October 2015

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 the process 1 is executed. If it is false, then the 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 processing, 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, 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 concerning email security, and then answer Subquestions 1 through 3.

Email (electronic mail) is the most heavily used network-based distributed application. All Internet users worldwide communicate with others who are connected directly or indirectly to the Internet via email, regardless of system architecture, host operating system, vendor platform, and communication suites.

PGP (Pretty Good Privacy) is an encryption program that provides authentication for data communication by adding a digital signature to an email message. To generate a digital signature from the email body, (1) the SHA-1 (Secure Hash Algorithm-1) is used to generate a 160-bit hash code of the message, and (2) the hash code is then encrypted with RSA algorithm using the sender's private key. Then, the result is appended to the message. To save bandwidth during email transmission and the space taken by the file, the ZIP algorithm having a compression ratio of approximately 2.0 is used to compress the message before it is sent.

Figures 1 and 2 show the message authentication operations at a sender site and a receiver site, respectively.

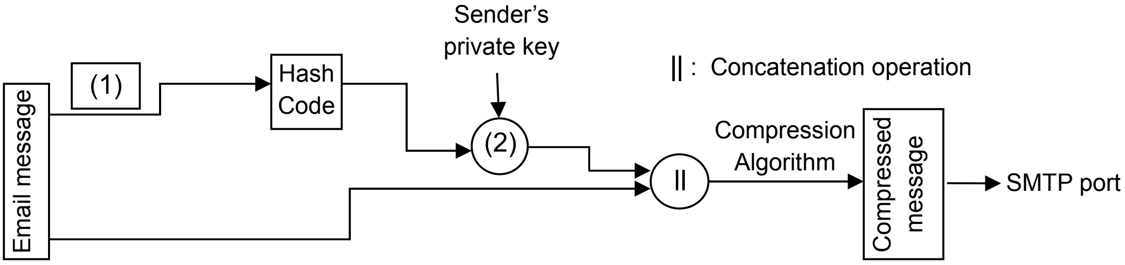


Figure 1. Message authentication operations at a sender site

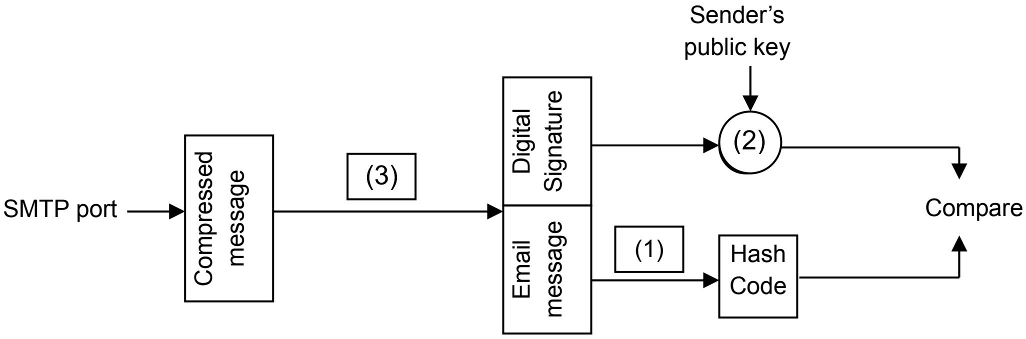


Figure 2. Message authentication operations at a receiver site

At the receiver site, the digital signature is separated from the original email message. RSA is used with the sender's public key to decrypt and recover the hash code.

The receiver generates a new hash code for the email message and compares it with the decrypted hash code. If the two codes match, the message is accepted as authentic.

The addition of a signature to an email ensures the possessor's authenticity, but does not ensure the confidentiality of the email. A symmetric encryption algorithm is used to encrypt email messages in 64-bit cipher feedback mode. In PGP, the symmetric key is used only once, and a new random 128-bit session key is generated for each message. An email is transmitted in the store and forward manner, and thus, using a handshake to ensure that both sides have the same session key is not practical. Because it is to be used only once, the key is bound to the message and transmitted with it. For protection, the session key is encrypted by using the receiver's public key.
Figure 3 shows the sequences of operations performed at a sender site and a receiver site.

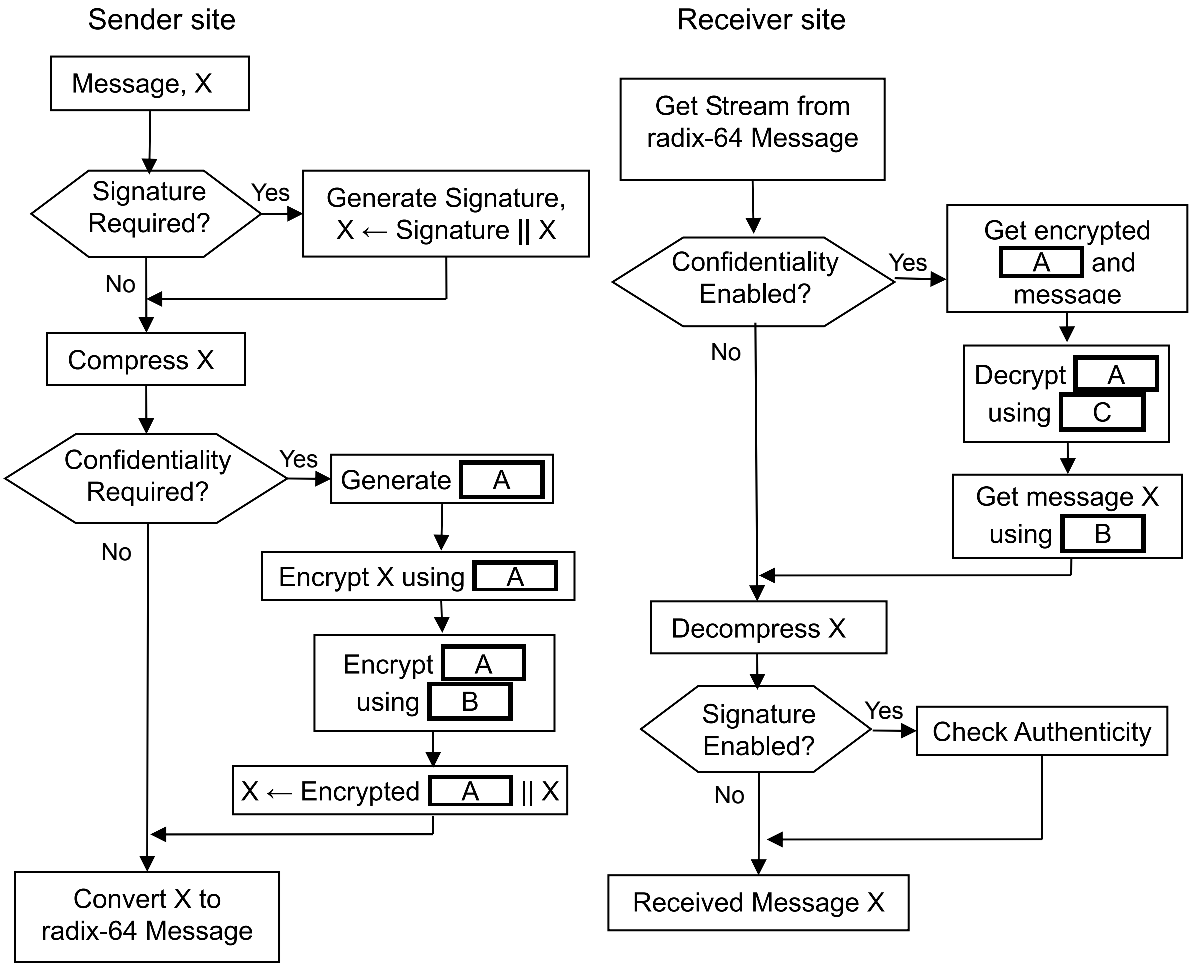


Figure 3. Operations at sender and receiver sites for ensuring both authentication and confidentiality of the email body

