## May 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 this spring Fundamental IT Engineer Examination conducted?

Answer group
a) April b) May c) June d) July

Since the correct answer is " b) May ", 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<br>process<br>▼ | Indicates a one-way selection process.<br>If the conditional expression is `true`, then the process is executed. |
| | ▲ conditional expression<br>process 1<br>─<br>process 2<br>▼ | Indicates a two-way selection process.<br>If the conditional expression is `true`, then the process 1 is executed.<br>If it is `false`, then the process 2 is executed. |
| | ■ conditional expression<br>process<br>■ | Indicates a pre-test iteration process.<br>While the conditional expression is `true`, the process is executed repeatedly. |
| | ■<br>process<br>■ conditional expression | Indicates a post-test iteration process.<br>The process is executed, and then while the conditional expression is `true`, the process is executed repeatedly. |
| | ■ variable: init, cond, incr<br>process<br>■ | Indicates an iteration process.<br>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.<br>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.

---

## Notations used for E-R diagrams

In questions that use E-R diagrams, the following notations are used unless otherwise stated.

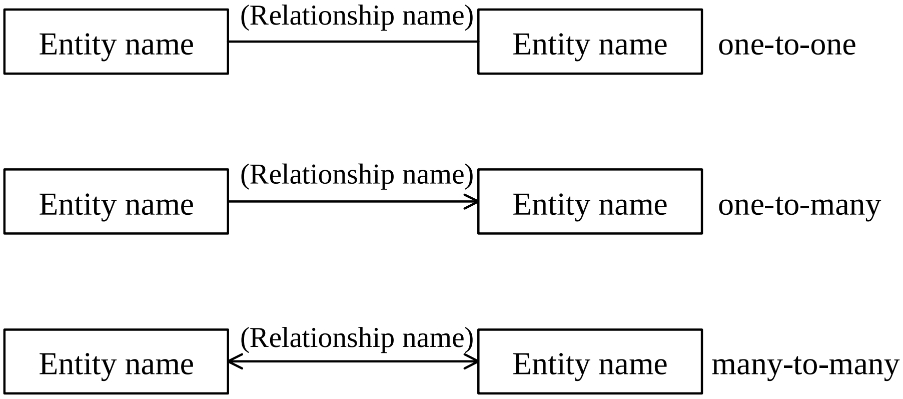


Figure  Notations used for entities and relationships

1. Entities are represented by rectangles.
2. Entity names are indicated within the rectangles.
3. A relationship between entities is represented by a line.
   The relationship name is indicated at the side of the line as “(Relationship name)”.
   The relationship name can be omitted.
4. A “one-to-one” relationship is represented by a line.
   A “one-to-many” relationship is represented by a line with an arrow pointing towards the “many” side.
   A “many-to-many” relationship is represented by a line with an arrow on both ends.

---

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

**Q1.** Read the following description concerning memory management, and then answer Subquestions 1 and 2.

Many applications for mobile devices make use of data that is partially stored in a data buffer on a storage media of the devices. Most of the data is stored in an external server accessed through the Internet. When a user tries to access data not found in the device, the application connects to the server to retrieve the data. The new data will replace some of the data existing in the device due to memory constraints. An efficient implementation reduces the number of times data is retrieved from the server. This efficiency is influenced by factors such as the amount of a data buffer needed to store the partial data, the method of determining the data to be replaced by the new data, and the pattern of how the user accesses the data.

In the case of an application that accesses 6 different data blocks repeatedly by using a data buffer that can hold 5 data blocks, fetching data from an external server is necessary all the time.

Figure 1 illustrates how the data buffer is being replaced in this case. In Figure 1, the set of data blocks at the top represents the data being accessed by the application, and the set of data blocks at the bottom represents the data buffer and its corresponding data. When the application accesses the data block [6], it is fetched from the server and replaces [1]. When the application accesses [1] after [6], it is fetched from the server and replaces [2], and so on.

This uses a First-In-First-Out (FIFO) method. But if the data buffer can hold 6 data blocks, then there will be only 6 fetches at the start of the application, and there is no need to fetch data till the end of the application.

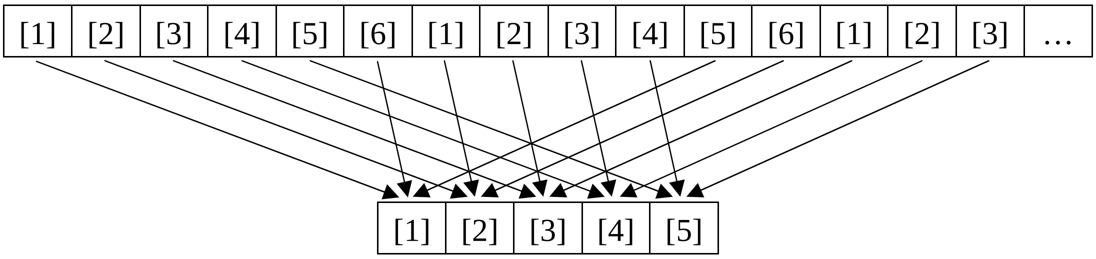


Figure 1. Accessing 6 data blocks repeatedly by using data buffer

### Subquestion 1

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

Assuming that an application for a mobile device accesses 6 data blocks [1], [2], … , [6] in the following order.

| | | | | | | | | | ▼ | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| [1] | [2] | [3] | [5] | [6] | [1] | [2] | [3] | [4] | [6] | [1] | [2] | [5] | [6] | [4] | [1] | [2] |

The device has a data buffer that can hold 5 data blocks. When the application starts, the data blocks [1], [2], … , [5] have already been loaded into the data buffer in this sequence.

| | | | | |
|---|---|---|---|---|
| [1] | [2] | [3] | [4] | [5] |

When the FIFO method is used, at the point ▼ shown above immediately after the data block [6] is accessed, the contents of the data buffer will be A, and at the end of the series of data block accesses, the contents of the data buffer will be B with a total of C data block replacements for the entire duration of the application.

Answer group for A and B

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| a) | [1] | [2] | [3] | [6] | [4] | b) | [1] | [2] | [5] | [6] | [4] |
| c) | [4] | [1] | [2] | [5] | [6] | d) | [4] | [2] | [3] | [5] | [6] |
| e) | [5] | [6] | [1] | [2] | [4] | f) | [6] | [1] | [2] | [3] | [4] |

Answer group for C

a) 3 b) 4 c) 5 d) 8 e) 9 f) 10

### Subquestion 2

From the answer group below, select the correct answer to be inserted into each blank in Table 1.

Two alternative methods are explored to replace the FIFO method.

The first alternative is called the Most Frequently Used (MFU) method. In the MFU method, the system counts how many times each data block was used while it remains in the data buffer. The data block with the lowest count is the data block that is replaced when a new data block needs space. If multiple data blocks have the same count, then the oldest data block is replaced according to the FIFO method.

The second alternative is called the Least Recently Used (LRU) method. In the LRU method, the system keeps track of which data block was used when while it remains in the data buffer. The least recently used data block is the data block that is replaced when a new data block needs space.

Table 1 shows the analysis of the two methods for different program behaviors.
Here, the data buffer can hold only 5 data blocks.

Table 1. Analysis of the two methods

| Program behavior | MFU method | LRU method |
|---|---|---|
| A program accesses different data blocks randomly throughout its use. | Since each new access will require to bring in new data block, there will be no improvement. | Since each new access will require to bring in new data block, there will be no improvement. |
| A program accesses 8 data blocks with the following cyclic pattern: First 3 data blocks → 2 data blocks out of the remaining 5 → first 3 data blocks → 2 data blocks out of the remaining 5 → … . | D | D |
| A program first accesses 5 data blocks repeatedly (10 times), then accesses the other 3 data blocks repeatedly (10 times). | E | F |

Answer group

a) For the first 5 data blocks, once they are loaded into the data buffer, no replacement will take place during their repeated use. For the next 3 data blocks, replacement will always take place for each succeeding data block access.

b) For the first 5 data blocks, once they are loaded into the data buffer, no replacement will take place during their repeated use. Similarly, for the next 3 data blocks, once they are loaded, no replacement will take place during their repeated use.

c) No data blocks out of 8 will stay in the data buffer permanently during the program execution. All 8 data blocks will be subject to the replacement.

