



October, 2007

Fundamental IT Engineer Examination (Afternoon)

Questions must be answered in accordance with the following:

Question Nos.	Q1 - Q5	Q6 , Q7	Q8 , Q9
Question Selection	Compulsory	Select 1 of 2	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 test 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 (Q6-Q7 and Q8-Q9)**

Mark the ⑤ 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 next 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 Reply]

SQ	a	b	c	d
1	Ⓐ	●	Ⓒ	Ⓓ


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

Company names and product names appearing in the test questions are trademarks or registered trademarks of their respective companies. Note that the ® and ™ symbols are not used within.

[Explanation of the Pseudo-Code Description Format]

Pseudo-Language Syntax	Description
○	Declares names, types, etc. of procedures, variables, etc.
▪ Variable ← Expression	Assigns the value of an Expression to a Variable.
 <p>Conditional expression</p> <p>▪ Process</p>	A selection process. If the Conditional expression is True, then Process is executed.
 <p>Conditional expression</p> <p>▪ Process 1</p> <p>▪ Process 2</p>	A selection process. If the Conditional expression is True, then Process 1 is executed. If it is False, then Process 2 is executed.
 <p>Conditional expression</p> <p>▪ Process</p>	A repetition process with the termination condition at the top. The Process is executed while the Conditional expression is True.

[Operator]

Operation	Operator	Priority
Unary operation	+ - not	<div style="text-align: center;">  </div>
Multiplication and division operation	* /	
Addition and subtraction operation	+ -	
Relational operation	> < >= <= =	
Logical product	and	
Logical sum Exclusive logical sum	or	
		Low

[Logic type constant]

true false

Questions 1 through 5 are all compulsory. Answer every question.

Q1. Read the following description of the state machine, and then answer Subquestions 1 through 4.

Figure 1 depicts the state machine which takes two inputs J and K and the current state inputs Y and Z to determine its next state. The state transition is synchronized by the clock input, Ck . The four output states of the state machine are A , B , C and D , and the possible combination of the output is shown in Table 1. These output states are encoded by a 4-to-2 encoder before they are fed back to the input of the state machine. Table 2 shows the truth table of the 4-to-2 encoder.

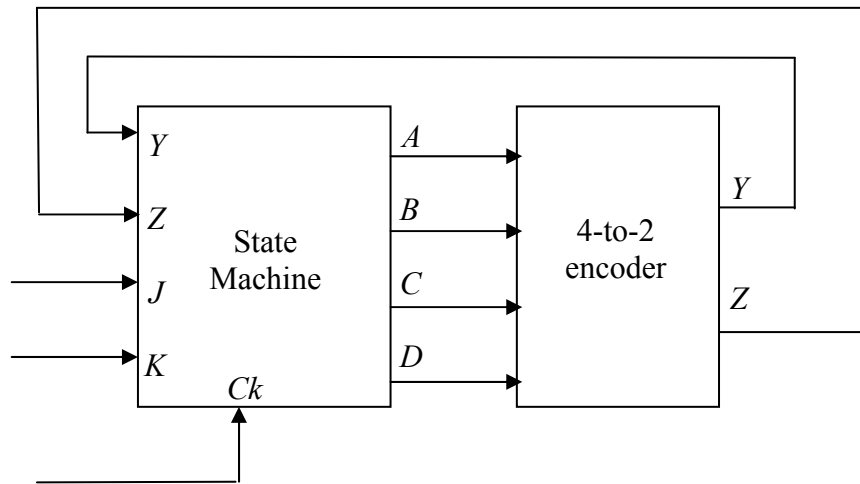


Fig. 1 State machine

Table 1 The possible combination of the output state of the State Machine

Output			
A	B	C	D
1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1

Table 2 4-to-2 encoder truth table

Input				Output	
A	B	C	D	Y	Z
1	0	0	0	0	0
0	1	0	0	0	1
0	0	1	0	1	0
0	0	0	1	1	1

Figure 2 depicts the state transition diagram of the state machine. The two logic inputs at the arc of each state transition represent the J and K inputs, and the 'X' represents both logic 0 and 1, i.e. 01 stands for $J=0$ and $K=1$, whilst 1X stands for $J=1$ and $K=0$ or 1.