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 to strengthen cryptography security: because the compressed message has less redundancy than the original plaintext, crypt-analysis is more difficult.

The resulting encrypted message at the sender site consists of a stream of arbitrary 8-bit binary codes. However, many email systems permit the use only of blocks that consist of ASCII text. To accommodate this restriction, PGP provides a service that comprises converting an 8-bit binary stream to a stream of printable ASCII characters using radix-64 conversion. Each group of three 8-bit binary codes is mapped into four ASCII characters expanding the length of the message by 33%. Fortunately, the session key and signature portions of the message are relatively compact.

Subquestion 1

From the answer group below, select the correct combination of the answers for (1), (2), and (3) in Figures 1 and 2.

Answer group

	(1)	(2)	(3)
a)	Compression algorithm	RSA algorithm	ZIP algorithm
b)	RSA algorithm	Compression algorithm	SHA-1
c)	RSA algorithm	ZIP algorithm	Compression algorithm
d)	SHA-1	RSA algorithm	Compression algorithm
e)	SHA-1	RSA algorithm	Decompression algorithm
f)	ZIP algorithm	SHA-1	Decompression algorithm

Subquestion 2

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

Answer group

a) 64-bit cipher key
b) Compression key
c) Hash key
d) Receiver's private key
e) Receiver's public key
f) Sender's private key
g) Sender's public key
h) Session key

Subquestion 3

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

An email with a message size of 9 kB is transmitted enabling both authenticity and confidentiality. The ZIP algorithm is used to compress the message of the email. The size of the digital signature and encrypted session key is small and can be ignored. In this case, the size of the ASCII stream to be transmitted will be $\boxed{\text{D}}$ (in nearest kB).

Answer group

a) 4.5 kB
b) 6 kB
c) 9 kB
d) 12 kB

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

Modern multiprogramming-based operating systems allow more than one process to be loaded into the memory at a time. The loaded processes share the CPU among them through time multiplexing. In general, an operating system chooses a process for the execution based on a scheduling algorithm. The scheduling is in fact the activity of the process manager that handles the removal of the running process from the CPU and the selection of another process on the basis of a certain strategy. Some of the most popular scheduling algorithms are:

- First Come First Serve (FCFS) Scheduling
- Shortest-Job-First (SJF) Scheduling
- Round Robin(RR) Scheduling

Table 1 shows the descriptions of the FCFS, SJF, and RR scheduling algorithms.

Table 1. Descriptions of FCFS, SJF, and RR algorithms

Algorithm	Description
FCFS	Processes are executed on a first come, first serve basis. The process manager selects the process to be executed based on arrival time.
SJF	The process manager selects the process that has the shortest execution time among the waiting processes. In SJF, the scheduling is initiated when all the processes are ready for execution.
RR	The process manager provides a fixed time period, known as a quantum, for each process. When a process has been executed for the given quantum, the process is halted and the next process in the queue is chosen for execution for the given quantum. The performance of RR varies with the quantum and the number of processes in the queue.

Several criteria are applied to evaluate the performance of scheduling algorithms, such as waiting time, response time, and turnaround time. Table 2 shows the definitions of the waiting time, response time, and turn-around time.

Table 2. Definitions of waiting time, response time, and turn-around time

	Definition
Waiting time	The time the process has to wait before it receives a time slice for its execution.
Response time	The amount of time between the submission of a request and the first response (i.e., the time the process takes to start responding).
Turn-around time	The total of the waiting time and the processing time.

Subquestion

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

There are four processes, named P1, P2, P3, and P4. Table 3 shows the arrival time and processing time of each process. The processing time is the execution time required to complete the process. The arrival times in Table 3 are applicable to the FCFS and RR algorithms. The arrival times in Table 3 are not applicable to the SJF algorithm, because the scheduling starts when all the processes are ready for execution. Assume that processes P1 through P4 make their first response immediately after they start.

Table 3. Arrival time and processing time of each process

Process	Arrival time	Processing time
P1	0	5
P2	1	3
P3	2	8
P4	3	6

Figure 1 shows the execution sequence of the processes and Table 4 shows the execution results of the processes, in the case of the FCFS algorithm.

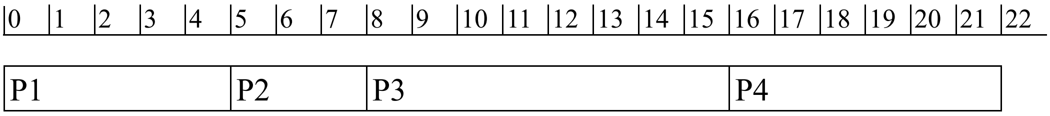


Figure 1. Execution sequence of the processes in the FCFS algorithm

Table 4. Execution results of the processes in the FCFS algorithm

Process	Waiting time	Response time	Turn-around time
P1	0 =(0−0)	0 =(0−0)	5 =(0+5)
P2	4 =(5−1)	4 =(5−1)	7 =(4+3)
P3	A		
P4	13 =(16−3)	13 =(16−3)	19 =(13+6)

Figure 2 shows the execution sequence of the processes and Table 5 shows the execution results of the processes, in the case of the SJF algorithm.

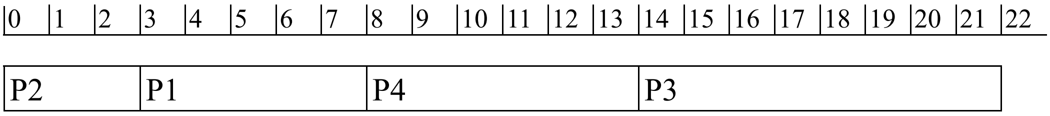


Figure 2. Execution sequence of the processes in SJF algorithm

Table 5. Execution results of the processes in SJF algorithm

Process	Waiting time	Response time	Turn-around time
P1	3	3	8
P2	0	0	3
P3	14	14	22
P4	B		

Figure 3 shows the execution sequence of the processes and Table 6 shows the execution results of the processes, in the case of the RR algorithm.
Here, the quantum used in the RR algorithm is 3.