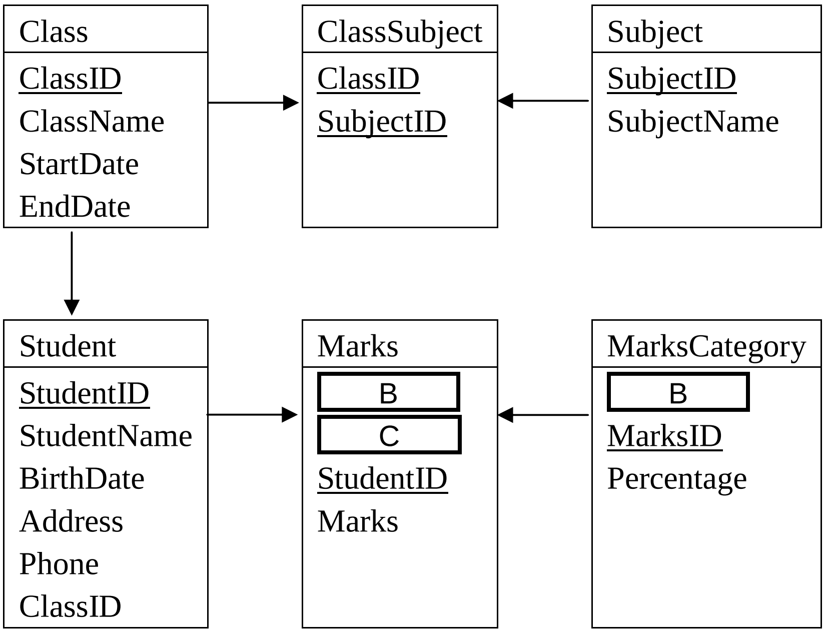
d) The 3 data blocks out of 8 will stay in the data buffer permanently during the program execution. Other 5 data blocks will be subject to the replacement.

**Q2.** Read the following description concerning a database for student management, and then answer Subquestions 1 and 2.

An IT faculty of a university is going to develop the student management system. Ms. *T* is in charge of getting requirements for this system. She summarizes main requirements:

1) Each student can be enrolled in only one class. Information about the student are ID, name, birth date, address, and phone number. Information about the class are ID, name, start date, and end date.
2) Some different subjects will be taught in one class.
3) Each subject has mark items such as exercises, tests, and final exam, to evaluate the students' competence for the subject. The responsible lecturer will decide the mark items and their contribution percentage. For example, the total marks of the subject CNET will be evaluated by the mark items Lab, Test, Project, Practical exam, and Final exam with percentages 15%, 15%, 20%, 20%, and 30% respectively.

Based on the above requirements, Ms. *T* is designing the database for the system. Figure 1 shows an E-R diagram of the database she is now creating. This E-R diagram is incomplete, and "one-to-many" relationship from $\boxed{\text{A}}$ to MarksCategory is missing.



(Note) A solid underline _____ indicates an attribute of a primary key.

Figure 1. E-R Diagram of the database (incomplete)

### Subquestion 1

From the answer groups below, select the correct answer to be inserted into each blank in the above description and Figure 1.

Answer group for A

- a) Class
- b) ClassSubject
- c) Student
- d) Subject

Answer group for B and C

- a) <u>ClassID</u>
- b) <u>MarksID</u>
- c) <u>StudentID</u>
- d) <u>SubjectID</u>

### Subquestion 2

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

At the end of a semester, the responsible lecturer outputs the list of all students with total marks for the specified class and subject as shown below, based on the data in the database.

```
Student ID  Student Name  Total Marks
   S014       B. Eeeee         92
   S015       B. Fffff         65
   S035       J. Ccccc         89
   ...        ...              ...
```

Total marks of each student is sum of individual marks of mark items multiplied with their contribution percentages. For example, when the total marks of the subject CNET will be evaluated by the mark items Lab, Test, Project, Practical exam, and Final exam with percentages 15%, 15%, 20%, 20%, and 30% respectively, the total marks will be:

$$\text{Total marks} = (\text{Lab marks} \times 15 + \text{Test marks} \times 15 + \text{Project marks} \times 20 + \text{Practical exam marks} \times 20 + \text{Final exam marks} \times 30) \div 100$$

Here, ClassID and SubjectID are given by the host variables `:ClassID` and `:SubjectID` respectively.

```
SELECT   Student.StudentID, StudentName,
                D           AS TotalMarks
FROM     Class, Student, Marks, MarksCategory
WHERE    Class.ClassID = :ClassID
  AND    Marks.SubjectID = :SubjectID
  AND    Class.ClassID = Student.ClassID
  AND           E
GROUP BY         F
```

Answer group

a) `AVG(Marks * Percentage)`

b) `Student.StudentID = Marks.StudentID`
`AND Marks.SubjectID = MarksCategory.SubjectID`

c) `Student.StudentID = Marks.StudentID`
`AND Marks.SubjectID = MarksCategory.SubjectID`
`AND Marks.MarksID = MarksCategory.MarksID`

d) `Student.StudentID, StudentName`

e) `StudentID`

f) `StudentID, StudentName`

g) `SUM(Marks * Percentage) / 100`

**Q3.** Read the following description concerning network trouble-shooting, and then answer Subquestions 1 and 2.

Company *U* has setup its IP network which mainly provides Intranet application services, network-based office automation, and the Internet access services to its employees. The network can be accessed via wired Ethernet connections within the company covering two physical rooms. In the network, Internet access router and 3 Layer-2 switches (SW1, SW2 and SW3) are installed. Figure 1 shows the configuration of the network in Company *U*.

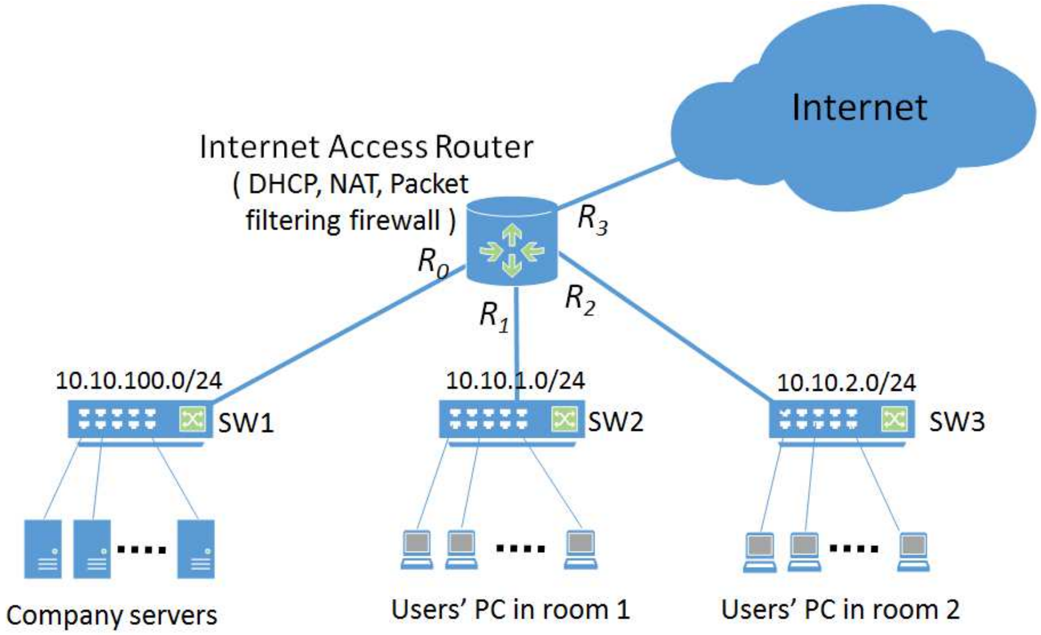


Figure 1. Configuration of the network in Company *U*

Table 1. OSI reference model with examples of possible problems

| Layer | Examples of possible problems |
|---|---|
| 7 Application | • DSN misconfiguration<br>• Server problem<br>• Application program problem |
| 6 Presentation | |
| 5 Session | |
| 4 Transport | • TCP or UDP port blocking |
| 3 Network | • IP address misconfiguration<br>• IP routing table incomplete<br>• Default router problem |
| 2 Data link | • MAC address blocking<br>• MAC level authentication |
| 1 Physical | • Cable unplug<br>• Cable defect<br>• Network port malfunction |

