

April 2013

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 **(S)** 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 spring Fundamental IT Engineer Examination conducted?

Answer group

- a) March b) April c) May d) June

Since the correct answer is “b) April”, mark your answer sheet as follows:

[Sample Answer]

Sample	<input type="radio"/> a	<input checked="" type="radio"/> b	<input type="radio"/> c	<input type="radio"/> d	<input type="radio"/> e	<input type="radio"/> f	<input type="radio"/> g	<input type="radio"/> h	<input type="radio"/> i	<input type="radio"/> j
--------	-------------------------	------------------------------------	-------------------------	-------------------------	-------------------------	-------------------------	-------------------------	-------------------------	-------------------------	-------------------------






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.
		Indicates a one-way selection process. If the conditional expression is true, then the process is executed.
		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.
		Indicates a pre-test iteration process. While the conditional expression is true, the process is executed repeatedly.
		Indicates a post-test iteration process. The process is executed, and then while the conditional expression is true, the process is executed repeatedly.
		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	<div> <div>High</div> <div>↑</div> <div>↓</div> <div>Low</div> </div>
Multiplication, division	×, ÷, %	
Addition, subtraction	+, −	
Relational operation	>, <, ≥, ≤, =, ≠	
Logical product	and	
Logical sum	or	

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 logic operations, and then answer Subquestions 1 and 2.

[Description of “On - Off” logic game]

Three persons named X, Y, and Z participate the game called “On - Off”. Each person has his/her own electronic switch, and everyone will turn ON or turn OFF the switch all at once when a commander gives a start command. They will win the game if one of the following three cases (1) through (3) is true. Otherwise, they will lose the game.

- (1) Person X turns ON, Person Y and Person Z turn OFF
- (2) Person X and Person Z turn ON, Person Y turns OFF
- (3) Person Y turns ON, Person X and Person Z turn OFF

Table 1 shows the truth table for the rule described above.

Table 1 Truth table

Inputs			Output
X	Y	Z	Win / Lose
0 (OFF)	0 (OFF)	0 (OFF)	0 (Lose)
0 (OFF)	0 (OFF)	1 (ON)	0 (Lose)
0 (OFF)	1 (ON)	0 (OFF)	1 (Win)
0 (OFF)	1 (ON)	1 (ON)	0 (Lose)
1 (ON)	0 (OFF)	0 (OFF)	1 (Win)
1 (ON)	0 (OFF)	1 (ON)	1 (Win)
1 (ON)	1 (ON)	0 (OFF)	0 (Lose)
1 (ON)	1 (ON)	1 (ON)	0 (Lose)

Subquestion 1

From the answer group below, select the correct logic expression which will output the correct Win / Lose value from three inputs X, Y and Z.

Answer group

- a) $X \text{ AND NOT}(Y) + \text{NOT}(X) \text{ AND } Y \text{ AND NOT}(Z)$
- b) $X \text{ AND NOT}(Y) \text{ AND } Z + \text{NOT}(X) \text{ AND NOT}(Z)$
- c) $X \text{ AND NOT}(Y) \text{ AND } Z + X \text{ AND } Y$
- d) $X \text{ AND } Y \text{ AND NOT}(Z) + \text{NOT}(X) \text{ AND } Z$

Subquestion 2

From the answer group below, select the correct answer to be inserted into each blank in Figure 1. If necessary, select the same answer twice or more.

Assuming that the given rule is inversed (i.e. Win is changed to Lose, and Lose is changed to Win). Figure 1 shows the logic circuit which will output the correct inversed Win / Lose value from three inputs X, Y, and Z.

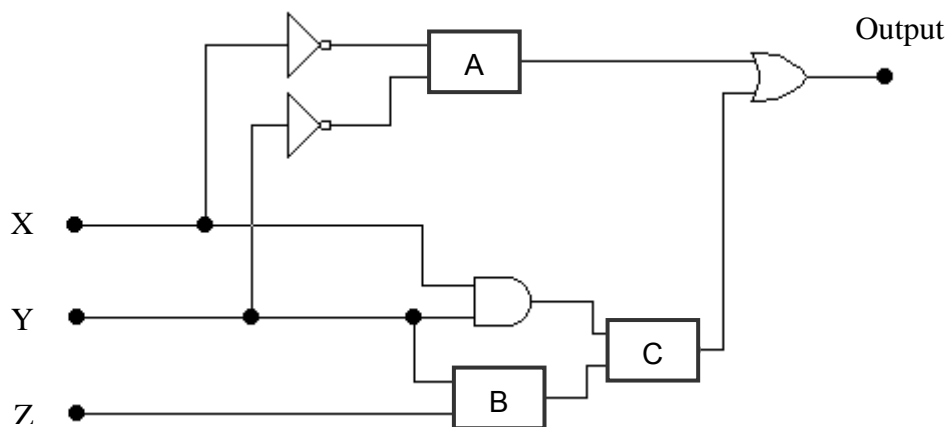





Figure 1 Logic circuit which will output the correct inversed Win / Lose value

Answer group

a)  (AND gate)

b)  (OR gate)

c)  (NOT gate)

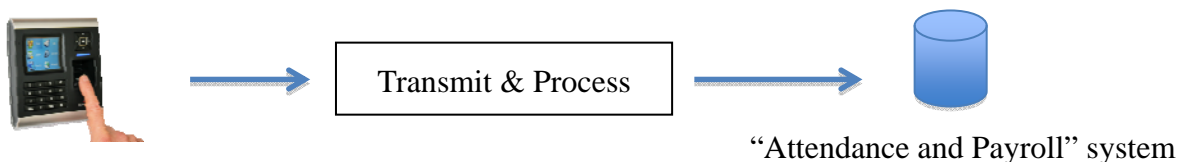
Q2. Read the following description concerning a database for overtime management, and then answer Subquestions 1 and 2.

ES Trading Company uses fingerprint time recorder to manage employees' daily attendance. Every employee has to press his/her finger to the fingerprint reader when he/she comes to the office in the morning and when he/she leaves the office in the evening to record his/her in-time and out-time. Fingerprints of all employees have been registered to the fingerprint reader together with their Employee IDs. When an employee presses his/her registered finger to the fingerprint reader, it records the date, time and employee ID.

ES Trading Company also uses "Attendance and Payroll" system. The fingerprint reader is connected to the system via an internal LAN. When the system receives attendance data from the fingerprint reader, the system saves that data into DailyAttendance table, performs some necessary operations, and inserts the processed data into AttendanceDetail table.

The process includes –

- 1) Checking in-time and out-time of each employee for each day.
- 2) Calculating OT (overtime) hours when daily working hours exceed 8 hours.
Overtime is rounded off into 30 minutes intervals, e.g. 0.5, 1.0, 1.5, 2.0
- 3) Categorizing OT (1 for OT in weekday, 2 for OT in holiday).



Normal working hours of ES Trading Company are 8 hours (from 9:00 to 17:00). According to the working nature of ES Trading Company, employees may need to work after the business hours, i.e. overtime working occurs only in the evening.

DailyAttendance table contains the following data received from the fingerprint reader.