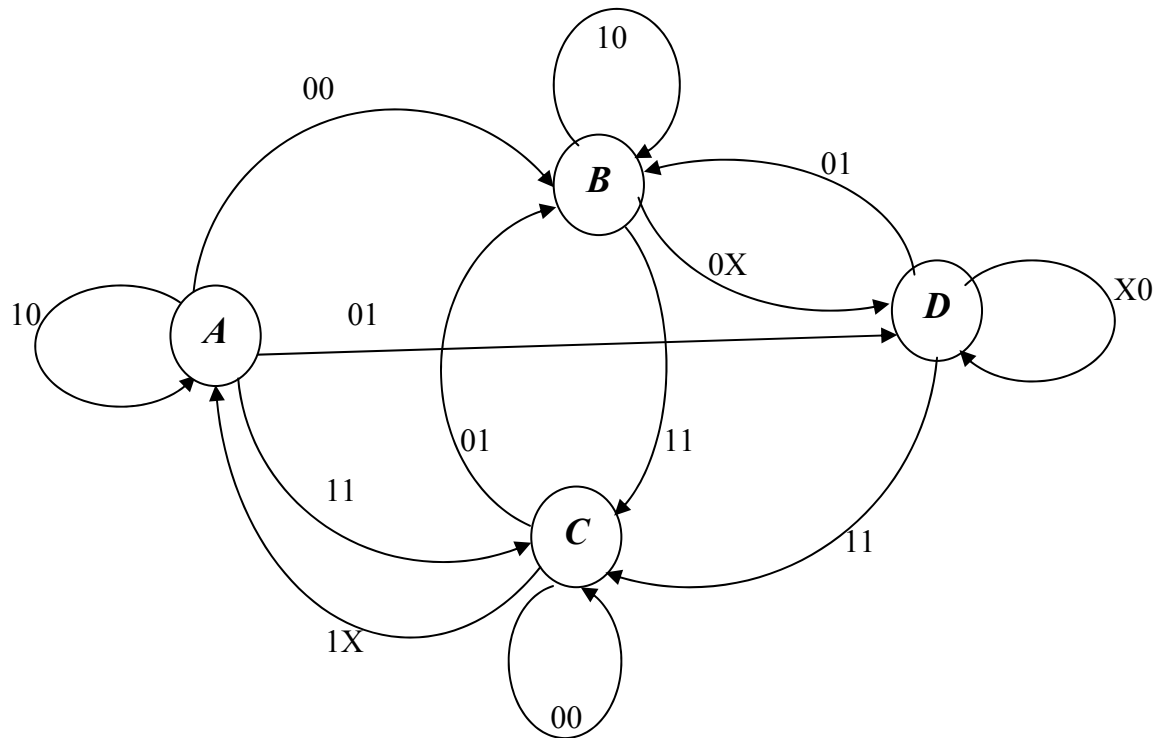


Fig. 2 State transition diagram

Subquestion 1

Suppose the initial state is C . What is the final state if the JK inputs at the next four clocks are 01, 01, 00, 11?

Answer group:

- a) A b) B c) C d) D

Subquestion 2

Based on the truth table in Table 2, which of the Boolean equation given in the answer selection can represent the output Z ?

Answer group:

- | | | |
|---|--|--|
| a) $A + B$ | b) $\overline{A} \bullet \overline{B}$ | c) $A \bullet \overline{B} + \overline{A} \bullet B$ |
| d) $B \bullet \overline{C} + \overline{B} \bullet \overline{C}$ | e) $B \bullet D$ | f) $B \bullet \overline{C} + \overline{C} \bullet D$ |

Subquestion 3

Suppose the initial state is C , Which of the following JK input sequences produce the state transition of $CABDD$?

(Two answers)

Answer group:

- | | | |
|-------------------|-------------------|-------------------|
| a) 00, 01, 00, 11 | b) 10, 00, 00, 10 | c) 10, 00, 01, 11 |
| d) 11, 00, 01, 00 | e) 11, 00, 10, 00 | f) 11, 00, 10, 01 |

Subquestion 4

What is the Boolean equation that can represent the following output of the state machine?