In Company *U*, Mr. *M* has a responsibility to take care of any technical problems relevant to the network services. Based on his 3-year experiences, he has developed a procedure for network trouble-shooting which can cover 90% of daily network problems. The procedure uses bottom-up approach to identify the causes of the problems according with OSI reference model as shown in Table 1.

## Subquestion 1

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

One day, an organization change was performed. A large number of employees changed their workplaces from room 1 to room 2, or from room 2 to room 1. After the change, some employees made complaints about the network connection. When being informed about the problems, Mr. *M* took the first step to determine whether the problems can be categorized as Layer 1 or 2 problems or not, by checking the information in the Layer-2 switches used by those employees.

Table 2 shows the list of users who could not use the network, and Table 3 shows the snapshot of the switch usage data taken by Mr. *M*.

Table 2. List of users who could not use the network

| User name | Switch/Port ID | User PC’s MAC address |
|---|---|---|
| User *A* | SW2 Port09 | `AA:AA:AA:AA:AA:AA` |
| User *B* | SW2 Port13 | `BB:BB:BB:BB:BB:BB` |
| User *C* | SW3 Port01 | `CC:CC:CC:CC:CC:CC` |
| User *D* | SW3 Port12 | `DD:DD:DD:DD:DD:DD` |
| User *E* | SW3 Port21 | `EE:EE:EE:EE:EE:EE` |

Table 3. Snapshot of the switch usage data

| Switch/Port ID | Link status | Frame Send/Receive CRC error | Recognized MAC | MAC filter setting |
|---|---|---|---|---|
| SW2 Port09 | Down | 0% | Not Detected | Allow: All |
| SW2 Port13 | Up | 0% | `BB:BB:BB:BB:BB:BB` | Allow: All |
| SW3 Port01 | Up | 0% | `CC:CC:CC:CC:CC:CC` | Allow: `DD:DD:DD:DD:DD:DD` |
| SW3 Port12 | Up | 85% | `DD:DD:DD:DD:DD:DD` | Allow: `DD:DD:DD:DD:DD:DD` |
| SW3 Port21 | Up | 0% | `EE:EE:EE:EE:EE:EE` | Allow: All |

Judging from Table 2 and Table 3, Mr. *M* found out that $\boxed{\quad\text{A}\quad}$ had cabling or switch port problems. He solved these problems by taking the steps such as checking the cable plugging status, replacing the network cable, and changing the plugging switch port.
Also, Mr. *M* found out that $\boxed{\quad\text{B}\quad}$ had MAC configuration problem. He solved this problem by changing the MAC filter setting.

Answer group

| | | | |
|---|---|---|---|
| a) | User *A*, User *B* and User *E* | b) | User *A* and User *C* |
| c) | User *A* and User *D* | d) | User *C* |
| e) | User *C* and User *D* | f) | User *D* |

## Subquestion 2

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

Mr. *M* was informed that User *F*'s PC could not send IP packets to both Intranet and the Internet. By checking the IP address configuration in Table 4 and the IP routing table in Table 5, Mr. *M* found out that this was a Layer 3 problem concerning the IP routing table. Accordingly, he added the missing routing record $\boxed{\qquad\text{C}\qquad}$ to the IP routing table in Table 5, and then User *F*'s PC was able to access both Intranet and the Internet.
Here, the IP addresses of interfaces $R_0$, $R_1$, $R_2$ and $R_3$ at Internet access router in Figure 1 are configured as 10.10.100.254, 10.10.1.254, 10.10.2.254 and 202.170.10.1, respectively.

Table 4. IP address configuration of User *F*'s PC

| User name | User PC's IP address | Netmask | Switch/Port ID |
|---|---|---|---|
| User *F* | 10.10.2.55 | 255.255.255.0 | SW3 Port19 |

Table 5. IP routing table of User *F*'s PC

| Destination | Netmask | Gateway |
|---|---|---|
| 10.10.2.0 | 255.255.255.0 | ( not specified ) |

After clearing Layer 1-3 problems, Mr. *M* set about solving the remaining problem related to Web site accesses from a Web browser. In order to identify the cause, he tried to access the Web site "http://www.itpec.org", whose IP address is 64.22.66.88, from the user PC's Web browser. As the test result showed that $\boxed{\qquad\text{D}\qquad}$, he was confident that the cause of the problem was DNS misconfiguration. Then he changed the related settings, and the problem was solved.

Answer group for C

| | Destination | Netmask | Gateway |
|---|---|---|---|
| a) | 0.0.0.0 | 0.0.0.0 | 10.10.1.254 |
| b) | 0.0.0.0 | 0.0.0.0 | 10.10.2.254 |
| c) | 0.0.0.0 | 0.0.0.0 | 202.170.10.1 |
| d) | 10.0.0.0 | 255.0.0.0 | 10.10.1.254 |
| e) | 10.10.0.0 | 255.255.0.0 | 10.10.2.254 |
| f) | 10.10.2.0 | 255.255.255.0 | 202.170.10.1 |

Answer group for D

a) http://www.itpec.org was accessible and http://64.22.66.88 was accessible
b) http://www.itpec.org was accessible and http://64.22.66.88 was not accessible
c) http://www.itpec.org was not accessible and http://64.22.66.88 was accessible
d) http://www.itpec.org was not accessible and http://64.22.66.88 was not accessible

**Q4.** Read the following description concerning Web site security, and then answer Subquestions 1 through 3.

A mid-sized company has developed a Web application for online sales reporting system to enable its sales force to process sales data from remote sites. The application transfers certain secured sales data through the Internet, and that is a big concern for the company. Therefore, the company has established an SSL (Secure Sockets Layer) enabled Web site for this application.

Processing information securely through the Internet means that the information need to be transmitted between the sender and the receiver in a manner that makes it difficult for other people to intercept and read. SSL takes care of this, and it works through a combination of programs and encryption/decryption routines that exist on both the application on the Web site and the browser program on the user computer.

## Subquestion 1

From the answer group below, select the correct answer to be inserted into each blank ☐ in Figure 1.

In HTTPS communication, the handshake is the most complicated phase in the process. The handshake synchronizes the client and the server up with the encryption methods and keys that will be used for the subsequent communication.

The HTTPS handshake consists of multiple steps as shown in Figure 1.