<u>Column name</u>	<u>Type</u>	<u>Description</u>
TranID	numeric	Transaction ID
AttdDate	Date	Date attended
RecordTime	Time	Time recorded
EmpID	numeric	Employee ID

AttendanceDetail table contains the following data processed for payroll calculations.

<u>Column name</u>	<u>Type</u>	<u>Description</u>
AttdID	numeric	Attendance ID
AttdDate	Date	Date attended
EmpID	numeric	Employee ID
InTime	Time	Time employee comes to the office
OutTime	Time	Time employee leaves the office
OTHour	numeric	Calculated overtime hours, 0 if no overtime
OTType	numeric	1 for weekday, 2 for holiday

The system also has other database tables. For ease of use in SQL statements, the system also has EmployeeDetail view that is created based on employee, department and post table for payroll calculations.

EmployeeDetail view contains the following data.

<u>Column name</u>	<u>Type</u>	<u>Description</u>
EmpID	numeric	Employee ID
EmpName	text	Employee name
DeptID	numeric	Department ID
DeptName	text	Department name
BasicPay	numeric	Monthly basic pay
HourRate	numeric	Normal pay per hour (calculated from basic pay)

Subquestion 1

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

When an employee works on holidays, the whole working hours are considered as overtime. The overtime pay for holidays is calculated as follows:

$$\text{Overtime pay} = \text{Normal pay per hour} \times \text{Overtime hours} \times 2$$

For example, when an employee, whose normal pay per hour is 10 dollars, works 6 hours on Sunday, the overtime pay is $6(\text{hours}) \times 10(\text{dollars}) \times 2.0$, i.e. 120 dollars.

The following report shows (A) the total of monthly basic pay, (B) the total of overtime pay for holidays, and the ratio of (B)/(A), summarized by each department for the month of March, 2013.

<u>Department Name</u>	<u>Total Basic Pay</u>	<u>Total Holiday OT Pay</u>	<u>Ratio</u>
Admin Office	800,000	50,000	6.25
Sales	1,200,000	350,000	29.16
Service	1,150,000	400,000	34.78

The report shown above is created by the following SQL statement.

```

SELECT DeptName, SUM(XXX) AS TotalBasicPay,
              SUM(YYY) AS TotalHolidayOTPay,
              100 * (SUM(YYY)/ SUM(XXX)) AS Ratio
FROM ( SELECT e.DeptName,  AS XXX, 0 AS YYY
      FROM EmployeeDetail e
      GROUP BY e.DeptName

      UNION ALL

      SELECT e.DeptName, 0 AS XXX,  AS YYY
      FROM AttendanceDetail a JOIN EmployeeDetail e
      ON a.EmpID= e.EmpID
      WHERE a.OTType = 2 AND
            a.AttdDate BETWEEN '2013-03-01' and '2013-03-31'
      GROUP BY e.DeptName
    ) T
GROUP BY DeptName ORDER BY DeptName

```

Answer group

- | | |
|-----------------------------------|------------------------------|
| a) a.OTHour | b) e.HourRate * a.OTHour * 2 |
| c) e.BasicPay | d) SUM(a.OTHour) |
| e) SUM(e.HourRate * a.OTHour * 2) | f) SUM(e.BasicPay) |

Subquestion 2

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

Overtime working on weekdays occurs only in the evening. Concretely, overtime working starts at 18:00 after the dinner break. The overtime pay per hour for weekdays is calculated as follows:

From 18:00 to 20:00: Overtime pay = Normal pay per hour × Overtime hours

From 20:00 to 22:00: Overtime pay = Normal pay per hour × Overtime hours × 1.3

From 22:00 onwards: Overtime pay = Normal pay per hour × Overtime hours × 1.5

For example, an employee, whose normal pay per hour is 10 dollars, works 4.5 hours after the business hours (i.e. from 18:00 to 22:30), the overtime pay is calculated as:

$$10 \times 2.0 + 10 \times 2.0 \times 1.3 + 10 \times 0.5 \times 1.5 = 53.5(\text{dollars})$$

The following report shows the total overtime pay on weekdays by each employee who worked overtime for the month of March, 2013.

<u>Emp ID</u>	<u>Emp Name</u>	<u>Dept ID</u>	<u>Dept Name</u>	<u>Total weekday OT Pay</u>
003	Ko Ko	20	Sales	300
006	Min Min	30	Service	1,200

The report shown above is created by the following SQL statement.

```

SELECT a.EmpID, e.EmpName, e.DeptID, e.DeptName,
       SUM( 
           WHEN a.OTHour > 4
             THEN e.HourRate * (  )
           WHEN 
             THEN e.HourRate * (  )
           ELSE e.HourRate * a.OTHour
           END
       ) as TotalWeekdayOTPay
FROM AttendanceDetail a JOIN EmployeeDetail e
  ON a.EmpID = e.EmpID
WHERE a.OTType = 1 AND a.OTHour > 0 AND
      a.AttdDate BETWEEN '2013-03-01' and '2013-03-31'
GROUP BY a.EmpID, e.EmpName, e.DeptID, e.DeptName ORDER BY a.EmpID

```

Answer group for C

- a) case b) insert c) select d) set

Answer group for D and F

- a) $2 + (a.OTHour - 2) * 1.3$
 b) $2 + 2 * 1.3 + (a.OTHour - 4) * 1.5$
 c) $a.OTHour * 1.3$
 d) $a.OTHour * 1.3 - (a.OTHour - 2)$
 e) $a.OTHour * 1.5$
 f) $a.OTHour * 1.5 - (a.OTHour - 2) * 1.3 - (a.OTHour - 4)$

Answer group for E

- a) $a.OTHour \leq 4$ b) $a.OTHour = 3 \text{ OR } a.OTHour = 4$
 c) $a.OTHour > 2 \text{ AND } a.OTHour \leq 4$ d) $a.OTHour \text{ BETWEEN } 3 \text{ AND } 4$

Q3. Read the following description concerning a communication protocol, and then answer Subquestion.

In communication links and computer networks, ARQ (automatic repeat request) protocols are commonly used to provide error controls. They allow for detection of errors in a link or network (e.g. corrupted packets, lost packets), and subsequent correction of the errors so that a receiver eventually receives the correct data. The mechanisms used by ARQ protocols include:

- o Positive acknowledgement (ACK): a receiver returns ACK after successfully receiving an error-free packet.
- o Retransmission after timeout: a sender retransmits the data packet that has not been ACKed within a predetermined timeout interval.
- o Negative acknowledgement (NACK): a receiver returns NACK when an error is detected.