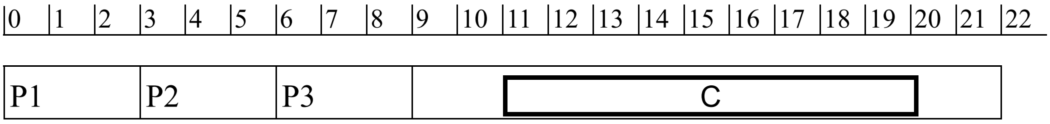


Figure 3. Execution sequence of the processes in RR algorithm

Table 6. Execution results of the processes in RR algorithm

Process	Waiting time	Response time	Turn-around time
P1	D		
P2	2 =(3-1)	2 =(3-1)	5 = (2+3)
P3	12 =(6-2)+(14-9)+(20-17)	4 =(6-2)	20 =(12+8)
P4	11 =(9-3)+(17-12)	6 =(9-3)	17 =(11+6)

Answer group for A (Note: detailed expressions are not shown)

a)	2	2	10
b)	3	3	16
c)	6	6	14
d)	8	8	18

Answer group for B

a)	5	5	11
b)	8	8	14
c)	13	13	19
d)	16	16	22

Answer group for C (Note: select the correct sequence of the processes)

a)	P1	P2	P3	P4	P1
b)	P2	P1	P4	P3	P1
c)	P4	P1	P3	P4	P3
d)	P4	P3	P1	P3	P4

Answer group for D (Note: detailed expressions are not shown)

a)	3	14	22
b)	9	0	14
c)	12	0	14
d)	12	0	17

Q3. Read the following description concerning a gymnastic center’s database, and then answer Subquestions 1 through 3.

Given the busy life of people nowadays, exercise is an essential activity for maintaining and improving health. Training at a gymnastic center does not consist of simply doing random exercises, but is planned based on a clear training strategy and effective training methods.

A gymnastic center introduced indexes to measure the progress of the training, but the data thus far have not been utilized effectively. Therefore, the gymnastic center decides to develop a gymnastic index management system to provide better services for its customers. Figure 1 shows an E-R diagram of the gymnastic index management system (incomplete), and Table 1 shows the description of each entity in Figure 1.

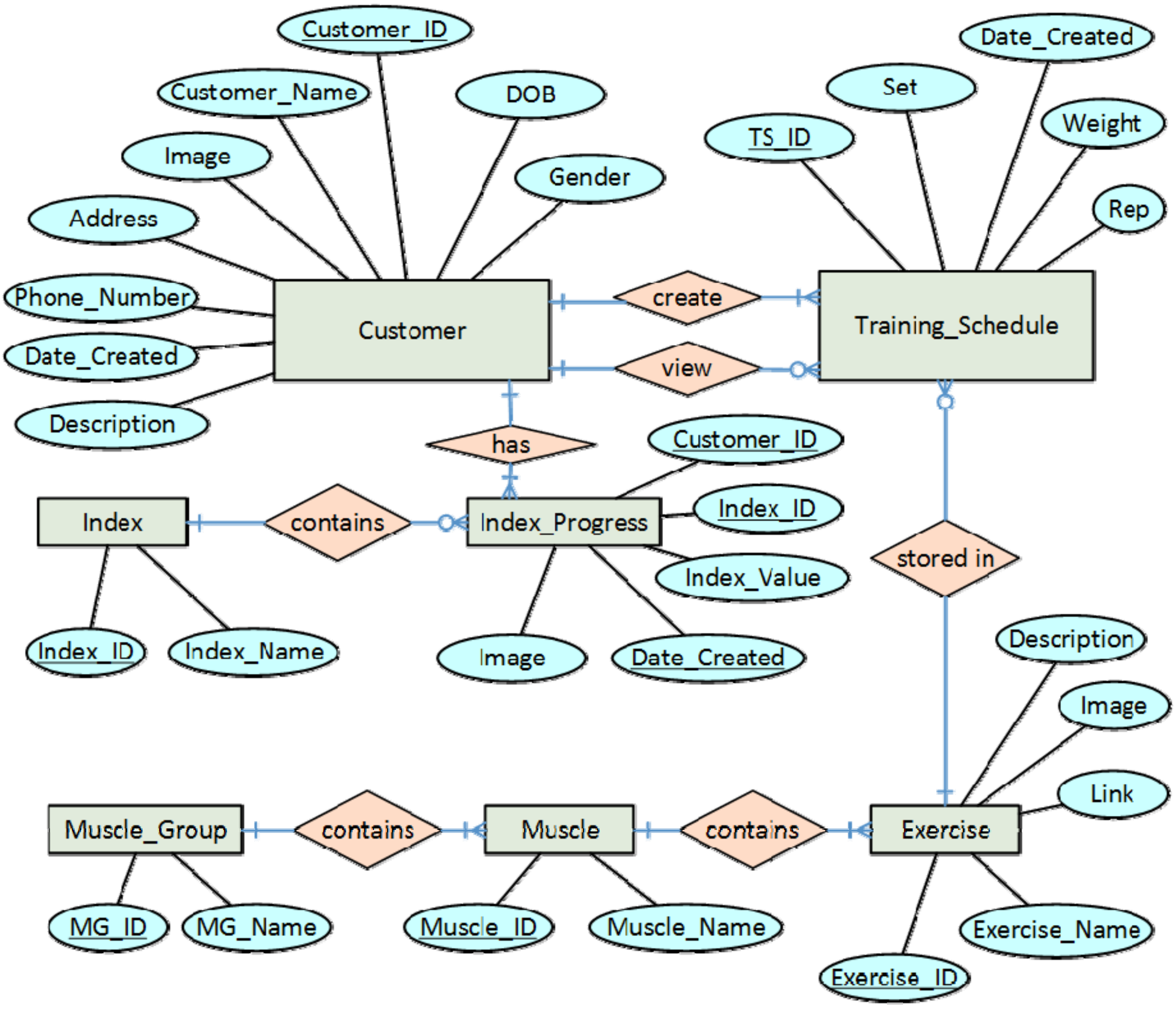


Figure 1. E-R diagram of the gymnastic index management system (incomplete)

Table 1. Description of each entity in Figure 1

Entity name	Description
Customer	Information about each registered customer.
Index	Name of each index.
Index_Progress	Index value and image of a customer at a difference time. A customer can monitor the changes in his/her index values and images.
Muscle_Group	Major muscle group in the human body. "Muscle Group" is abbreviated to "MG".
Muscle	Name of each muscle that belongs to the major muscle group.
Exercise	Information about each exercise that is associated with the muscle.
Training_Schedule	Training schedule of exercises for a customer. "Training Schedule" is abbreviated to "TS".

Subquestion 1

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

Figure 1 is incomplete because it has four missing attributes which are primary keys.

In order to complete the E-R diagram, Figure 1 should be revised as follows:

(1) Add the attribute $\boxed{\text{A}}$ to the entity Exercise.