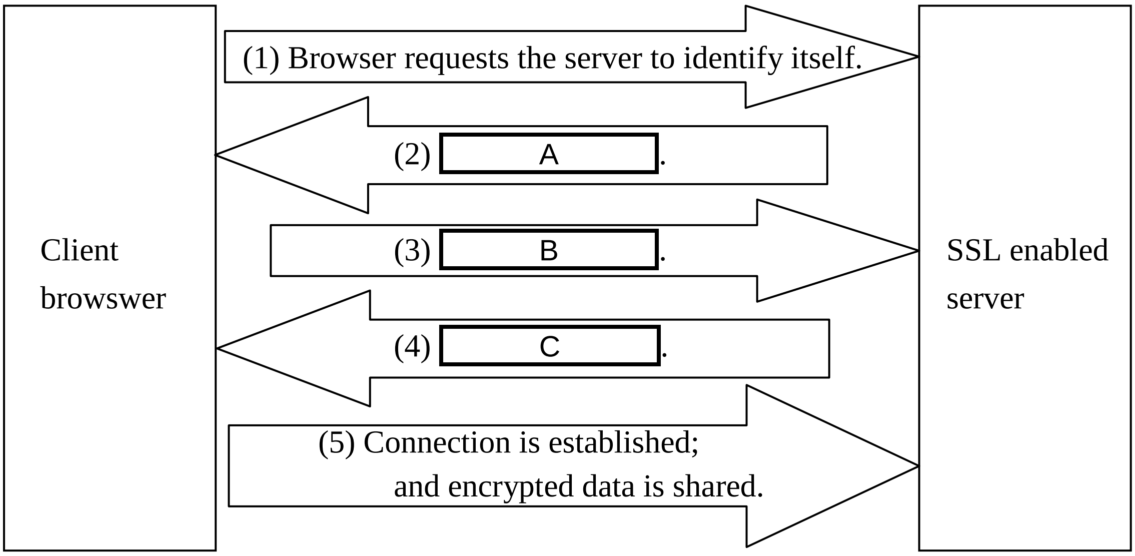

Client browswer
(1) Browser requests the server to identify itself.
(2) A .
(3) B .
(4) C .
(5) Connection is established; and encrypted data is shared.
SSL enabled server

Figure 1. Flow of HTTPS handshake

Answer group

a) Browser sends a copy of the SSL certificate to the server
b) Browser verifies the certificate and sends an acknowledgment message to the server
c) Server sends a copy of the SSL certificate to the browser
d) Server sends a digitally signed acknowledgement to the browser

### Subquestion 2

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

The HTTPS protocol has several commands such as GET, POST, PUT and DELETE. The application uses POST command to transfer secured sales data from remote sites. The software developer wanted to test if SSL is properly configured or not, so he used a network monitoring tool and made a POST request to the server https://www.example.com (IP address: 93.184.216.119) from his PC (IP address: 192.168.1.103).
In the monitoring tool, he found a “Server Hello” packet header as shown in Figure 2.
Here, the port number for https service is 443.

```
Frame 43: 1414 bytes on wire (11312 bits), 1414 bytes captured (11312 bits) on interface 0
Ethernet II, Src: Tp-LinkT_d0:d2:44 (74:ea:3a:d0:d2:44), Dst: Azurewav_e7:10:41 (48:5d:60:e7:10:41)
Internet Protocol Version 4, Src:       D       , Dst:       E
Transmission Control Protocol, Src Port: https (443), Dst Port: 3par-evts (5781), Seq: 1, Ack: 230, Len: 13
    Source port: https (443)
    Destination port: 3par-evts (5781)
    [Stream index: 6]
    Sequence number: 1    (relative sequence number)
    [Next sequence number: 1361    (relative sequence number)]
    Acknowledgment number: 230    (relative ack number)
    Header length: 20 bytes
```

Figure 2. “Server Hello” packet header

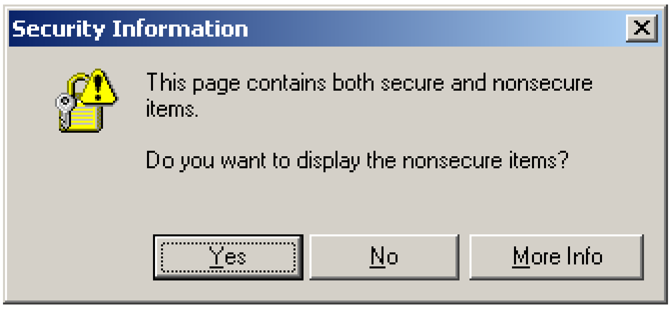
Answer group

a) 93.184.216.119
b) 192.168.1.103

## Subquestion 3

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

Once SSL was configured properly and tested successfully as well, the software developer started a browser on his PC and tried to browse the Web pages. He found that the system was working smoothly with non-secured Web pages. When he tried to browse a secured page at https://www.example.com/order, he got the security message as shown in Figure 3.



This page contains both secure and nonsecure items.

Do you want to display the nonsecure items?

Figure 3. Security message

Source code

```
       <!DOCTYPE html>
       <html>
       <head>
       <title>Secured Page</title>
(1) →  <link rel="stylesheet" type="text/css"
             href="http://www.example.com/themes/main.css">
(2) →  <link rel="stylesheet" type="text/css"
             href="http://www.example.com/themes/header.css">
(3) →  <script type="text/javascript"
               src="https://www.example.com/scripts/jquery.js"></script>
       </head>
       <body>
       <h1>Order Form</h1>
       <div>
(4) →  <img src="http://www.example.com/images/shirt.jpg">
       <h4>Polo T-Shitr</h4>
       </div>
       <form action="/order/create" method="post">  Quantity:
       <input type="text" name="quantity" value="1">
       <input type="submit" value="Buy">
       </form>
       </body>
       </html>
```

Figure 4. HTML source code of the requested Web page

When the software developer opened the source file, he found the HTML (HyperText Markup Language) source code of the requested Web page, as shown in Figure 4. HTML is used most commonly for publishing Web pages.

In HTML, the contents are encoded using <tags> for annotating and formatting. Different tags are used for different purposes. A few of them are used to mention external resources such as images, link and Java scripts.

The tag `<link rel="…" …>` is used for making a relational link to an external resource of type style sheets.

The tag `<script src="…" …>` is used for embedding Java script files.

The tag `<img src="…" …>` is used for embedding images to the page.

Here, `rel="…"` and `src="…"` contain path information for a physical file. The path can be either an absolute (full) path or a relative path.

In the HTML source code in Figure 4, four tags (marked as (1), (2), (3) and (4) with "→") refer to the external resources. Out of these tags, the tags that caused the security message as shown in Figure 3 were [ F ].

Answer group

| | |
|---|---|
| a) (1), (2) and (3) | b) (1), (2), (3) and (4) |
| c) (1), (2) and (4) | d) (1), (3) and (4) |

**Q5.** Read the following description concerning program development and testing, and then answer Subquestion.

Ministry of Labor (MOL) maintains a statistic file that contains the information about population and employment status by state.
A staff of MOL creates a program which reads the statistic file and outputs a statistic report.

[Program description]

(1) Each record of the statistic file has 4 fields. Those are, from left to right, the name of the state, population, adult population, employed and unemployed. The following is an example of the contents of the statistic file:

| State | Population | Adult Population | Employed | Unemployed |
|---|---|---|---|---|
| East | 1240000 | 1040000 | 900000 | 100000 |
| North | 2520000 | 2080000 | 1760000 | 240000 |
| South | 1320000 | 1060000 | 860000 | 140000 |
| West | 2600000 | 2120000 | 1680000 | 320000 |

(2) For each record, the program calculates and outputs the labor force rate and unemployment rate, by using the following formulas:

$$\text{Labor force rate (\%)} = 100 \times (\text{ Employed} + \text{Unemployed }) \div \text{Adult Population}$$
$$\text{Unemployment rate (\%)} = 100 \times \text{Unemployed} \div (\text{ Employed} + \text{Unemployed })$$

(3) When all the records are processed, the program outputs the total of population, adult population, employed and unemployed, and the nationwide values of labor force rate and unemployment rate. After that, the program outputs the highest unemployment rate with the name of that state, and the lowest unemployment rate with the name of that state.

