 0; i < [ E ] - 1; i++) {
            for (int j = i + 1; j < [ E ]; j++) {
                if (order.compare(students[i], students[j]) [ F ] 0) {
                    swap(i, j);
```

```
            }
         }
      }
   }

   private void swap(int i, int j) {
      Student temp = students[i];
      students[i] = students[j];
      students[j] = temp;
   }

   @Override
   public String toString() {
      StringBuilder records = new StringBuilder();
      for (int i = 0; i < size; i++) {
         records.append(students[i]).append(System.lineSeparator());
      }
      return records.toString();
   }
}
```

[Program 3]

```
import java.util.Comparator;

enum SortOrder implements Comparator<Student> {
   BY_ID((s1, s2) -> Integer.compare(s1.getId(), s2.getId())),
   BY_NAME((s1, s2) -> s1.getName().compareToIgnoreCase(s2.getName())),
   BY_SCORE((s1, s2) -> -Double.compare(s1.getScore(), s2.getScore()));

   private final Comparator<Student> comparator;

   SortOrder(Comparator<Student> comparator) {
      this.comparator = comparator;
   }

   String getAttributeName() {
      return name().substring(3);
   }

   // Compares its two arguments for order. Returns a negative integer, zero, or a positive
   // integer as the first argument is less than, equal to, or greater than the second.
   @Override
   public int compare(Student s1, Student s2) {
      return comparator.compare(s1, s2);
   }
}
```

[Program 4]

```
public class StudentGroupTester {
   public static void main(String[] args) {
      StudentGroup studentGroup = new StudentGroup(4);
      studentGroup.addStudents(new Student(111, "Abco", 100),
                               new Student(222, "Ghio", 0),
                               new Student(333, "Jokl"),
                               new Student(444, "Cdef", 70));

      sortAndPrint(studentGroup, SortOrder.BY_SCORE);
      sortAndPrint(studentGroup, SortOrder.BY_NAME);
      sortAndPrint(studentGroup, SortOrder.BY_ID);
      /* α */
   }

   private static void sortAndPrint(StudentGroup studentGroup,
                                    SortOrder order) {
      studentGroup.sort(order);
      System.out.printf("Sorted by %s:%n%s%n",
                        order.getAttributeName(), studentGroup);
   }
}
```

Subquestion 1

From the answer groups below, select the correct answer to be inserted in each blank [ ] in Program 1 and Program 2.

Answer group for A and B

a) `new Student` b) `Student` c) `super` d) `this`

Answer group for C

a) `!=` b) `<` c) `<=` d) `==`
e) `>` f) `>=`

Answer group for D

a) `++size` b) `--size` c) `size` d) `size++`
e) `size--`

Answer group for E

a) `size` b) `size + 1` c) `size - 1`

d) `student.length` e) `student.length + 1` f) `student.length - 1`

Answer group for F

a) `!=` b) `<` c) `==` d) `>`

Subquestion 2

From the answer group below, select the correct answer to be inserted in each blank ☐ in the program output.

Replacing the comment line `/* α */` in the `main` method of `StudentGroupTest` with the following two lines will produce the output shown in the box below, in addition to the previous output.

```
studentGroup.updateScore(333, 70);
sortAndPrint(studentGroup, SortOrder.BY_SCORE);
```

```
Sorted by SCORE:
    (111, Abco, 100.0)
    (  G  , 70.0)
    (  H  , 70.0)
    (222, Ghio, 0.0)
```

Answer group for G and H

a) `333, Cdef` b) `333, Jokl`

c) `444, Cdef` d) `444, Jokl`