The stop-and-wait ARQ protocol uses positive ACKs and retransmissions. It can be described as follows:

- o When a sender has a data packet to send, and it has received an ACK for the previous data packet, the sender transmits the data packet. The header in each data packet contains a 1-bit sequence number: value 0 for the first packet, value 1 for the second packet, value 0 for the third packet, and so on.
- o After the data packet transmission is complete, the sender then starts a timer.
- o When the receiver receives the data packet, if the packet is error-free, then the receiver replies with an ACK. The ACK contains a 1-bit sequence number: the opposite value of the sequence number of the just received data packet.
- o If the receiver receives a data packet containing errors, the receiver discards the packet and does not send an ACK.
- o If the receiver receives a data packet with a sequence number the same as the previous received data packet, the receiver discards the packet and re-sends the previous ACK.
- o If the sender receives an ACK, then the sender may proceed to transmit the next data packet (when it has data ready to send).
- o If the sender does not receive an ACK in its entirety before the timer expires (i.e. a timeout occurs), then the sender retransmits the previous sent data packet.

Figure 1 illustrates an example of the protocol operation, showing a case of successful data delivery, as well as a lost data packet. In Figure 1, A is a sender and B is a receiver.

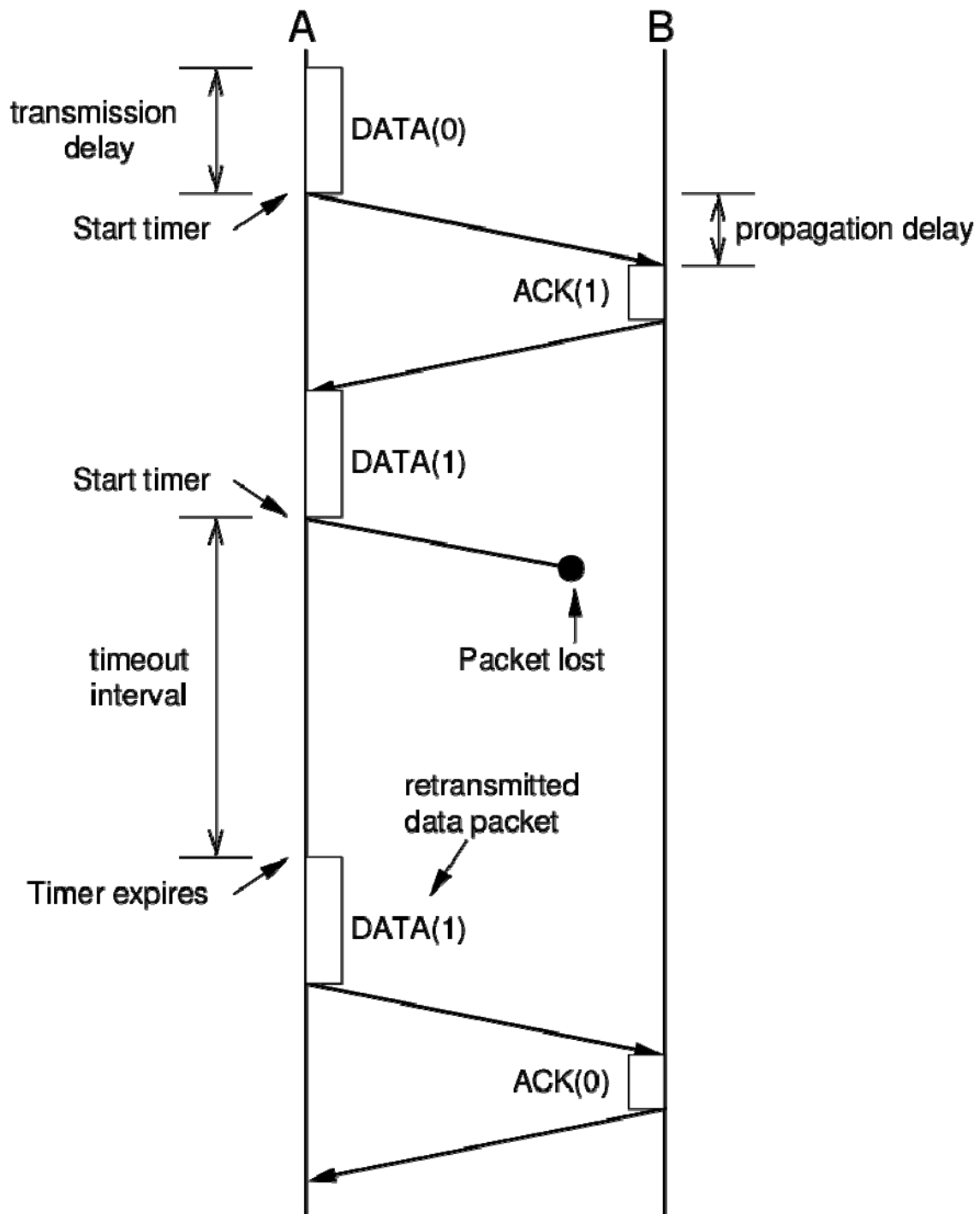


Figure 1 Example of protocol operation

Subquestion

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

The specifications of communication illustrated in Figure 1 are as follows:

- o Propagation delay across the link in one direction is 15 milliseconds. Propagation delay in the opposite direction is also 15 milliseconds. The propagation delay is fixed.
- o The sender uses a timeout interval of 60 milliseconds.
- o The link data rate is $1\text{M}(=10^6)$ bits per second. It is the same in both directions.
- o Each data packet is fixed in size. It contains 125 bytes of header and 1,125 bytes of user data.
- o Each ACK is fixed in size. It is 125 bytes long.
- o The processing delays within computers are negligible, i.e. 0 seconds.

- (1) The transmission delay (i.e. the time to transmit an entire packet on to a link) of a single data packet (including header) is milliseconds.
- (2) Starting from when computer A starts the transmission of a data packet to computer B, if there are no errors, it takes milliseconds until computer A completely receives an ACK from computer B for that data packet.
- (3) When computer A transmits a data packet to computer B, the data packet is lost before reaching computer B. Starting from when computer A starts the transmission of the original data packet to computer B, if there are no other errors, it takes milliseconds until computer A completely receives an ACK from computer B for that data packet.
- (4) In the specifications of communication listed above, the timeout interval is set to 60 milliseconds. The timeout interval can be changed to another value, and when it is changed, the new timeout interval must be milliseconds. Assuming that the other values in the specifications remain unchanged, and the timeout interval can be specified in multiple of 5 milliseconds (i.e. 5, 10, 15, ... milliseconds).

Answer group for A

- | | | |
|---------|---------|--------|
| a) 1 | b) 1.25 | c) 10 |
| d) 12.5 | e) 100 | f) 125 |

Answer group for B

- | | | |
|-----------|----------|----------|
| a) 16.375 | b) 28.75 | c) 31.1 |
| d) 41 | e) 140 | f) 152.5 |

Answer group for C

- | | | |
|---------|----------|--------|
| a) 92.1 | b) 107.1 | c) 111 |
| d) 126 | e) 300 | f) 315 |

Answer group for D

- | | |
|--------------------------------|--------------------------------|
| a) greater than or equal to 20 | b) greater than or equal to 30 |
| c) greater than or equal to 35 | d) less than or equal to 115 |

Q4. Read the following description concerning encryption and decryption, and then answer Subquestions 1 and 2.

Asymmetric cryptography uses an asymmetric key pair, i.e. public key and private key. The following RSA algorithm is used for generation of key pairs. Public key is used to encrypt the message, and private key is used to decrypt the message. Messages encrypted with public key can only be decrypted using its private key.