(4) The following is an example of the statistic report:

```
State        Pop  AdultPop       Emp     Unemp  Labor%  Unemp%
------  --------  --------  --------  --------  ------  ------
East     1240000   1040000    900000    100000    96.2    10.0
North    2520000   2080000   1760000    240000    96.2    12.0
South    1320000   1060000    860000    140000    94.3    14.0
West     2600000   2120000   1680000    320000    94.3    16.0
------  --------  --------  --------  --------  ------  ------
TOTAL    7680000   6300000   5200000    800000    95.2    13.3

Highest Unemp%: 16.0, State: West
Lowest  Unemp%: 10.0, State: East
```

[Program]

```
     ○ character type: State, StateHigh ← "????", StateLow ← "????"
     ○ integer type:   Pop, AdultPop, Emp, Unemp
     ○ integer type:   TotalPop ← 0, TotalAdultPop ← 0,
                       TotalEmp ← 0, TotalUnemp ← 0
     ○ float type:     LaborRate, UnempRate
 X →○ float type:     UnempRateHigh ← 0.0, UnempRateLow ← 100.0

     • OpenFile("statistic")      /* Open the statistic file */
     • Put("State        Pop  AdultPop       Emp     Unemp  Labor%  Unemp%")
     • Put("------  --------  --------  --------  --------  ------  ------")
     • GetRecord(State, Pop, AdultPop, Emp, Unemp)
     ■ Not end-of-file      /* Loop until the statistic file reaches the end-of-file */
     │ • LaborRate ← 100.0 × (Emp + Unemp) ÷ AdultPop
     │ • UnempRate ← 100.0 × Unemp ÷ (Emp + Unemp)
 Y → │ ▲ UnempRate > UnempRateHigh
     │ │ • StateHigh ← State
     │ │ • UnempRateHigh ← UnempRate
     │ ▼
 Z → │ ▲ UnempRate < UnempRateLow
     │ │ • StateLow ← State
     │ │ • UnempRateLow ← UnempRate
     │ ▼
     │ • TotalPop ← TotalPop + Pop
     │ • TotalAdultPop ← TotalAdultPop + AdultPop
     │ • TotalEmp ← TotalEmp + Emp
     │ • TotalUnemp ← TotalUnemp + Unemp
     │ • Put(State, Pop, AdultPop, Emp, Unemp, LaborRate, UnempRate)
     │ • GetRecord(State, Pop, AdultPop, Emp, Unemp)
     ■
     • Put("------  --------  --------  --------  --------  ------  ------")
     • LaborRate ← 100.0 × (TotalEmp + TotalUnemp) ÷ TotalAdultPop
     • UnempRate ← 100.0 × TotalUnemp ÷ (TotalEmp + TotalUnemp)
     • Put("TOTAL", TotalPop, TotalAdultPop,
                    TotalEmp, TotalUnemp, LaborRate, UnempRate)
     • Put()      /* Print a blank line */
     • Put("Highest Unemp%: ", UnempRateHigh, ", State: ", StateHigh)
     • Put("Lowest  Unemp%: ", UnempRateLow,  ", State: ", StateLow )
     • CloseFile("statistic")      /* Close the statistic file */
```

Note: The function `GetRecord(`$v_1$`, `$v_2$`, …)` reads the next record from the statistic file, and stores the values into variables $v_1$, $v_2$, … .
The function `Put(`$p_1$`, `$p_2$`, …)` prints variables/constants $p_1$, $p_2$, … in one line.

## Subquestion

From the answer groups below, select the correct answer to be inserted into each blank in the following description.
Note that the shaded parts are not shown.

The program development is now in test phase, and testing is performed by using various test cases. Four test cases and their test results are shown below:

[Test case 1]

This is the case where two states have the same unemployment rate for both the highest and lowest rates. In this case, the program outputs the following report.

```
State        Pop  AdultPop       Emp     Unemp  Labor%  Unemp%
------  --------  --------  --------  --------  ------  ------
East     1200000   1040000    900000    100000    96.2    10.0
North    1200000   1050000    900000    100000    95.2    10.0
South    1200000   1060000    850000    150000    94.3    15.0
West     1200000   1070000    850000    150000    93.5    15.0
------  --------  --------  --------  --------  ------  ------
TOTAL    4800000   4220000   3500000    500000    94.8    12.5

Highest Unemp%: 15.0, State:    A
Lowest  Unemp%: 10.0, State:
```

[Test case 2]

This is the particular case where all the states have 0% unemployment rate. In this case, expected output cannot be obtained. According to the program logic, the states shown on the last two lines are as follows:

```
State        Pop  AdultPop       Emp     Unemp  Labor%  Unemp%
------  --------  --------  --------  --------  ------  ------
East     1200000   1040000    900000         0    86.5     0.0
North    1200000   1050000    900000         0    85.7     0.0
South    1200000   1060000    850000         0    80.2     0.0
West     1200000   1070000    850000         0    79.4     0.0
------  --------  --------  --------  --------  ------  ------
TOTAL    4800000   4220000   3500000         0    82.9     0.0

Highest Unemp%:  0.0, State:    B
Lowest  Unemp%:  0.0, State:
```

[Test case 3]

The problem in test case 2 can be resolved by changing the operator “>” to “≥” on line Y, and changing the operator “<” to “≤” on line Z.

After this change, another particular case where all the states have 100% unemployment rate is tested. The problem is resolved, and the program outputs the following report.

```
State          Pop  AdultPop       Emp     Unemp  Labor%  Unemp%
------  --------  --------  --------  --------  ------  ------
East     1200000   1040000         0    900000    86.5   100.0
North    1200000   1050000         0    900000    85.7   100.0
South    1200000   1060000         0    850000    80.2   100.0
West     1200000   1070000         0    850000    79.4   100.0
------  --------  --------  --------  --------  ------  ------
TOTAL    4800000   4220000         0   3500000    82.9   100.0

Highest Unemp%:100.0, State: 
Lowest  Unemp%:100.0, State: C
```

[Test case 4]

After the change mentioned in test case 3, test case 1 is tested again. This time, because the program logic is changed, the program outputs the following report.

```
State          Pop  AdultPop       Emp     Unemp  Labor%  Unemp%
------  --------  --------  --------  --------  ------  ------
East     1200000   1040000    900000    100000    96.2    10.0
North    1200000   1050000    900000    100000    95.2    10.0
South    1200000   1060000    850000    150000    94.3    15.0
West     1200000   1070000    850000    150000    93.5    15.0
------  --------  --------  --------  --------  ------  ------
TOTAL    4800000   4220000   3500000    500000    94.8    12.5

Highest Unemp%: 15.0, State: 
Lowest  Unemp%: 10.0, State: D
```

Incidentally, the problem mentioned above can be resolved in a different way by replacing the initial values on line X by, for example, the following values:

○ `float type:   UnempRateHigh ←` E, `UnempRateLow ←` F

Answer group for A through D

a) `????`    b) `East`    c) `North`    d) `South`    e) `West`

Answer group for E and F

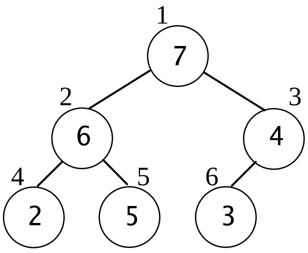
a) `-0.1`    b) `0.1`    c) `99.9`    d) `100.1`

**Q6.** Read the following description of a heap sort program and the program itself, and then answer Subquestions 1 through 3.

"Heap" is a kind of tree-based data structure. A heap can be defined as a binary tree, in which for each node, the values of its children are less than or equal to the node value (this is called max heap) or greater than or equal to the node value (min heap).
In a max heap, for each node except the root node, the value of the node is less than or equal to the value of its parent. Hence, in a max heap, the root contains the maximum value.