(2) Add the attribute $\boxed{\text{B}}$ to the entity Muscle.

(3) Add the attributes Customer_ID and Exercise_ID to the entity $\boxed{\text{C}}$.

Answer group for A and B

a) Customer_ID	b) Exercise_ID
c) Index_ID	d) MG_ID
e) Muscle_ID	f) TS_ID

Answer group for C

a) Index	b) Index_Progress
c) Muscle_Group	d) Training_Schedule

Subquestion 2

From the answer groups below, select the correct answer to be inserted in each blank $\boxed{}$ in Figure 2.

Figure 2 shows an SQL statement that lists all the fitness index progress of a specific customer during the specified period from the beginning date to the end date.
Here, `:FromDate` and `:ToDate` are host variables that contain the beginning date and the end date, respectively, and `:CustID` is a host variable that contains the ID of the specific customer.

```
SELECT Customer.Customer_ID, Customer_Name, Index_Name, Index_Value
FROM   Customer, Index, Index_Progress
WHERE  Customer.Customer_ID = :CustID
       AND Customer.Customer_ID = Index_Progress.Customer_ID
       [           D          ]
[          E         ]
```

Figure 2. SQL statement

Figure 3 shows an example of the output list. The list is arranged in ascending order of the date created.

```
Customer ID    Customer Name  Index Name     Index Value
C00001         Peter          Pectorals      7
C00001         Peter          Deltoids       5
C00001         Peter          Obliques       12
C00001         Peter          Deltoids       8
```

Figure 3. Example of the output list

Answer group for D

a) `AND Customer.Date_Created = Index_Progress.Date_Created`
`AND Index_Progress.Date_Created BETWEEN :FromDate AND :ToDate`

b) `AND Customer.Date_Created = Index_Progress.Date_Created`
`AND Index_Progress.Date_Created IN (:FromDate, :ToDate)`

c) `AND Index.Index_ID = Index_Progress.Index_ID`
`AND Index_Progress.Date_Created BETWEEN :FromDate AND :ToDate`

d) `AND Index.Index_ID = Index_Progress.Index_ID`
`AND Index_Progress.Date_Created IN (:FromDate, :ToDate)`

Answer group for E

a) `GROUP BY Index_Progress.Date_Created`

b) `GROUP BY Index_Progress.Index_ID, Index_Progress.Date_Created`

c) `ORDER BY Index_Progress.Date_Created`

d) ORDER BY Index_Progress.Index_ID, Index_Progress.Date_Created DESC

Q4. Read the following description concerning Analog/Digital conversion, and then answer Subquestion.

Pulse Code Modulation (PCM) is a method of converting an analog signal into digital data. The PCM encoder has three processes: (1) Sampling → (2) Quantizing → (3) Encoding.
An Analog/Digital conversion of *n* bits is described below, using the voltage of a direct current as an example:

(1) Sampling

In sampling, a voltage that is a continuous-time analog signal is measured at a fixed time interval. In Figure 1, the time axis is divided into t0, t1, … at equally-spaced time intervals of *d*, and the voltage at a given time is represented as v(t0), v(t1), … .

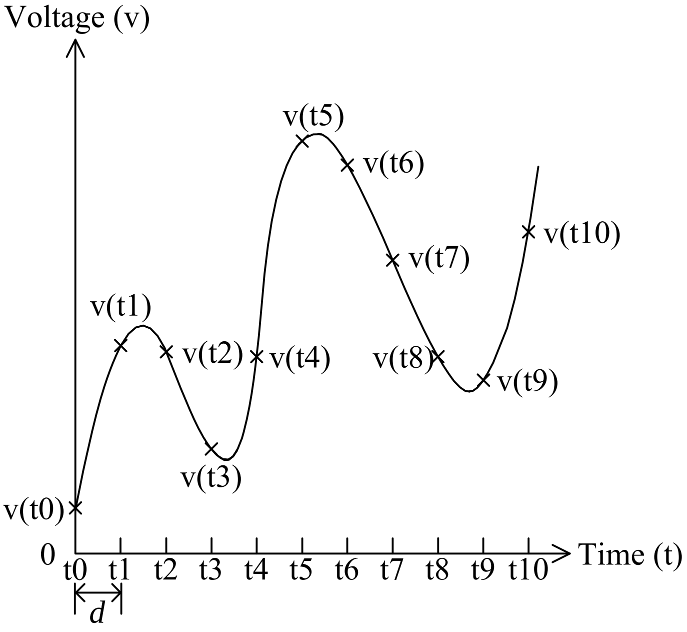


Figure 1. Example of sampling

(2) Quantization

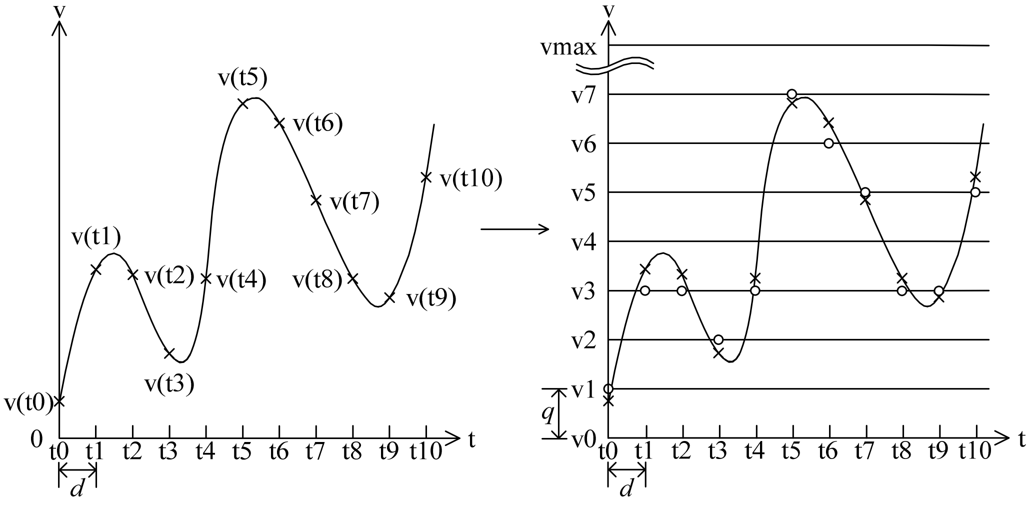
In quantization, the voltages v(t0), v(t1), … obtained by the sampling in (1) are approximated with integral multiple values of step size *q*.

In order to perform quantization, first, the FSR, which is the maximum width of the voltage to be measured, is decided. Next, in order to perform quantization with *n* bits, step size *q* is determined by equally dividing the FSR into (2^n – 1) parts. In this case, step size *q* is FSR ÷ (2^n – 1).