The following steps show the key generation algorithm.

- (1) Select two large random prime numbers, p and q .
- (2) Compute the value n , by $n = p \times q$.
- (3) Compute the value ϕ , by $\phi = (p - 1)(q - 1)$.
- (4) Choose an integer e , such that $1 < e < \phi$ and that the greatest common divisor of e and ϕ is 1.
- (5) Compute the secret exponent d , such that $d \times e = 1 \bmod \phi$, i.e. the remainder of $(d \times e)$ divide by ϕ to be 1.
- (6) Thus, public key is obtained as the set of values (n, e) , and private key is obtained as the set of values (n, d) .

The following steps show the encryption algorithm when a sender sends a message to a receiver.

- (1) Provide the plain message M to be sent. Here, M is a positive integer.
- (2) Obtain the receiver's public key (n, e) .
- (3) Compute the cipher message C , by $C = M^e \bmod n$.
- (4) Send the cipher message C to the receiver.

The following steps show the decryption algorithm when a receiver receives a message from a sender.

- (1) Provide the cipher message C that was received.
- (2) Provide the private key (n, d) .
- (3) Compute the plain message M , by $M = C^d \bmod n$.

Subquestion 1

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

- (1) Ms. Aye Aye is going to send a message to Mr. Thida, using these algorithms. The selected two random prime numbers p and q for the key generation algorithm are 11 and 3 respectively. Then, the possible public key and private key generated are A respectively.
- (2) The selected two random prime numbers p and q for the key generation algorithm are 11 and 3 respectively. Then, the number of possible key pairs that can be generated by the key generation algorithm is B , with the condition that the values of both e and d are less than 10.
- (3) The value of plain message M is 7, the receiver's public key is (15, 3), and the receiver's private key is (15, 11). Then, the value of cipher message C created by the encryption algorithm is C .

Answer group for A

- | | |
|------------------------|------------------------|
| a) (33, 3) and (33, 5) | b) (33, 3) and (33, 7) |
| c) (33, 7) and (33, 7) | d) (33, 7) and (33, 9) |

Answer group for B

- | | | | |
|------|------|------|------|
| a) 1 | b) 2 | c) 3 | d) 4 |
|------|------|------|------|

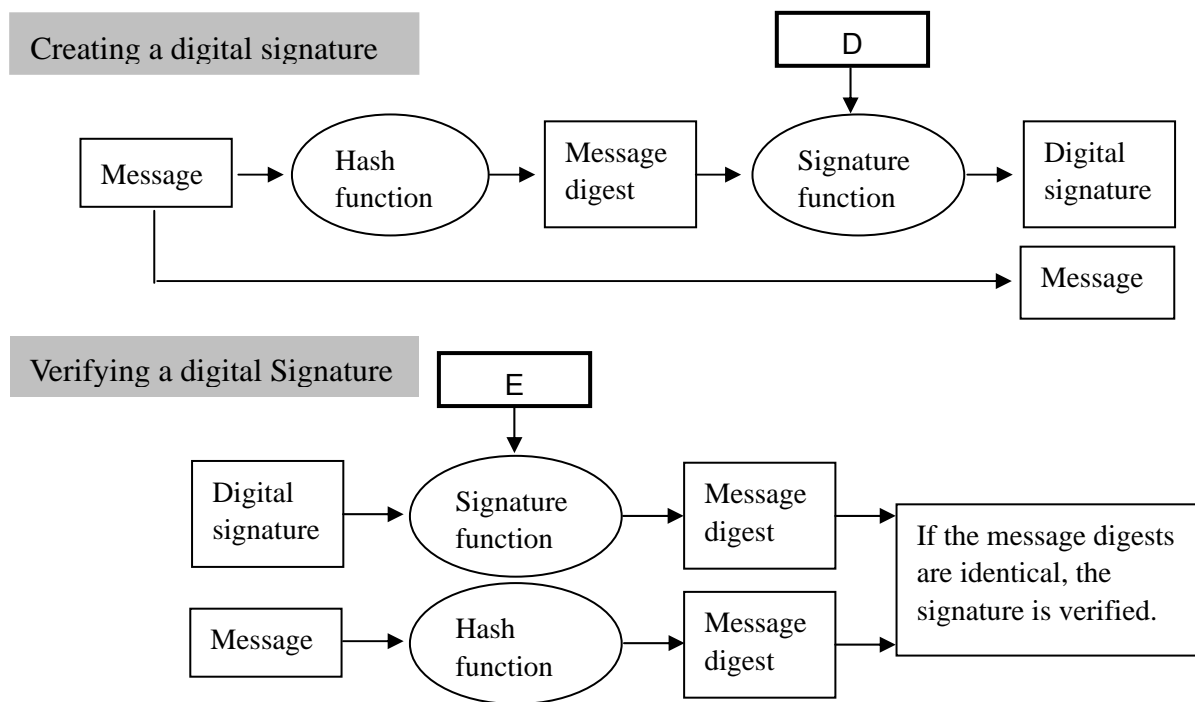
Answer group for C

- | | | | |
|------|-------|-------|-------|
| a) 7 | b) 13 | c) 15 | d) 22 |
|------|-------|-------|-------|

Subquestion 2

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

The following figure shows a general flow where a sender sends a message with a digital signature to a receiver. As shown in the figure, two keys D and E are needed to create and verify the digital signature.



Answer group

- | | |
|---------------------------|--------------------------|
| a) Receiver's private key | b) Receiver's public key |
| c) Sender's private key | d) Sender's public key |

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

[Program Description]

The customs department will accept a customs declaration form from an agent when the agent imports the goods to the country. A customs declaration form may contain one or more items. Every item has a unique item code. The agent has to mention in the customs declaration form that it has the license to import the goods.

When evaluating an Assessable Value (hereinafter AV) for each item, if the item does not have previous registered unit price to assess a Customs Duty (hereinafter CD), the customs department will collect some money as a Revenue Deposit (hereinafter RD) in advance from the agent. RD is calculated as 20% of AV.

On the customs declaration form, for each item, the agent inputs the item code, preference type that the agent wants to use when assessing CD rate, commodity type, quantity, and unit price. The customs department collects CD, Commercial Tax (hereinafter CT), and RD (if applicable) from the agent for each item on the customs declaration form. The values of AV, CD, CT are calculated as follows:

Assessable Value (AV) = unit price \times quantity

Customs Duty (CD) = AV \times CD rate

Commercial Tax (CT) = (AV + CD) \times CT rate

Total payment = CD + CT + RD

For each item, the customs department compares the declared unit price given by the agent with the registered unit price in Registered Price File. If the declared unit price is less than the registered unit price, the registered unit price becomes the declared unit price to calculate the AV of that item.

The customs department uses the following files to calculate the payment for each customs declaration form that an agent declares.

CD Rate File: Indexed file (see Table 1).

The primary and secondary keys are item code and preference type. This file records the CD rate by item code and preference type.