[Program Description]

(1) The program `HeapSort` sorts a given array `A[]` with `n` elements by adopting the heap sort algorithm. Heap sort is a comparison-based sorting algorithm. Figure 1 shows an example of a max heap and its array representation. The node index *i* is associated with `A[`*i*`]`. Here, the index of array `A[]` starts at 1.



Number of elements `n`: 6

| | | | | | | |
|---|---|---|---|---|---|---|
| (index) | 1 | 2 | 3 | 4 | 5 | 6 |
| Values in `A[]`: | 7 | 6 | 4 | 2 | 5 | 3 |

Figure 1. Example of a max heap and its array representation (heap size = 6)

(2) For any arbitrary node index *i*, the index of its left child is $2 \times i$, the index of its right child is $2 \times i + 1$, and the index of its parent is (integer quotient of) $i \div 2$.

(3) The program `BuildHeap` builds a max heap on the input array `A[]`.

(4) Sorting operation proceeds as follows:

- (i) Build the max heap with `n` elements `A[1]`, `A[2]`, … , `A[n]`. Then, the maximum value among them is obtained in the root `A[1]`. So, exchange the value in `A[1]` with the value in `A[n]`. Thus, the maximum value is obtained in `A[n]`.
- (ii) Discard the sorted element `A[n]` from the heap. In the heap with `n-1` elements, any parent node except the root node keeps the heap property, but the new root node may violate the heap property as it has been exchanged with the last node.
- (iii) Build the max heap with `n-1` elements `A[1]`, `A[2]`, … , `A[n-1]`. Then, the maximum value among them is obtained in `A[1]`. So, exchange the value in `A[1]` with the value in `A[n-1]`. Thus, the next-maximum value is obtained in `A[n-1]`.
- (iv) In this way, execute the same process repeatedly for the heap size `n-2` down to the heap size 2. Eventually, `A[]` will contain `n` sorted elements in ascending order.

[Program]

```
    ○ program: HeapSort(integer type: A[], integer type: n)
      ○ integer type: i, temp

      • BuildHeap(A, n)
      • i ← n
      ■ i ≥ 2
      │ • temp ← A[1]
      │ • A[1] ← A[i]
      │ • A[i] ← temp
      │ • i ← i - 1
      │ • Heapify(A, 1, i)
      ■

    ○ program: BuildHeap(integer type: A[], integer type: n)
      ○ integer type: i

      ■ i: n ÷ 2, i ≥ 1, -1
      │ • Heapify(A, i, n)
      ■

    ○ program: Heapify(integer type: A[],
                        integer type: i, integer type: heapsize)
      ○ integer type: lChild, rChild, largest, temp
                    /* lChild: left child, rChild: right child  */

      • lChild ← [     A     ]
      • rChild ← [     B     ]
      ▲ lChild ≤ heapsize and A[lChild] > A[i]
X →   │ • largest ← lChild
     ─┼──
      │ • largest ← i
      ▼
      ▲ rChild ≤ heapsize and A[rChild] > A[largest]
Y →   │ • largest ← rChild
      ▼
      ▲ largest ≠ i
      │ • temp ← A[i]
      │ • A[i] ← A[largest]
      │ • A[largest] ← temp
      │ • Heapify(A, largest, heapsize)
      ▼
```

## Subquestion 1

From the answer group below, select the correct answer to be inserted into each blank $\boxed{\phantom{XXXX}}$ in the above program.

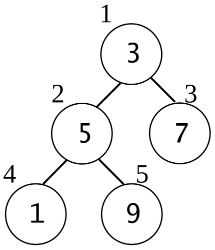
Answer group

| | |
|---|---|
| a) `2 × i` | b) `2 × i + 1` |
| c) `2 × i + 2` | d) `i ÷ 2` |

## Subquestion 2

From the answer groups below, select the correct answer to be inserted into each blank $\boxed{\phantom{XXXX}}$ in the following description and Figure 4.

Consider the case where the unsorted data shown in Figure 2 is passed to `HeapSort`.



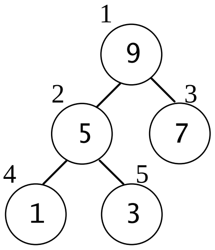
Number of elements `n`: 5

| (index) | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| values in `A[]`: | 3 | 5 | 7 | 1 | 9 |

Figure 2. Unsorted data passed to `HeapSort` (heap size = 5)

First, `HeapSort` executes `BuildHeap(A, n)`. When returned from `BuildHeap(A, n)`, the max heap is built in `A[]`, and the root `A[1]` contains the maximum value 9, as shown in Figure 3.

During this process, the program `Heapify` is called several times, and `largest ← lChild` on line X is executed $\boxed{\text{C}}$ time(s), and `largest ← rChild` on line Y is executed $\boxed{\text{D}}$ time(s).



Number of elements `n`: 5

| (index) | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Values in `A[]`: | 9 | 5 | 7 | 1 | 3 |

Figure 3. The max heap is built in `A[]` (heap size = 5)

Next, `HeapSort` executes the following 3 steps repeatedly in the loop, while `i ≥ 2`.

Step 1: Exchange the values of `A[1]` and `A[i]`, and `A[i]` becomes the sorted element.

Step 2: Decrement the heap size `i` by 1.

Step 3: Execute `Heapify(A, 1, i)` to set the next maximum value in root `A[1]`.

Figure 4 shows the contents of `i` and `A[]` at some checkpoints in the loop. This shows how the sorting operation proceeds. Here, shaded parts ▨ indicate the sorted elements.

| Checkpoint | Value `i` | Values in `A[]` 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|---|
| start of the loop (as in Figure 3) | 5 | 9 | 5 | 7 | 1 | 3 |
| after step 1 (1st time) | 5 | 3 | 5 | 7 | 1 | 9 |
| after step 3 (1st time) | 4 | E | | | | 9 |
| after step 1 (2nd time) | 4 | 1 | 5 | 3 | 7 | 9 |
| after step 3 (2nd time) | 3 | F | | | 7 | 9 |
| after step 1 (3rd time) | 3 | 3 | 1 | 5 | 7 | 9 |
| after step 3 (3rd time) | 2 | 3 | 1 | 5 | 7 | 9 |
| after step 1 (4th time) | 2 | 1 | 3 | 5 | 7 | 9 |
| after step 3 (4th time) | 1 | 1 | 3 | 5 | 7 | 9 |

Figure 4. Contents of `i` and `A[]` at the checkpoints in the loop

Answer group for C and D

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

Answer group for E

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| a) | 1 | 5 | 7 | 3 | b) | 1 | 7 | 5 | 3 |
| c) | 3 | 1 | 5 | 7 | d) | 3 | 5 | 1 | 7 |
| e) | 7 | 3 | 5 | 1 | f) | 7 | 5 | 3 | 1 |

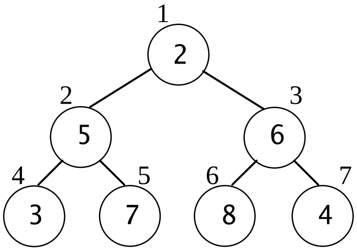
Answer group for F

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

### Subquestion 3

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

Consider another case where the unsorted data shown in Figure 5 is passed to `HeapSort`.



Number of elements `n`: 7

| (index) | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| Values in `A[]`: | 2 | 5 | 6 | 3 | 7 | 8 | 4 |

Figure 5. Unsorted data passed to `HeapSort` (heap size = 7)

`HeapSort` first executes `BuildHeap(A, n)`. When returned from `BuildHeap(A, n)`, the max heap is built in `A[]`, and the values in `A[]` are as follows:

| (index) | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| Values in `A[]`: | G | | | | | | |

Answer group

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| a) | 8 | 5 | 7 | 2 | 3 | 4 | 6 |
| b) | 8 | 5 | 7 | 2 | 3 | 6 | 4 |
| c) | 8 | 7 | 6 | 3 | 5 | 2 | 4 |
| d) | 8 | 7 | 6 | 3 | 5 | 4 | 2 |
| e) | 8 | 7 | 6 | 5 | 3 | 2 | 4 |
| f) | 8 | 7 | 6 | 5 | 3 | 4 | 2 |

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 Subquestion.