In quantization, when v0 = 0, v1 = q, …, vmax = $(2^n - 1) \times q$ represent the voltage values used as approximate values in ascending order, with regard to the voltage v(tm) of the analog signal measured in sampling at the time tm, a non-negative integer N that satisfies the following condition is found in quantization, and the voltage $N \times q$ is taken as the measured voltage v(tm). This is called n bit quantization:

$$N \times q - q \div 2 \ \leq \ \text{v(tm)} \ < \ N \times q + q \div 2$$

As shown on the right in Figure 2, after the voltage axis is divided into voltages v0, v1, …, vmax with step size q, for each of v(t0), v(t1), …, the closest voltage from v0 to vmax is taken as the measured value. For example, the measured value of v(t3) is v2.



Note: "○" is an approximated value of v(tm) in the integral multiples of q

Figure 2. Example of quantization

(3) Encoding

In encoding, the voltages v0, v1, …, vmax used in quantization in (2) are assigned binary codes. Each measured value is represented using these binary codes.

Subquestion

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

The encoding of the voltages v(t0), v(t1), …, v(t10) as shown on the left in Figure 2 is considered. Table 1 shows a case where the voltages v0, v1, …, v7 shown on the right in Figure 2 are assigned binary codes 000, 001, …, 111 in sequence.

Table 1. Voltages and corresponding codes

Voltage(v)	Code
v0	000
v1	001
v2	010
…	…
v7	111

Table 2 shows the results when the values measured for v(t0), v(t1), …, v(t10) shown on the left in Figure 2 are encoded in accordance with Table 1.

Table 2. Encoding of values measured at respective times

Time	t0	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10
Code	001	011	011	010	▨	▨	$\boxed{\text{A}}$	▨	▨	▨	$\boxed{\text{B}}$

Note: The shaded parts are not shown.

When the voltage range of an analog signal is between 0 and 9V, if the FSR is set to 9V and quantization with 4 bits is performed, *q* is $\boxed{\text{C}}$ V. The measured value of the voltage 7.49…V of the analog signal is $\boxed{\text{D}}$ V, and when the measured value is encoded with the binary codes 0000, 0001, …, and 1111 in sequence as in Table 1, the binary code is $\boxed{\text{E}}$.

Answer group for A and B

a) 011　　b) 100　　c) 101

d) 110　　e) 111

Answer group for C and D

a) 0.5625　　b) 0.6　　c) 1.2　　d) 2.25

e) 5.5　　f) 7.0　　g) 7.2　　h) 7.5

i) 7.8　　j) 8.0

Answer group for E

a) 1010　　b) 1011　　c) 1100

d) 1101　　e) 1110

Q5. Read the following description concerning a program design, and then answer Subquestions 1 and 2.

Company *X* is a software development company. It accepts a project for developing an Online Job Matching System (OJMS). OJMS is a social Web site that offers services to three types of person: users, applicants and employers.

Users can search jobs, but can not apply for jobs. Applicants can search and apply for jobs, and update their profiles. Employers can post jobs.

If a user wants to apply for a job, he/she must be registered in this system. After registration, he/she becomes an applicant. When an applicant wants to search and apply for jobs, or update the profile, he/she must login to this system.

Figure 1 shows the use case diagram of OJMS, focused on the user/applicant portion.

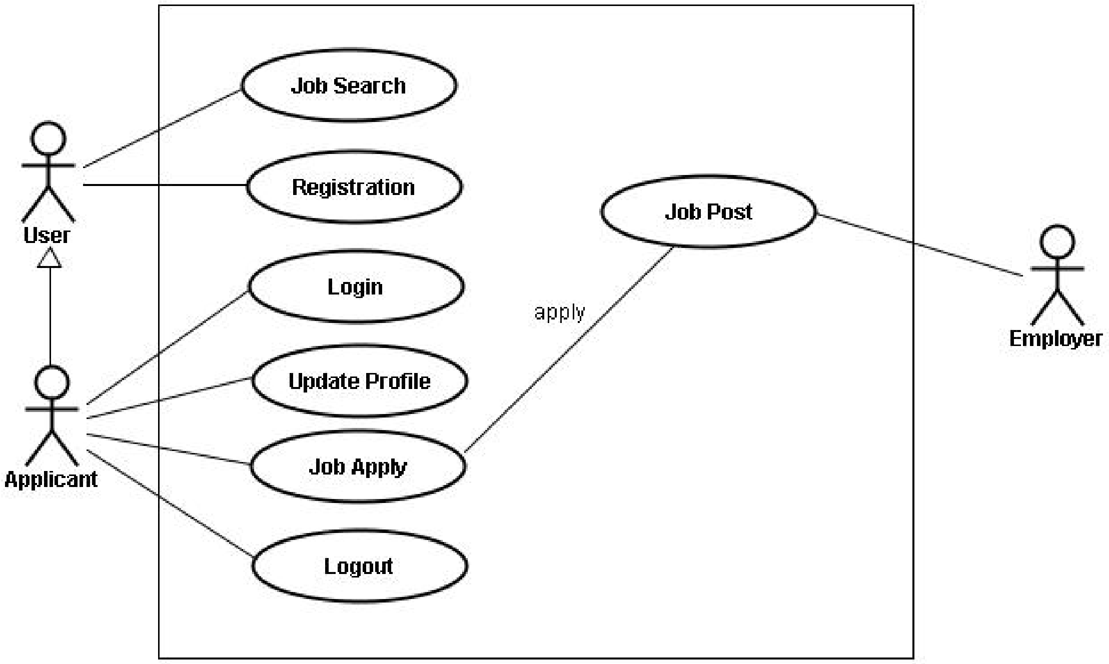


Figure 1. Use case diagram of OJMS

The following list shows the description of each use case shown in Figure 1.

Use case — Description

Job Search: A user or an applicant searches jobs and views the details of the jobs. An applicant, if he/she wishes, can go directly to the Job Apply page.

Registration: A user registers himself/herself on this system by entering information, such as user name, password, email, and phone number. After the user's successful completion of the registration, the system will send an email to the user, and he/she becomes an applicant.

Login: An applicant logs in to the system. When he/she has logged in, the menu page is displayed.

Update Profile: An applicant changes his/her registered information.

Job Apply: An applicant applies for a selected job. When the applicant has completed the job application operation, the system sends an email to the applicant and go to the Logout page to exit. When the applicant cancels the job application operation, go to the Login page to display menu.

Job Post: An employer posts jobs.

Logout: An applicant logs out from the system.

Figure2 shows the high-level OJMS navigation map for users and applicants.