When the program executes the following statement,

- `Get CDRateFile(CDRate) Key(ItemCode, PrefType) Set(Exist)`

CD Rate File is accessed with item code and preference type as the keys. If the record exists, variable `Exist` is set to `true` and the CD rate is loaded into variable `CDRate`, otherwise, variable `Exist` is set to `false` and no value is loaded into variable `CDRate`.

CT Rate File: Indexed file (see Table 2).

The primary key is commodity type. This file records the CT rate by commodity type.

When the program executes the following statement,

- `Get CTRateFile(CTRate) Key(ComType) Set(Exist)`

CT Rate File is accessed with commodity type as the key. If the record exists, variable `Exist` is set to true and the CT rate is loaded into variable `CTRate`, otherwise, variable `Exist` is set to false and no value is loaded into variable `CTRate`.

Registered Price File: Indexed file (see Table 3).

The primary key is item code. This file records the registered unit price by item code.

When the program executes the following statement,

- `Get RegisteredPriceFile(RegPrice) Key(ItemCode) Set(Exist)`

Registered Price File is accessed with item code as the key. If the record exists, variable `Exist` is set to true and the registered unit price is loaded into variable `RegPrice`, otherwise, variable `Exist` is set to false and no value is loaded into variable `RegPrice`.

Table 1 CD Rate File

Item code	Preference type	Description	CD rate
1001	1	Falsermal	0.10
1001	2	ASEAN+China	0.08
1001	3	ASEAN+Japan	0.07
1001	4	ASEAN+India	0.05
1002	1	Falsermal	0.20
1002	2	ASEAN+China	0.18
1002	3	ASEAN+Japan	0.10
1002	4	ASEAN+India	0.10
...

Table 2 CT Rate File

Commodity type	Description	CT rate
1	Farm products	0.08
2	Motor Vehicle	0.10
3	Cosmetics	0.20
4	Medicine	0.05
5	Stationary	0.05
6	Furniture	0.15
7	Food	0.10
8	Beverage	0.10
...

Table 3 Registered Price File

Item code	Registered unit price	Date
1001	5,000	2013-01-05
1002	7,500	2013-03-28
...

Customs Declaration Form: Sequential file (see Table 4).

The file contains one or more items to be declared. A record contains the item code, preference type, commodity type, quantity, and unit price of the item.

When the program executes the following statement,

- `Get InputData(ItemCode, PrefType, ComType, Quantity, UnitPrice)`
`EOF(Endfile)`

the next record is read from Customs Declaration Form, and item code, preference type, commodity type, quantity, and unit price are loaded into variables `ItemCode`, `PrefType`, `ComType`, `Quantity`, and `UnitPrice`, respectively. When end-of-file is reached, variable `Endfile` is set to true and no value is loaded into the variables.

Table 4 Customs Declaration Form

Item code	Preference type	Commodity type	Quantity	Unit price
1001	1	1	100	5,000
1001	5	1	50	5,000
1002	3	4	100	7,500

When the program is executed using the sample data shown in Tables 1 through 4, the following output will be obtained.

Item	Pref	Com	Quantity	UnitPrice	AV	CD	CT	RD	Message
1001	1	1	100	5000	500000	50000	44000	0	OK
1001	5	1	50	5000	250000	0	20000	0	ERROR
1002	3	4	100	7000	750000	75000	41250	0	OK
TotalPay			230250			125000	105250	0	

[Program]

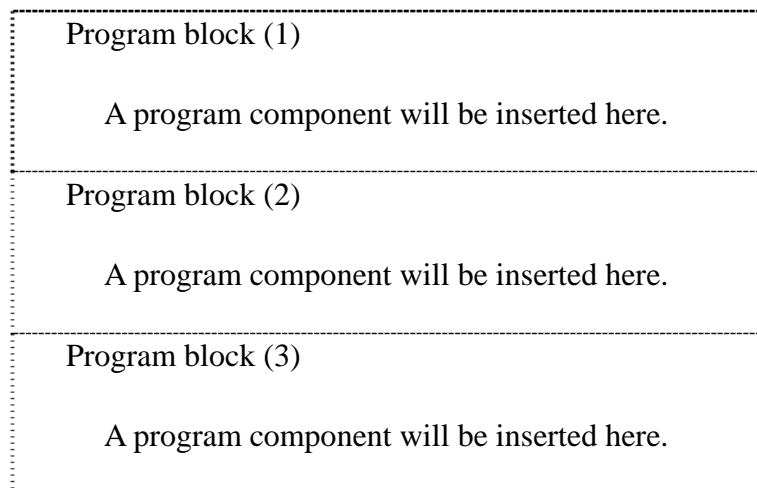
- Program: CustomsDeclaration
- Integer: ItemCode, PrefType, ComType, Quantity, UnitPrice
- Integer: CDRate, CTRate, RegPrice
- Integer: AV, CD, CT, RD, TotalPay, TotalCD, TotalCT, TotalRD
- Logical: Endfile, Error, Exist
- Character: Message

- TotalCD \leftarrow 0
- TotalCT \leftarrow 0
- TotalRD \leftarrow 0
- Print("Item Pref Com Quantity UnitPrice",
" AV CD CT RD Message")
- Endfile \leftarrow false

C

■ Endfile = false

D



▲ Error = false

- Message \leftarrow "OK"

- Message \leftarrow "ERROR"

- Print(ItemCode, PrefType, ComType, Quantity, UnitPrice,
AV, CD, CT, RD, Message)
- TotalCD \leftarrow TotalCD + CD
- TotalCT \leftarrow TotalCT + CT
- TotalRD \leftarrow TotalRD + RD

E

- TotalPay \leftarrow TotalCD + TotalCT + TotalRD
- Print(" TotalPay", TotalPay, TotalCD, TotalCT, TotalRD)
- /* End of program */

Each one of the following three program components will be inserted into either of the program blocks (1), (2), or (3) in the program.

[Program component “Process CD Rate”]

- Get CDRateFile(CDRate) Key(ItemCode, PrefType) Set(Exist)
- ▲

↑

↓

▼

Exist = false

• CDRate ← 0

• Error ← true
- CD ← AV × CDRate

[Program component “Process CT Rate”]

- Get CTRateFile(CTRate) Key(ComType) Set(Exist)
- ▲

↑

↓

▼

Exist = false

• CTRate ← 0

• Error ← true
- CT ← (AV + CD) × CTRate

[Program component “Process Registered Price”]

- Get RegisteredPriceFile(RegPrice) Key(ItemCode) Set(Exist)
- ▲

↑

↓

▼

F

• UnitPrice ← RegPrice
- AV ← UnitPrice × Quantity
- ▲

↑

↓

▼

Exist = true

• RD ← 0

• RD ← 0.20 × AV

Subquestion 1

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

Each one of the three program components, “Process CD Rate”, “Process CT Rate”, and “Process Registered Price”, is to be inserted into either of the program blocks (1), (2), or (3) in the program, whichever is appropriate. Specifically, Program component “Process CD Rate” must be inserted into A, and Program component “Process Registered Price” must be inserted into B.