This program is a simple text editor which is able to perform editing operations on a line of text.

[Program Description]

(1) The editor first gets a line of text to edit. Then, the editor repeatedly accepts editor commands “`D`” (delete), “`F`” (find), or “`I`” (insert), until it receives “`Q`” (quit) command.

(2) The editor command “`D`” deletes a specified character string from the text, “`F`” finds a specified character string in the text, and “`I`” inserts a specified character string into the text at a specified location.

(3) The following 5 functions are used in the program.

`char * deleted(char *source, int index, int n)`

The function deletes `n` characters starting from `source[index]`, and returns the resulted `source`. If `source` is too short for full deletion, characters from the position `index` to the end of the text are deleted.

`char * do_edit(char *source, char command)`

The function performs the editing operation (delete, find, or insert) according to `command` (“`D`”, “`F`”, or “`I`”), and returns the resulted `source`.

`char get_command(void)`

The function gets a character representing an editor command, and returns the command as the uppercase character. The succeeding input characters are ignored.

`char * insert(char *source, const char *to_insert, int index)`

The function inserts the character string `to_insert` to the position `index`, and returns the resulted `source`. If `source` is too short and the position `index` does not exist, `to_insert` is added after the end of the text.

`int pos(const char *source, const char *to_find)`

The function returns the position of the first occurrence of `to_find` in `source`, or the value `NOT_FOUND` if `to_find` is not found in `source`.

(4) Assuming that the length of any input character string and the length of `source` after editing operations range between 1 and 99. Also, it is assumed to be no errors in input character strings and arguments. The editor does not check for overflows.

(5) The following is a sample run of the text editor program. For each input character string, to show whether it contains space characters or not, an underline ("_") is used instead of a space (" ").

```
Enter the source string:
> Internet_use_is_growing_rapidly.
Enter D(Delete), I(Insert), F(Find), or Q(Quit)> d
String to delete> _growing
New source: Internet_use_is_rapidly.

Enter D(Delete), I(Insert), F(Find), or Q(Quit)> f
String to find> .
'.' found at position 23
New source: Internet_use_is_rapidly.

Enter D(Delete), I(Insert), F(Find), or Q(Quit)> i
String to insert> _expanding
Position of insertion> 23
New source: Internet_use_is_rapidly_expanding.

Enter D(Delete), I(Insert), F(Find), or Q(Quit)> q
String after editing: Internet_use_is_rapidly_expanding.
```

(6) The following library functions are used in the program.

`char * strcat(char *str1, const char *str2)`

Appends the string pointed to by `str2` to the end of the string pointed to by `str1`.

`int strcmp(const char *str1, const char *str2)`

Compares the string pointed to by `str1` with the string pointed to by `str2`.

`char * strcpy(char *s1, const char *s2)`

Copies the string pointed to by `str2` to the string pointed to by `str1`.

`size_t strlen(const char *str)`

Computes the length of the string `str`, excluding the terminating null character.

`char * strncpy(char *s1, const char *s2, size_t n)`

Copies up to `n` characters from the string pointed to by `str2` to the string pointed to by `str1`.

`int toupper(int c)`

If `c` is an alphabetical lowercase character, converts it to the uppercase character.

[Program]

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

#define MAX_LEN 100
#define NOT_FOUND -1

char * deleted(char *source, int index, int n);
char * do_edit(char *source, char command);
char get_command(void);
char * insert(char *source, const char *to_insert, int index);
int pos(const char *source, const char *to_find);

int main(void) {
   char source[MAX_LEN], command;

   source[0] = '\0';
   printf("Enter the source string: \n> ");
   fgets(source, sizeof(source), stdin);
   source[strlen(source)-1] = '\0';
   for (command = get_command();
        command != 'Q'; command = get_command()) {
      do_edit(source, command);
      printf("New source: %s\n\n", source);
   }
   printf("String after editing: %s\n", source);
   return (0);
}

char * deleted(char *source, int index, int n) {
   char rest_str[MAX_LEN];

   if (strlen(source) <= index + n) {
      source[index] = '\0';
   }
   else {
      strcpy(rest_str, &source[index + n]);
      [      A      ];
   }
   return (source);
}
```

```
char * do_edit(char *source, char command) {
   char str[MAX_LEN], buf[4];
   int index;

   switch (command) {
      case 'D':
         printf("String to delete> ");
         fgets(str, sizeof(str), stdin);
         str[strlen(str)-1] = '\0';
         index = pos(source, str);
         if (index == NOT_FOUND)
            printf("'%s' not found\n", str);
         else
            deleted(source, index, strlen(str));
         break;
      case 'I':
         printf("String to insert> ");
         fgets(str, sizeof(str), stdin);
         str[strlen(str)-1] = '\0';
         printf("Position of insertion> ");
         fgets(buf, sizeof(buf), stdin);
         sscanf(buf, "%d", &index);
                B       ;
         break;
      case 'F':
         printf("String to find> ");
         fgets(str, sizeof(str), stdin);
         str[strlen(str)-1] = '\0';
         index =        C       ;
         if (index == NOT_FOUND)
            printf("'%s' not found\n", str);
         else
            printf("'%s' found at position %d\n", str, index);
         break;
      default:
         printf("Invalid edit command '%c' \n", command);
   }
   return (source);
}
```

```
char get_command(void) {
   char command, buf[100];

   printf("Enter D(Delete), I(Insert), F(Find), or Q(Quit)> ");
   fgets(buf, sizeof(buf), stdin);
   buf[strlen(buf)-1] = '\0';
   sscanf(buf, "%c", &command);
   return (toupper(command));
}

char * insert(char *source, const char *to_insert, int index) {
   char rest_str[MAX_LEN];

   if (strlen(source) <= index) {
      strcat(source, to_insert);
   }
   else {
      [      D      ];
      strcpy(&source[index], to_insert);
      strcat(source, rest_str);
   }
   return (source);
}

int pos(const char *source, const char *to_find) {
   int i = 0, find_len, found = 0, position;
   char substring[MAX_LEN];