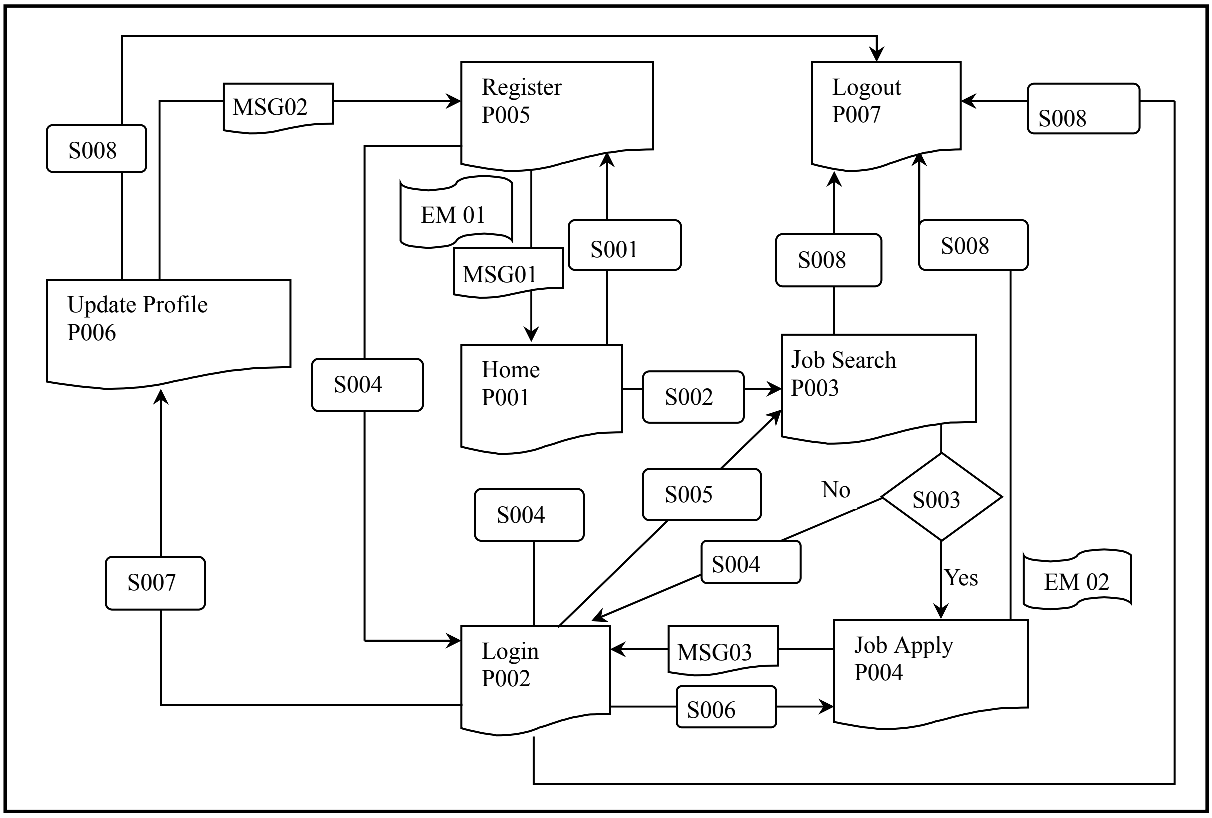


Figure 2. High-level OJMS navigation map

Table 1 shows the logical pages, Table 2 shows the screen transactions, Table 3 shows the display messages, and Table 4 shows the email messages.

Table 1. Logical pages

P001	Home Page
P002	Login Page
P003	Job Search Page
P004	Job Apply Page
P005	Register Page
P006	Update Profile Page
P007	Logout Page

Table 2. Screen transactions

S001	Create New Account
S002	Job Search (actor: A)
S003	Is actor B
S004	Login
S005	Job Search (actor: C)
S006	Job Apply (actor: C)
S007	Update Profile
S008	Logout

Table 3. Display messages

MSG01	"Registration completed"
MSG02	"Profile successfully updated"
MSG03	" D "

Table 4. Email messages

EM01	"Registration completed"
EM02	" E "

Subquestion 1

From the answer groups below, select the correct answer to be inserted in each blank □ in Tables 2, 3 and 4.

Here, the same answer can be selected twice or more if needed.

Answer group for A through C

a)	applicant	b)	either applicant or user
c)	employer	d)	user

Answer group for D and E

a)	Job application canceled	b)	Job application completed
c)	Job application is for applicants only	d)	Login is not yet complete

Subquestion 2

From the answer group below, select the incorrect sequence of page transition.

a) Home → Login → Job Apply → Logout

b) Home → Login → Update Profile

c) Home → Register → Home → Login

d) Home → Update Profile → Register → Login

Q6. Read the following description of a program and the program itself, and then answer Subquestion.

A program obtains all combinations when selecting *K* elements out of *N* elements. For example, when selecting 3 elements out of 5 elements, the program obtains all 10 combinations.
In the program, an array `S` containing `N` elements (element number: 1 to `N`) is prepared. In order to express a specific combination, the program sets 1s for the selected `K` elements in array `S` and 0s for the remaining elements. For example, when the three elements 2, 4, and 5 are selected from the five elements 1 to 5 as shown in Figure 1 (1), the program expresses the combination as shown in Figure 1 (2).

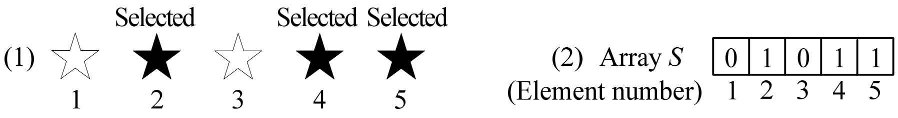


Figure 1. Example of selecting 3 elements out of 5 elements, and its expression in the program

[Program Description]

The program comprises the main program `Main`, as well as the functions `Init` and `Next` for obtaining the combinations.

```
○Main program: Main
```

Description: The program obtains all 10 combinations for the case `N` = 5 and `K` = 3 (selecting 3 out of 5 elements). The result is set in array `S` sequentially.

```
○Integer type function: Init(Integer type: S[],
                             Integer type: N, Integer type: K)
```

Arguments: `S[]` is an output argument. `N` and `K` are input arguments.

Description: When 1 ≤ `K` ≤ `N`, the function sets the first `K` elements of array `S` to 1, and sets the remaining `N` − `K` elements to 0. The function then returns 0 as the return value. Otherwise, the function returns −1 as the return value without setting any values in array `S`.

```
○Integer type function: Next(Integer type: S[], Integer type: N)
```

Arguments: `S[]` is an input / output argument. `N` is an input argument.

Description: The combination obtained most recently is already set in the first `N` elements of the received array `S`. By performing a predetermined operation for the received combination, the function obtains a new combination and sets it in array `S`, and then returns 0 as the return value. However, if the received combination is the final form that can be obtained by the algorithm used in this function, the function returns −1 as the return value without setting any values in array `S`.