- i. output C
- ii. output D

Answer group:

- a) $\overline{Y} \bullet \overline{Z} \bullet J \bullet \overline{K} + Y \bullet \overline{Z} \bullet J$
- b) $\overline{Y} \bullet \overline{Z} \bullet \overline{J} \bullet K + \overline{Y} \bullet Z \bullet \overline{J}$
- c) $\overline{Y} \bullet \overline{Z} \bullet \overline{J} \bullet K + Y \bullet Z \bullet J \bullet \overline{K}$
- d) $\overline{Y} \bullet \overline{Z} \bullet \overline{J} \bullet K + \overline{Y} \bullet Z \bullet \overline{J} + Y \bullet Z \bullet \overline{K}$
- e) $\overline{Y} \bullet \overline{Z} \bullet J \bullet K + Y \bullet Z \bullet J \bullet K + \overline{Y} \bullet Z \bullet J \bullet K$
- f) $Y \bullet Z \bullet J \bullet K + Y \bullet \overline{Z} \bullet \overline{J} \bullet \overline{K} + \overline{Y} \bullet Z \bullet J \bullet K + \overline{Y} \bullet \overline{Z} \bullet J \bullet K$
- g) $\overline{Y} \bullet \overline{Z} \bullet \overline{J} \bullet \overline{K} + Y \bullet \overline{Z} \bullet J \bullet \overline{K} + \overline{Y} \bullet Z \bullet \overline{J} \bullet K + Y \bullet Z \bullet \overline{J} \bullet K$

Q2. Read the following description of a communication network, and then answer Subquestions 1 through 5.

Figure 1 depicts the connection of the servers and PCs. Company A is given the IP gateway 199.141.27.129/26.

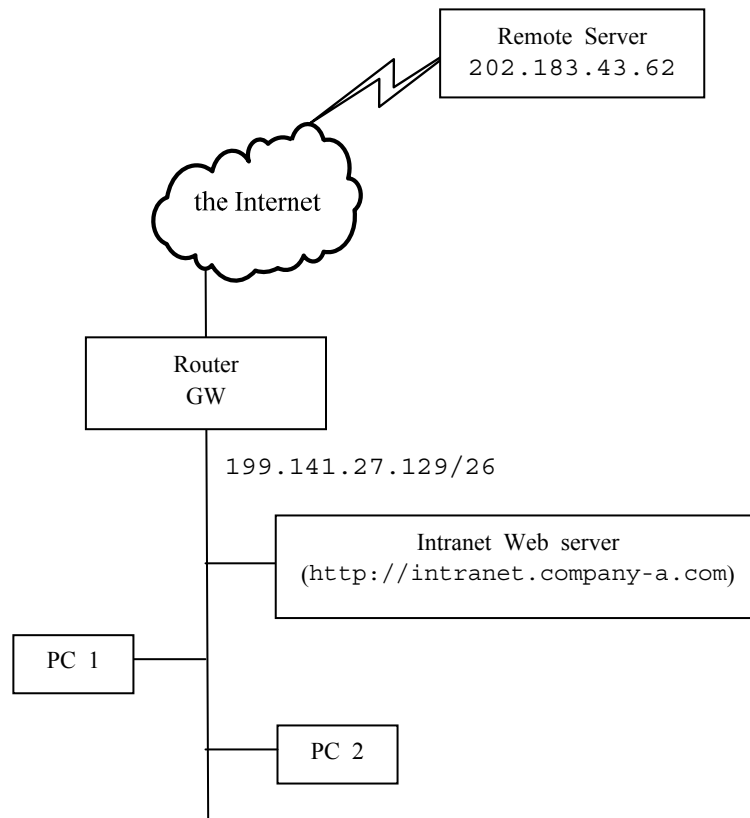


Fig. 1 Network Topology

Subquestion 1

What is the subnet mask that must be used by Company A?