   [      E      ];
   while (!found && i <= (int)strlen(source) - find_len) {
      strncpy(substring, &source[i], find_len);
      [      F      ];
      if (strcmp(substring, to_find) == 0)
         found = 1;
      else
         ++i;
   }
   if (found)
      position = i;
   else
      position = NOT_FOUND;
   return (position);
}
```

**Subquestion**

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

Answer group for A and D

a) `strcpy(&source[index], rest_str)`
b) `strcpy(rest_str, &source[index])`
c) `strcpy(rest_str, source[index])`
d) `strcpy(source[index], rest_str)`

Answer group for B

a) `insert(source, str, index)`
b) `insert(str, source, index)`
c) `strcpy(source, str)`
d) `strcpy(str, source)`

Answer group for C

a) `pos(*source, *str)`
b) `pos(*str, *source)`
c) `pos(source, str)`
d) `pos(str, source)`

Answer group for E

a) `find_len = (int)strlen(source)`
b) `find_len = (int)strlen(to_find)`
c) `position = (int)strlen(source)`
d) `position = (int)strlen(to_find)`

Answer group for F

a) `source[find_len] = '\0'`
b) `source[find_len] = '\n'`
c) `substring[find_len] = '\0'`
d) `substring[find_len] = '\n'`

**Q8.** Read the following description of Java programs and the programs themselves, and then answer Subquestion.

[Program Description]

Rock-paper-scissors is a hand game played by two (or more) people, where each player simultaneously forms one of three shapes (“rock”, “scissors” and “paper”) with his/her outstretched hand. The rock beats scissors, the scissors beats paper, and the paper beats rock. If both players throw the same shape, the game is tied.

This game is implemented as two-player game; single player vs. computer. Computer chooses the shape randomly. A winner gets one point. After playing three times, the program ends automatically.

The program is composed of the following interface and classes:

| | |
|---|---|
| `GameLoop` | The interface defining the actions to start game, play game, and end game. |
| `Player` | The class describing the player. It saves the name and score of the player. It can sort the players by their scores in descending order. |
| `Game` | The main class describing game playing options. It implements `GameLoop`. It can have as many as players (`List<Player> players`). |
| `GamePlayer` | Inherited from the class `Player`. It holds the player’s chosen shape (`choice`). If the chosen shape is rock, the value of `choice` is 0. If the chosen shape is paper, the value of `choice` is 1. If the chosen shape is scissors, the value of `choice` is 2. |
| `GameLogic` | The class describing the game logic. The method `checkWinner()` determines the player who beats. |
| `RoPaScGame` | Inherited from the class `Game`. It implements the “RockPaperScissors” game. |
| `TestGame` | The class for testing the game. |

The program uses the following java classes from `java.util` and `java.io` packages.

| | |
|---|---|
| `ArrayList` | Resizable-array implementation of the `List` interface. Implements all optional list operations, and permits all elements, including `null`. In addition to implementing the `List` interface, this class provides methods to manipulate the size of the array that is used internally to store the list. |
| `Collections` | The class consists exclusively of static methods that operate on or return collections. It contains polymorphic algorithms that operate on collections, “wrappers”, which return a new collection backed by a specified collection, and a few other odds and ends. |

| | |
|---|---|
| `List` | An ordered collection (also known as a sequence). The user of this interface has precise control over where in the list each element is inserted. The user can access elements by their integer index (position in the list), and search for elements in the list. |
| `BufferedReader` | It reads text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines. The method `readLine()` reads a line of text. |
| `InputStreamReader` | An `InputStreamReader` is a bridge from byte streams to character streams. Each invocation of one of an `InputStreamReader`'s method `read()` may cause one or more bytes to be read from the underlying byte-input stream. To enable the efficient conversion of bytes to characters, more bytes may be read ahead from the underlying stream than are necessary to satisfy the current read operation. For top efficiency, consider wrapping an `InputStreamReader` within a `BufferedReader`. |

When the program is executed, the following result, for example, will be printed out.

```
Input r (rock), p (paper) or s (scissors)
_________________________________________
Input your choice
r
Winner is Player
_________________________
Input your choice
p
Winner is Computer
_________________________
Input your choice
s
Winner is Player
_________________________
Player vs. Computer
Player's score :        2
Computer's score :      1
Player won.
_________________________
```

[Program 1]

```
public interface GameLoop {
   public void startGame();
   public void playGame();
   public void endGame();
}
```

[Program 2]

```
// Object list of this class can be sorted by score.
public class Player implements       A       {
   private String name;
   private int score;
   public Player(String name) {
      setName(name);
      score = 0;
   }
   public void setName(String name) {
      this.name = name;
   }
   public int getScore() {
      return score;
   }
   public void setScore(int score) {
      this.score = score;
   }
   // This is overridden method, that used to sort players by their scores in descending order.
   public int compareTo(Player p) {
      return p.getScore() - this.score;
   }
   public String toString() {
      return name;
   }
}
```

[Program 3]

```
import java.util.ArrayList;
import java.util.List;
      B       class Game implements GameLoop {
   protected List<Player> players;
   public Game() {
      players = new ArrayList<Player>();
   }
   public void play() {
      startGame();
   }
}
```

[Program 4]

```
public class GamePlayer extends Player {
   private int choice;
   public GamePlayer(String name) {
      super(name);
   }
   public void incScore() {
      setScore(getScore() + 1);
   }
   public int getChoice() {
      return choice;
   }
   public void setChoice(int choice) {
      this.choice = choice;
   }
}
```

[Program 5]

```
public class GameLogic {
   // If game is tied, this method return null value,
   // otherwise returns the information about the player who beats.
   [     C     ]  GamePlayer checkWinner(GamePlayer player1,
                                         GamePlayer player2) {
      GamePlayer winner;
      if (player1.getChoice() == player2.getChoice())
         return null;
      else {
         if (player1.getChoice() < player2.getChoice()) {
            if ([     D     ] == 1)
               winner = player2;
            else
               winner = player1;
         }
         else {
            if ([     D     ] == 1)
               winner = player1;
            else
               winner = player2;
         }
         winner.incScore();
         return winner;
      }
   }
}
```

[Program 6]

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.util.Collections;
public class RoPaScGame extends Game {
   // Read data from keyboard input stream.
   public BufferedReader reader;
   private final int PLAYCOUNT = 3;
   public RoPaScGame() {
      super();
      InputStreamReader input = new InputStreamReader(System.in);
      reader = new BufferedReader(input);
   }
   public void initGame() {
      players.clear();
      players.add(new GamePlayer("Computer"));
      players.add(new GamePlayer("Player"));
   }
   public void startGame() {
      initGame();
      System.out.println("Input r (rock), p (paper) or s (scissors)");
      for (int count = 0; count < PLAYCOUNT; count++) {
         playGame();
      }
      endGame();
   }
   public void playGame() {
      GamePlayer computer = (GamePlayer) players.get(0);
      GamePlayer player = (GamePlayer) players.get(1);
      computer.setChoice((int)(Math.random() * 3));
      boolean isCorrect = false;
      System.out.println("_________________________");
      while (!isCorrect) {
         System.out.println("Input your choice");
         char choice = readInput();
         switch (choice) {
            case 'r': player.setChoice(0);
                      isCorrect = true;
                      break;
            case 'p': player.setChoice(1);
                      isCorrect = true;
                      break;
            case 's': player.setChoice(2);
                      isCorrect = true;
                      break;
```

```
              default:  System.out.println("Wrong shape");
                        break;
           }
        }
        GamePlayer winner = GameLogic.checkWinner(player,computer);
        if (winner != null)
           System.out.println("Winner is " + winner);
        else
           System.out.println("The game is tied");
     }
     public void endGame() {
              E      .sort(players);
        Player p1 = players.get(0);
        Player p2 = players.get(1);
        System.out.println("_________________________");
        System.out.println(p1 + " vs. " + p2);
        System.out.println(p1 + "'s score :\t" + p1.getScore());
        System.out.println(p2 + "'s score :\t" + p2.getScore());
        if (p1.getScore() != p2.getScore())
           System.out.println(p1 + " won.");
        else
           System.out.println("End in a tie.");
        System.out.println("_________________________");
     }
     public char readInput() {
        String result = null;
        try {
           result = reader.readLine();
        }
        catch (Exception e) {
        }
        return result.charAt(0);
     }
  }
```

[Program 7]

```
  public class TestGame {
     public static void main(String[] args) {
        Game game = new RoPaScGame();
        game.play();
     }
  }
```

## Subquestion

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

Answer group for A and E

- a) `ArrayList`
- b) `Collection<Player>`
- c) `Collections`
- d) `Comparable<Player>`
- e) `Comparables`
- f) `Comparator`
- g) `Comparator<Player>`
- h) `List`

Answer group for B and C

- a) `private`
- b) `private abstract`
- c) `private final`
- d) `private static`
- e) `public`
- f) `public abstract`
- g) `public final`
- h) `public static`

Answer group for D

- a) `Math.abs(player1.getChoice() - player2.getChoice())`
- b) `Math.min(player1.getChoice(), player2.getChoice())`
- c) `player1.getChoice()`
- d) `player1.getChoice() - player2.getChoice()`
- e) `player2.getChoice()`
- f) `Player2.getChoice() - player1.getChoice()`