[Program]

```
○Main program: Main
○Integer type: S[5], K, N, R          /*  1 ≤ K ≤ N  */
• K ← 3                     /*  Number of elements to be selected  */
• N ← 5                     /*  Number of total elements  */
- - - - - - - - - - - - - - - -
• R ← Init(S, N, K)
■ R = 0                                   α
│ • R ← Next(S, N)
■
- - - - - - - - - - - - - - - -  ←――――――――

○Integer type function: Init (Integer type: S[],
                              Integer type: N, Integer type: K)
○Integer type: L
▲ 1 ≤ K and K ≤ N
│ ■ L: 1, L ≤ N, 1
│ │ ▲ L ≤ K
│ │ │ • S[L] ← 1
│ │ ├―――――――
│ │ │ • S[L] ← 0
│ │ ▼
│ ■
│ • return 0
├―――――――
│ • return −1
▼

○Integer type function: Next(Integer type: S[], Integer type: N)
○Integer type: C, L, R
• C ← 0
• L ← 1
• R ← −1
■ L < N and R = −1
│ ▲ S[L] = 1
│ │ ▲ S[L+1] = 0
│ │ │ • S[L] ← 0
│ │ │ • S[L+1] ← 1
│ │ │ • Init(S, L−1, C)
│ │ │ • R ← 0
│ │ ├―――――――
│ │ │ • C ← C + 1
│ │ ▼
│ ▼
│ • L ← L + 1
■
• return R
```

Subquestion

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

(1) Modify the main program `Main` so that it can print the contents of array `S` whenever one state of a combination is obtained in array `S`. The following subprogram is used for printing:

○`Subprogram Dump(Integer type: S[], Integer type: N)`

Arguments: `S[]` and `N` are input arguments.

Description: The function prints the values stored in the first `N` elements of array `S` on a single line.

For printing, the part indicated by ***α*** in the main program `Main` is replaced with the following part:

```
• R ← Init(S, N, K)
■ R = 0
│  $\boxed{\quad A \quad}$
■
```

(2) The function `Next` searches the received array `S` from the smallest element number, and exchanges the contents of two consecutive elements that have the values $\boxed{\quad B \quad}$. Next, the function `Next` invokes the function `Init` for the part of array `S`, namely, $\boxed{\quad C \quad}$ the two consecutive elements. For example, if the contents of the element numbers 1 to 5 of array `S` are 1, 0, 1, 0, 1 at the beginning of the execution of the function `Next`, the contents of the element numbers 1 to 5 of array `S` at the termination of the execution are $\boxed{\quad D \quad}$.

(3) When this program is executed and the function `Init` is invoked from the function `Next`, the value of `N` received by the function `Init` ranges $\boxed{\quad E \quad}$, and the value of `K` ranges from 0 to 2. Thus, in some cases, the value of `N` and `K` received by the function `Init` may not satisfy 1 ≤ `K` ≤ `N`.

(4) When the execution of the main program `Main` is complete, the contents of the element numbers 1 to 5 of array `S` will be $\boxed{\quad F \quad}$.

Answer group for A

a) • `Dump(S, N)`

b) • `Dump(S, N)`
   • `R ←Next(S, N)`

c) • `R ←Next(S, N)`
   • `Dump(S, N)`

Answer group for B

a) 0, 1 that are found first

b) 0, 1 that are found last

c) 1, 0 that are found first

d) 1, 0 that are found last

Answer group for C

a) after

b) after and including

c) before

d) before and including

Answer group for D

a) 0, 1, 1, 0, 1 b) 1, 0, 0, 1, 1 c) 1, 0, 1, 1, 0 d) 1, 1, 0, 0, 1

Answer group for E

a) from 0 to 2 b) from 0 to 3 c) from 1 to 3 d) from 1 to 4

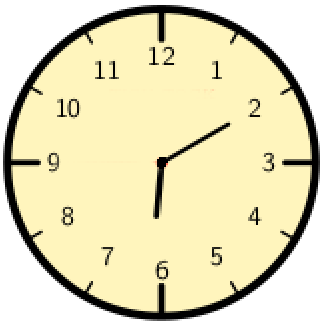
Answer group for F

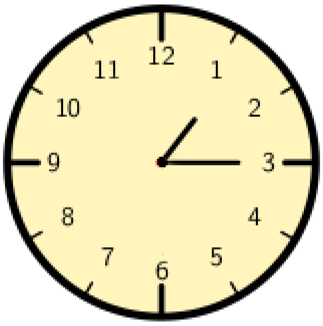
a) 0, 0, 0, 0, 0 b) 0, 0, 1, 1, 1 c) 1, 1, 1, 0, 0 d) 1, 1, 1, 1, 1

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.

Analog clocks indicate time using angles. Usually, an analog clock has three hands: the hour hand, minute hand, and second hand. Sometimes it has only two hands: the hour hand and minute hand. Every second, all three (or two) hands of the analog clock change their positions, creating different angles between them.





For instance, at 06:10, the angle between the hour hand and the minute hand is 125°, and at 01:15, the angle is 52.5°. Here, only acute, right, obtuse or straight angle between the hands are considered (namely, 0° ≤ the angle ≤ 180°).
Commonly, there are two types of time format: 12 hour and 24 hour. However, an analog clock can show time only in the 12 hour format. For example, at 06:10 and 18:10, the hour hand and minute hand of an analog clock are in the same position.

[Program Description]

(1) In this program, the analog clock is considered to have only two hands (hour and minute).

(2) The program reads a series of time data from the standard input. The input for the program is given in the following order. The first line contains the number of time data `n`, where 1 ≤ `n` ≤ 51. The next `n` lines contain two integer values `hh` (0 ≤ `hh` ≤ 23) and `mm` (0 ≤ `mm` ≤ 59), indicating the hour and minute of a particular time in the 24 hour time format. Two integer values are separated by one or more space characters.

(3) The program prints n lines as output. Each line contains three values: hour, minute, and the angle between the hour and minute hands. The output is sorted in ascending order of the angle. If there are time data with the same angle, then the earlier time should appear first (00:00 is the earliest time and 23:59 is the latest time)

(4) The following list shows an example of input data.

```
4
6 20
0 0
15 0
6 0
```

(5) The following list shows an example of output for the input data shown in (4) above.

```
 0  0   0.0
 6 20  70.0
15  0  90.0
 6  0 180.0
```

(6) In the program, a structure named time is used. This structure has three integer variables hh, mm, and angle, and one long integer variable sortValue. The variables hh and mm are used to store the hour value and minute value of the time data. The variable angle holds the 10 times value of the angle created by the hour and minute hands of the clock. The variable sortValue holds a value that is used to sort the list of time data.

(7) Two user defined functions are used.

(i) void setSortingValue(int i)

This function combines three values (hh, mm and angle) into a single value, and sets this value to the variable sortValue of a particular element having index i.

By using sortValue as the sort key, t[] can be sorted in ascending order of the angle and time.

(ii) void sort(int n)

This function sorts t[] in ascending order of sortValue, where n indicates the number of elements in t[].