Answer group

- a) either of Program block (1) or Program block (2)
- b) either of Program block (1) or Program block (3)
- c) either of Program block (2) or Program block (3)
- d) Program block (1)
- e) Program block (2)
- f) Program block (3)

Subquestion 2

From the answer groups below, select the correct answer to be inserted into each blank in the above program and the program component. If necessary, select the same answer twice or more.

Answer group for C through E

- a) • `CD ← 0`
 - `CT ← 0`
 - `RD ← 0`
- b) • `Endfile ← false`
- c) • `Error ← false`
- d) • `Get InputData(ItemCode, PrefType, ComType, Quantity, UnitPrice)`
• `EOF(Endfile)`

Answer group for F

- a) `(Error = false) and (UnitPrice < RegPrice)`
- b) `(Error = false) and (UnitPrice > RegPrice)`
- c) `(Exist = true) and (UnitPrice < RegPrice)`
- d) `(Exist = true) and (UnitPrice > RegPrice)`

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

[Program Description]

`SumInTriangle` is a subprogram that calculates and outputs the maximum sum of the numbers on the path from top to anywhere on bottom for a given triangle of numbers.

There are three conditions that must be followed:

- o For each moving from top to bottom, go down diagonally on the left hand side or the right hand side.
- o The number of rows in the triangle is greater than 1 and less than or equal to 100.
- o Every number in the triangle is an integer and ranges from 0 to 99.

Figure 1 shows an example of a triangle of numbers. In Figure 1, “7 – 8 – 1 – 7 – 2” is a valid path from top to bottom.

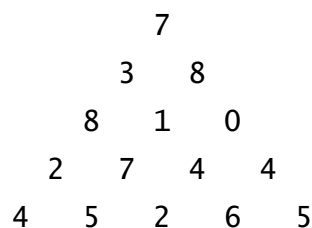


Figure 1 Example of a triangle of numbers

- (1) The numbers in the triangle have already stored properly in global 2-dimensional integer array `Triangle[][]`. Indexes of rows and columns of this array range from 1 to 100. The number at the top of the triangle is stored in `Triangle[1][1]`, and the numbers on the r -th row are stored in `Triangle[r][1]`, `Triangle[r][2]`, ... , `Triangle[r][r]`.
- (2) The number of rows in the triangle has already stored properly in global integer variable `Height`.
- (3) `SumInTriangle` uses subprogram `calculatePaths` to calculate the maximum value of the path in the triangle. Each maximum value of the path is stored in global 2-dimensional integer array `SumPaths[][]`. The maximum value of the path from top of the triangle to the position (row r , column c) is stored in `SumPaths[r][c]`. All values in `SumPaths[][]` can be calculated as follows:
 - a) First, store the value `Triangle[1][1]` into `SumPaths[1][1]`.
 - b) If $c = 1$ (leftmost column on the row) or $c = r$ (rightmost column on the row), then there is only one path to go to position (r, c) . So, if $c = 1$, calculate the value `Triangle[r][c] + SumPaths[r-1][c]`, and store that value into `SumPaths[r][c]`.

- c) If $1 < c < r$ (intermediate column on the row), then there are two paths to go to position (r, c) , i.e. from positions $(r-1, c-1)$ and $(r-1, c)$. So, calculate the value $\text{Triangle}[r][c] + \max(\text{SumPaths}[r-1][c-1], \text{SumPaths}[r-1][c])$, and store that value into $\text{SumPaths}[r][c]$.
- (4) Based on the calculated sum of paths stored in array $\text{SumPaths}[][]$, the numbers on the maximum path can be identified. Starting at the position of the maximum value, all numbers on the maximum path can be resolved from bottom to top using the subprogram `DisplayMaxPath`. This subprogram stores these all numbers in global 1-dimensional integer array `MaxPath`.
- (5) The function $\max(x, y)$ returns the maximum value of x and y .

[Program]

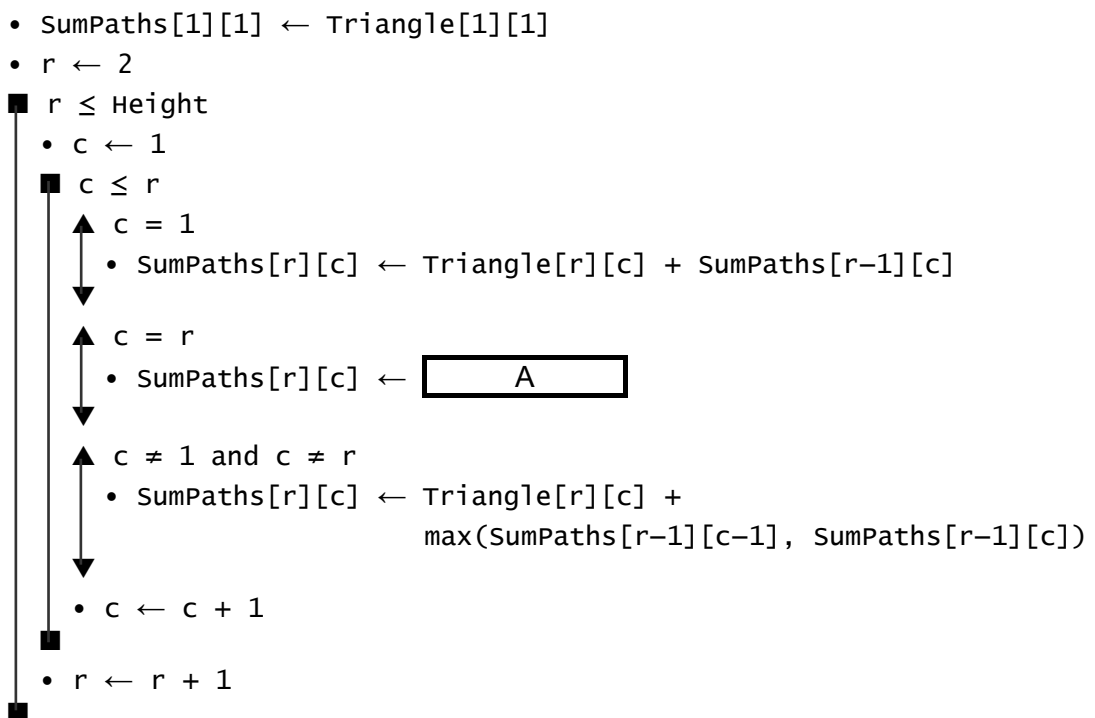
- Global: Integer: Height, MaxPath[100]
- Global: Integer: SumPaths[100][100]
- Global: Integer: Triangle[100][100]

○ Program: SumInTriangle()

- calculatePaths()
- displayMaxPath()

○ Program: calculatePaths()