Answer group:

- | | |
|--------------------|--------------------|
| a) 255.255.0.0 | b) 255.255.254.0 |
| c) 255.255.255.0 | d) 255.255.255.128 |
| e) 255.255.255.192 | f) 255.255.255.224 |
| g) 255.255.255.240 | h) 255.255.255.252 |

Subquestion 2

What is the subnet IP address that must be used by Company A?

Answer group:

- | | |
|-------------------|-------------------|
| a) 199.141.0.0 | b) 199.141.26.0 |
| c) 199.141.27.0 | d) 199.141.27.128 |
| e) 199.141.27.192 | f) 199.141.27.224 |
| g) 199.141.27.240 | h) 199.141.27.252 |

Subquestion 3

What is the possible IP address for the PC1 and PC2?

Answer group:

- a) 199.141.27.0, 199.141.27.112
- b) 199.141.27.112, 199.141.27.150
- c) 199.141.27.150, 199.141.27.168
- d) 199.141.27.168, 199.141.27.200

Subquestion 4

Users on network 199.141.27.0/26 are complaining that they cannot access the cooperate intranet server at intranet.company-a.com. In troubleshooting this problem, you find that you able to telnet a workstation (or PC.) on this network to the internal web server via its IP address. What is the likely cause of this problem?

Answer group:

- a) DNS failure.
- b) FTP failure.
- c) SNMP failure.
- d) TCP/IP failure.

Subquestion 5

You are the network administrator of the Company A. You receive a call from a user who is unable to reach a server at a remote site. After further review, you discover the following information:

Local PC – 199.141.27.X
Default gateway – 199.141.27.129
Remote server – 202.183.43.62

You then conduct to following tests from the offending local PC:

```
ping 127.0.0.1 - successful
ping 199.141.27.X - successful
ping 199.141.27.129 - successful
ping www.google.com - successful
ping 202.183.43.62 - unsuccessful
```

Which of the following problems would create the test results listed above?

Answer group:

- a) Local NIC not functioning.
- b) Local physical layer problem.
- c) Remote physical layer problem.
- d) TCP/IP not correctly installed.

Q3. Read the following description of a relational database, and then answer Subquestions 1 and 2.

You work as a Database Developer for ABC Inc. The company uses RDBMS for project management. As shown in **Fig. 1**, the Project database contains four tables, **Projects**, **Employee**, **Project_Modules** and **Work_Done**. The fragments of database schema are given in the exhibit. Each project is assigned to an employee called Project Manager and there can be only one Project Manager for a project. For better planning and management, each project is divided into smaller modules and responsibility of each module is assigned to an Employee. Reporting is done on a daily basis and employees report the work done on their modules. You remove the project information after three months of project completion.