[Program]

```
#include <stdio.h>

void setSortingValue(int);
void sort(int);

struct time {
   int hh, mm, angle;
   long sortValue;
} t[51];

void setSortingValue(int i) {
   t[i].sortValue = 10000 * (long)t[i].angle
                    + 100 * t[i].hh + t[i].mm;
}

void sort(int n) {
   int i,j;
   struct time temp;

   for (i = 0; i < n - 1; i++) {
      for (j = i + 1; j < n; j++) {
         if (      A      ) {
            temp = t[i];
            t[i] = t[j];
            t[j] = temp;
         }
      }
   }
}
```

```
int main() {
   int n, i, hAngle, mAngle, angle;

   scanf("%d", &n);

   for (i = 0; i < n; i++) {
      scanf("%d%d", &t[i].hh, &t[i].mm);
      hAngle = (t[i].hh % 12 * 60 + t[i].mm) * 5;
      mAngle = [      B      ];
      if (hAngle > mAngle) {
         t[i].angle = hAngle - mAngle;
      } else {
         t[i].angle = mAngle - hAngle;
      }
      if (t[i].angle > 1800) {
         t[i].angle = [      C      ];
      }
      setSortingValue(i);
   }

   sort(n);

   for (i = 0; i < n; i++) {
      printf("%2d %2d %5.1f\n", t[i].hh, t[i].mm, [      D      ]);
   }
}
```

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) `t[i].sortValue > t[j-1].sortValue`

b) `t[i].sortValue < t[j].sortValue`

c) `t[i].sortValue > t[j].sortValue`

d) `t[i+1].sortValue < t[j].sortValue`

Answer group for B

a) `t[i].mm * 5`　　b) `t[i].mm * 10`　　c) `t[i].mm * 30`

d) `t[i].mm * 60`　　e) `t[i].mm * 180`　　f) `t[i].mm * 360`

Answer group for C

a) `900 + t[i].angle`
b) `3600 - t[i].angle`
c) `t[i].angle - 900`
d) `t[i].angle – 1800`

Answer group for D

a) `t[i].angle % 10`
b) `t[i].angle % 3600`
c) `t[i].angle * 10`
d) `t[i].angle * 10.0`
e) `t[i].angle / 10`
f) `t[i].angle / 10.0`

Subquestion 2

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

When the values in `t[4]` are, `t[4].hh` = 23, `t[4].mm` = 59 and `t[4].angle` = 55, the value stored in `t[4].sortValue` will be $\boxed{\text{E}}$ after `SetSortingValue(4)` is executed.

Answer group

a) `551159`
b) `552359`
c) `5501159`
d) `5502359`

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

[Program Description]

(1) An education division in an IT company tasks its trainee programmers to develop a program that determines the day of the week of the birth date of given employees.

(2) Figure 1 shows the execution result of this program.

```
ID: 1
Department: IT
Name: Dela Cruz, John X.
BirthDate: 7/8/1974 (Monday)

ID: 2
Department: AC
Name: Victoria, Glenda Y.
BirthDate: 4/29/1980 (Tuesday)

ID: 3
Department: HR
Name: Hilary, Brent Z.
BirthDate: 10/25/1991 (Friday)
```

Figure 1. Execution result of the program

(3) The programs are composed of four classes: `Date`, `Person`, `Employee` and `BirthDayRooster`.

(4) The class `Date` contains a method `dayOfWeek()` that uses Zeller's algorithm for determining the day of the week. The method returns a value between 0 and 6 that represents the day of the week as follows:

0 – Saturday
1 – Sunday
2 – Monday
3 – Tuesday
4 – Wednesday
5 – Thursday
6 – Friday

[Program 1]

```
public class Date {
    private int month;
    private int day;
    private int year;

    public Date(int month, int day, int year) {
        this.month=month;
        this.day=day;
        this.year=year;
    }
    public int dayOfWeek() {
        int t, mm, dd, yy, cc;
        mm = month;
        yy = year;
        if (mm < 3) {
            mm = mm + 12;
            yy = yy - 1;
        }
        cc = (int) (yy / 100);
        yy = yy % 100;
        t = day + (int) (26 * (mm + 1) / 10) + yy;
        t = t + (int) (yy / 4) + (int) (cc / 4) - 2 * cc;
        dd = t % 7;
        if (dd < 0)
            dd = dd + 7;
        return dd;
    }
    public String toString() {
        return month+"/"+day+"/"+year;
    }
}
```

[Program 2]

```
public class Person {
    private String lastName;
    private String firstName;
    private char middleInitial;
    private Date birthdate;
    public Person(String lastName, String firstName,
                  char middleInitial, Date birthdate) {
        this.lastName = lastName;
        this.firstName = firstName;
        this.middleInitial = middleInitial;
        this.birthdate = birthdate;
    }
    public void setBirthdate(Date birthdate) {
        this.birthdate=birthdate;
    }
    public Date getBirthdate() {
        return birthdate;
    }
    public String toString() {
        return "Name: " + lastName + ", " + firstName
               + " " + middleInitial + ". \nBirthDate: " + birthdate;
    }
}
```

[Program 3]

```
public class Employee [      A      ] {
    private static int counter = 1;
    private int id;
    private String department;

    public Employee(String department, String lastName,
                    String firstName, char middleInitial,
                    Date birthdate) {
        [      B      ];
        id = counter++;
        this.department = department;
    }
    public int getId() {
        return id;
    }
    public String getDepartment() {
        return department;
    }
    public String toString() {
        return "ID: " + getId() + "\nDepartment: "
               + getDepartment() + "\n" + [      C      ];
    }
}
```

[Program 4]

```
public class BirthDayRooster {
    [     D      ] = {"Saturday", "Sunday", "Monday",
                      "Tuesday", "Wednesday", "Thursday", "Friday"};

    public static void main(String[] args) {
        Employee[] emp = {new Employee("IT", "Dela Cruz", "John",
                                       'X', new Date(7, 8, 1974)),
                          new Employee("AC", "Victoria", "Glenda",
                                       'Y', new Date(4, 29, 1980)),
                          new Employee("HR", "Hilary", "Brent",
                                       'Z', new Date(10, 25, 1991))};

        for (Employee e : emp) {
            System.out.println(e + " (" + [     E      ] + ")");
            System.out.println();
        }
    }
}
```

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) `extends BirthDayRooster`
b) `extends Date`
c) `extends Person`
d) `implements BirthDayRooster`
e) `implements Date`
f) `implements Person`

Answer group for B and C

a) `super()`
b) `super(lastName, firstName, middleInitial, birthdate)`
c) `super.toString()`
d) `this()`
e) `this(lastName, firstName, middleInitial, birthdate)`
f) `this.toString()`

Answer group for D

a) `private String[] day`
b) `static String[] day`
c) `String[] day`
d) `String[] static day`

Answer group for E

a) `day.e.getBirthdate().dayOfWeek()`
b) `day.emp.getBirthdate().dayOfWeek()`
c) `day[e.getBirthdate().dayOfWeek()]`
d) `day[emp.getBirthdate().dayOfWeek()]`

Subquestion 2

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

According to the method `dayOfWeek()` in the class `Date` shown in [Program 1], the day of the week of 01/20/1901 was F.

Answer group

a) Monday
b) Tuesday
c) Wednesday
d) Thursday
e) Friday
f) Saturday
g) Sunday