○ Integer: c, r



- Program: displayMaxPath()
- Integer: c, m, maxval, r

```

/* Find the maximum value of paths in triangle */
• maxval ← 0
• c ← 1
■ c ≤ Height
  ↑ B
  • m ← c
  • maxval ← SumPaths[Height][m]
  • MaxPath[Height] ← Triangle[Height][m]
  ↓
  • c ← c + 1
■
/* Resolve maximum path based on SumPaths */
• r ← Height - 1
■ r > 1
  ↑ m > 1 and MaxPath[r+1] = (SumPaths[r+1][m] - SumPaths[r][m-1])
  • m ← C
  ↓
  • MaxPath[r] ← Triangle[r][m]
  • r ← r - 1
■
• D
/* Display the maximum path and the sum value */
• r ← 1
■ r ≤ Height
  • print(MaxPath[r]) /* Print the value */
  • r ← r + 1
■
• print(maxval)

/* End of program */

```

Subquestion 1

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

Answer group for A

- a) $\text{Triangle}[r][c] + \text{SumPaths}[r-1][c-1]$
- b) $\text{Triangle}[r][c] + \text{SumPaths}[r-1][c+1]$
- c) $\text{Triangle}[r][c] + \text{SumPaths}[r+1][c-1]$
- d) $\text{Triangle}[r][c] + \text{SumPaths}[r+1][c+1]$

Answer group for B

- a) `SumPaths[c][Height] ≤ maxval` b) `SumPaths[c][Height] > maxval`
c) `SumPaths[Height][c] ≤ maxval` d) `SumPaths[Height][c] > maxval`

Answer group for C

- a) `m - 1` b) `m + 1`
c) `r` d) `r + 1`

Answer group for D

- a) `MaxPath[1] ← Height` b) `MaxPath[1] ← Triangle[1][1]`
c) `MaxPath[r] ← Triangle[r][m-1]` d) `MaxPath[r] ← Triangle[r][m+1]`

Subquestion 2

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

Figure 2 shows another example of a triangle of numbers. Regarding Figure 2, the maximum sum of the numbers on the paths calculated using `SumInTriangle` subprogram is .

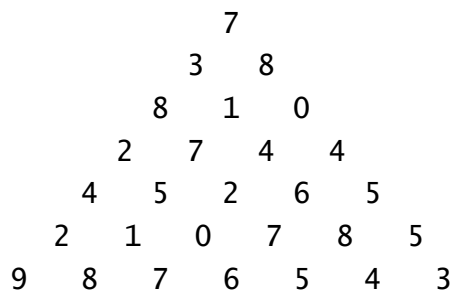


Figure 2 Another example of a triangle of numbers

Answer group

- a) 38 b) 39 c) 40 d) 41

Concerning questions **Q7** and **Q8**, **select one** of the two questions.

Then, mark the **(s)** 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.

[Program Description]

A team coach always tells his players to improve their performance every practice. He keeps tracks of each player's run time for every lap, most especially each player's best time (lowest time record) for the practice.

- (1) The coach records each player's run time for every lap. The number of laps made by each player differs from one player to another. At the end of the practice, the coach tells each player about his/her best time, and also tells the players about the team's best time.
- (2) The program gets the number of players who showed up for the practice, and the number of laps made by each player (this is equivalent to the number of rows). Then the program gets the run time of each lap for each player (this is equivalent to the number of columns per row).
- (3) Assuming that the input data has no errors, and the highest run time will never exceed 10,000.
- (4) The following list shows an output example of this program.

```
Enter the number of players who practiced today: 3
Enter number of laps made by player 1: 4
Enter number of laps made by player 2: 5
Enter number of laps made by player 3: 4
Enter the score of each player:
  Enter the 4 run times of player 1: 12 13 15 11
  Enter the 5 run times of player 2: 10 9 16 12 17
  Enter the 4 run times of player 3: 13 15 16 12
The best time for player 1 is 11
The best time for player 2 is 9
The best time for player 3 is 12
The best time for this practice is 9
```

(5) This program uses the following library function:

`void *calloc (size_t num, size_t size);`

Parameters: num Number of elements.

 size Length in bytes of each element.

Function: Allocates an array in memory with elements initialized to 0.

[Program]

```
#include <stdio.h>
#include <stdlib.h>
#define UPPER_LIMIT 10000

int** createArray();
void recordScores(int** arrayOfScores);
void findTime(int** arrayOfScores);
int scoreMinimum(int* rowPtr);

int main() {
    int ** arrayOfScores;

    arrayOfScores = createArray();
    recordScores(arrayOfScores);
    findTime(arrayOfScores);
}

int** createArray() {
    int rowNum;
    int numLaps;
    int ** arrayOfScores;
    int playersCtr = 0;

    printf("\nEnter the number of players who practiced today: ");
    scanf("%d", &rowNum);
    arrayOfScores = A;
    while (playersCtr < rowNum) {
        printf("Enter number of laps made by player %d: ",
            playersCtr + 1);
        scanf("%d", &numLaps);
        arrayOfScores[playersCtr] =
            (int*)calloc(numLaps + 1, sizeof(int));
        arrayOfScores[playersCtr][0] = numLaps;
        playersCtr++;
    }
    arrayOfScores[playersCtr] = NULL;
    return B;
}
```

```

void recordScores(int** arrayOfScores) {
    int lap, playernum = 0;

    printf("Enter the score of each player: \n");
    while (arrayOfScores[playernum] != NULL) {
        printf(" Enter the %d run times of player %d: ",
            arrayOfScores[playernum][0], playernum+1);
        for (lap = 1; lap <= *arrayOfScores[playernum]; lap++) {
            scanf("%d", arrayOfScores[playernum] + lap);
        }
        playernum++;
    }
}

void findTime(int** arrayOfScores) {
    int lapScore = 0;
    int bestTimeOfPlayer = 0;
    int bestOverAllTime = UPPER_LIMIT;
    int playerno = 0;

    while (arrayOfScores[lapScore] != NULL) {
        bestTimeOfPlayer = scoreMinimum(C);
        printf("The best time for player %d is %d\n",
            ++playerno, bestTimeOfPlayer);
        bestOverAllTime = bestOverAllTime <
            bestTimeOfPlayer ? bestOverAllTime : bestTimeOfPlayer;
        lapScore++;
    }
    printf("The best time for this practice is %d\n",
        bestOverAllTime);
}

int scoreMinimum(int* playerPtr) {
    int column;
    int playerMin = UPPER_LIMIT;
    for (column = 1; column <= *playerPtr; column++)
        if (playerMin < *(playerPtr + column))
            playerMin = playerMin;
        else
            playerMin = *(playerPtr + column);
    return playerMin;
}

```

Subquestion 1

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

Answer group for A

- a) `**rowNum`
- b) `(int*) calloc(rowNum + 1, sizeof(int))`
- c) `(int**) calloc(rowNum + 1, sizeof(int*))`
- d) `(int*) calloc(rowNum, sizeof(int))`
- e) `(int**) calloc(rowNum, sizeof(int*))`
- f) `(int**) calloc(rowNum, sizeof(int))`

Answer group for B

- a) `0`
- b) `arrayOfScores`
- c) `arrayOfScores[playersCtr]`
- d) `numLaps`
- e) `playersCtr`
- f) `rowNum`

Answer group for C

- a) `arrayOfScores`
- b) `* arrayOfScores`
- c) `** arrayOfScores`
- d) `arrayOfScores[lapScore]`
- e) `* arrayOfScores[lapScore]`
- f) `lapScore`

Subquestion 2

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

- (1) Function D requires a pointer to an array of integers as input parameter.
- (2) If an array of pointers that points to 8 arrays of integers was created after the function `int** createArray(void)` was executed. Variable E was assigned the integer value 8.

Answer group for D

- a) `int** createArray(void);`
- c) `void findTime (int** arrayOfScores);`
- b) `void recordScores (int** arrayOfScores);`
- d) `int scoreMinimum (int* playerPtr);`

Answer group for E

- a) `arrayOfScores`
- b) `numLaps`
- c) `playersCtr`
- d) `rowNum`

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

[Program Description]

The programs implement a subsystem of online bookstore. There are three types of customers: Guest, Member, and Gold Member. Guest customers purchase books without any discount, Member customers purchase books with 5% discount, and Gold Member customers purchase books with 10% discount. Only Gold Member customers can download e-book without paying (free). The programs print out the customer information, purchased book information, and amounts of payments.

When executing the programs, the following list will be printed out.

```
-----  
Customer No: 1  
Customer name: Bill Tomas  
Book title: 'Java'  
Price: $20  
Quantity: 3  
Total cost: $60  
-----  
Customer No: 2  
Customer name: George Bob  
Book title: 'Software engineering'  
Price: $40  
Quantity: 2  
Total cost: $76  
-----  
Customer No: 3  
Customer name: Dennis Jorge  
Book title: 'System analysis and design'  
Price: $35  
Quantity: 4  
Total cost: $126
```

[Program 1]

```
public A Downloadable {  
    void download();  
}
```

[Program 2]

```
public abstract class Customer {  
    private int customerNumber;  
    private String firstName;  
    private String lastName;  
  
    public Customer(int num, String first, String last){  
        customerNumber = num;  
        firstName = first;  
        lastName = last;  
    }  
    public void setFirstName(String first) { firstName = first; }  
    public String getFirstName() { return firstName; }  
    public void setLastName(String last) { lastName = last; }  
    public String getLastName() { return lastName; }  
    public void setCustomerNumber(int num) { customerNumber = num; }  
    public int getCustomerNumber() { return customerNumber; }  
    public abstract int  
        calculatePayment(int bookPrice, int bookQuantity);  
}
```

[Program 3]

```
public class Guest B Customer {  
    public Guest(int num, String first, String last) {  
        super(num, first, last);  
    }  
    public int calculatePayment(int bookPrice, int bookQuantity) {  
        int amount;  
        amount = bookPrice * bookQuantity;  
        return amount;  
    }  
}
```

[Program 4]

```
public class Member B Customer {  
    private int discount;  
    public Member(int num, String first, String last) {  
        super(num, first, last);  
        setDiscount(5);  
    }  
    public void setDiscount(int dis) { discount = dis; }  
    public int getDiscount() { return discount; }  
    public int calculatePayment(int bookPrice, int bookQuantity) {  
        int amount;  
        amount = bookPrice * bookQuantity * (100 - getDiscount()) / 100;  
        return amount;  
    }  
}
```

[Program 5]

```
public class GoldMember B C  
    D Downloadable {  
    public GoldMember(int num, String first, String last) {  
        super(num, first, last);  
        setDiscount(10);  
    }  
    public void download() {  


The program codes about download processing are not shown.

  
    }  
}
```

[Program 6]

```
public class Book {  
    private String bookTitle;  
    private int price;  
    public Book(String bname, int pr) {  
        bookTitle = bname;  
        price = pr;  
    }  
    public void setBookTitle(String bname) { bookTitle = bname; }  
    public String getBookTitle() { return bookTitle; }  
    public void setPrice(int price) {  
        price = (price < 0) ? 0 : price;  
    }  
    public int getPrice() { return price; }  
}
```

[Program 7]

```
public class Invoice {
    private Customer customer;
    private Book book;
    private int quantity;
    public Invoice(Customer cust, Book bk, int qt) {
        customer = cust;
        book = bk;
        quantity = qt;
    }
    public Customer getCustomer() { return customer; }
    public Book getBook() { return book; }
    public int getQuantity() { return quantity; }
}
```

[Program 8]

```
public class Purchase {
    private static final int BOOKS_COUNT = 3;
    private static final int CUSTOMERS_COUNT = 3;
    private static final int INVOICE_COUNT = 3;
    private Book cBks[];
    private Customer cCustomers[];
    private Invoice cInvoice[];

    public void createBooks() {
        cBks = new Book[BOOKS_COUNT];
        cBks[0] = new Book("Java", 20);
        cBks[1] = new Book("Software engineering", 40);
        cBks[2] = new Book("System analysis and design", 35);
    }

    public void createUsers() {
        cCustomers = new E[CUSTOMERS_COUNT];
        cCustomers[0] = new Guest(1, "Bill", "Tomas");
        cCustomers[1] = new Member(2, "George", "Bob");
        cCustomers[2] = new GoldMember(3, "Dennis", "Jorge");
    }

    public void createSales() {
        cInvoice = new Invoice[INVOICE_COUNT];
        cInvoice[0] = new Invoice(cCustomers[0], cBks[0], 3);
        cInvoice[1] = new Invoice(cCustomers[1], cBks[1], 2);
        cInvoice[2] = new Invoice(cCustomers[2], cBks[2], 4);
    }
}
```

```

public void printSalesInfo() {
    for (Invoice currentInvoice : cInvoice) {
        System.out.println("-----");
        System.out.println("Customer No: " +
            currentInvoice.getCustomer().getCustomerNumber());
        System.out.println("Customer name: " +
            currentInvoice.getCustomer().getFirstName() + " " +
            currentInvoice.getCustomer().getLastName());
        System.out.println("Book title: '" +
            currentInvoice.getBook().getBookTitle() + "'");
        System.out.println("Price: $" +
            currentInvoice.getBook().getPrice());
        System.out.println("Quantity: " +
            currentInvoice.getQuantity());
        System.out.println("Total cost: $" +
            F.calculatePayment(
            currentInvoice.getBook().getPrice(),
            currentInvoice.getQuantity()));
    }
}

```

[Program 9]

```

public class OnlineBookShopTest {
    public static void main(String[] args) {
        Purchase purchaseObj = new Purchase();
        purchaseObj.createBooks();
        purchaseObj.createUsers();
        purchaseObj.createSales();
        purchaseObj.printSalesInfo();
    }
}

```

Subquestion

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

Answer group for A, B and D

- | | |
|---------------|--------------|
| a) abstract | b) extends |
| c) implements | d) import |
| e) instanceof | f) interface |
| g) throws | |

Answer group for C and E

- | | |
|-------------|---------------|
| a) Customer | b) GoldMember |
| c) Guest | d) Member |
| e) Object | |

Answer group for F

- a) ((GoldMember)currentInvoice.getCustomer())
- b) ((Guest)currentInvoice.getCustomer())
- c) ((Member)currentInvoice.getCustomer())
- d) ((Object)currentInvoice.getCustomer())
- e) currentInvoice.getCustomer()