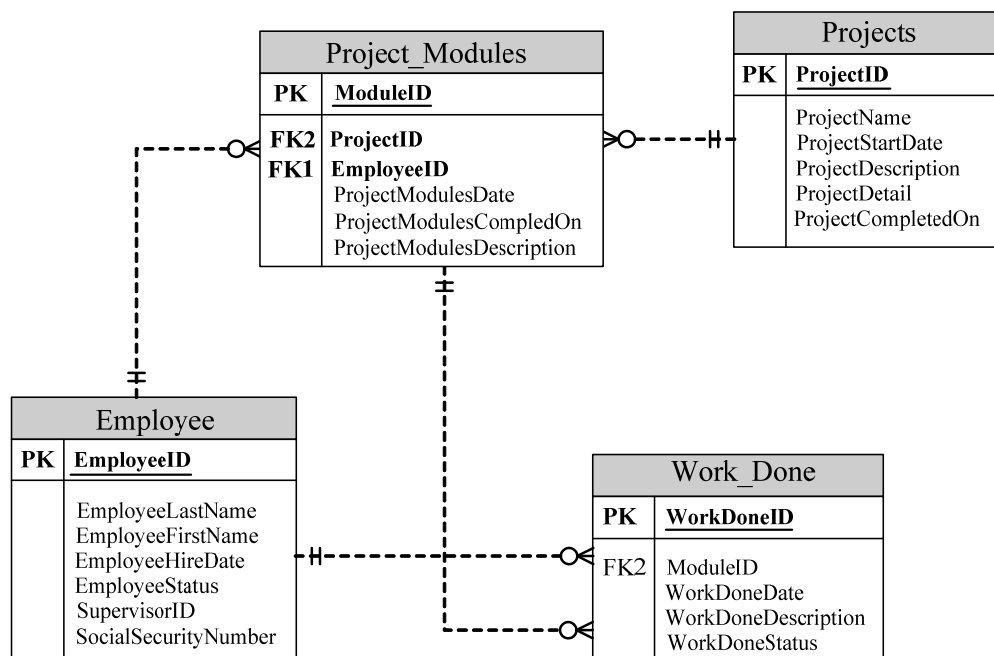


Fig.1 (PK: Primary Key; FK: Foreign Key)

Subquestion 1

Choose the correct statement sequence to successfully remove the projects. Let's say current date is '01/10/2007' and date format is DD/MM/YYYY. Choose only required statements.

1. `DELETE FROM Projects`
`WHERE Projects.ProjectCompletedOn <= '01/07/2007'`
2. `DELETE FROM Work_Done WHERE Work_Done.ModuleID`
`IN (SELECT Project_Modules.ModuleID FROM Project_Modules`
`JOIN Projects`
`ON Project_Modules.ProjectID = Projects.ProjectID`
`WHERE Projects.ProjectCompletedOn <= '01/07/2007')`
3. `DELETE FROM Project INNER JOIN Project_Modules`
`ON Projects.ProjectID = Project_Module.ProjectID`
`INNER JOIN WorkDone`
`ON Project_Modules.ModuleID = Work_Done.ModuleID`
`WHERE Projects.ProjectCompletedOn <= '01/07/2007'`
4. `DELETE FROM Project_Modules`
`WHERE Project_Modules.ProjectID`
`IN (SELECT Project.ProjectID FROM Projects`
`WHERE Projects.ProjectCompletedOn <='01/07/2007')`

Answer group:

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

Subquestion 2

The company also conducts some internal training. Currently, there is no database or application to maintain the training records such as records of students, trainers, courses, and classroom assignments. You are appointed to develop a database to record this information. You design the tables for this database. Fragments of the database are shown in **Fig. 2**.

Courses		ClassRooms		Students	
PK	<u>CourseID</u>	PK	<u>ClassRoomID</u>	PK	<u>StudentID</u>
	CourseDesscription CourseDuration CourseTitle CourseNumber TrainerName TrainerAddress TrainerPhone		ClassRoomNumber ClassTime		StudentName Address City State

Fig. 2 (PK: Primary Key)

You want to promote quick response times for queries and minimize redundant data for company internal training. What will you do?

Answer group:

- Create a new table named **Trainers**. Include **TrainerID** column, **TrainerName** column, **TrainerAddress** column, and **TrainerPhone** column. Add **TrainerID** column to the **Courses** table.
- Move all the columns from the **ClassRooms** table to the **Courses** table and drop the **ClassRooms** table.
- Remove the **ClassRoomID** column, and base the PRIMARY KEY constraint on the **ClassRoomNumber** and **ClassTime** columns.
- Remove the PRIMARY KEY constraint from the **Courses** table and replace the PRIMARY KEY constraint with a composite PRIMARY KEY constraint based on the **CourseID** and **CourseTitle**.

Q4. Read the following description of a program and the program itself, and then answer Subquestions 1 and 2.

[Program Description]

This is the program which manipulates a singly-linked list. The singly-linked list consists of at most 100 cells, and each cell is stored at the cell positions 1 - 100 in the cell area. When a cell is inserted to the list, the cell gets a position available in the cell area and is linked to the position. When a cell is deleted from the list, the link is disconnected from the position and the cell becomes available again.

Each cell consists of link part and data part. The position of the next cell is stored in the link part of a cell. The employee ID (1 - 1000) and the age (20 - 60) are stored in the data part. The position of the first cell of the list is stored in the variable ROOT. 0 is stored in the link part of the last cell. When a list has not established yet, i.e., there is no cell, 0 is stored in the ROOT. The figure 1 below shows that n cells are linked from ROOT.

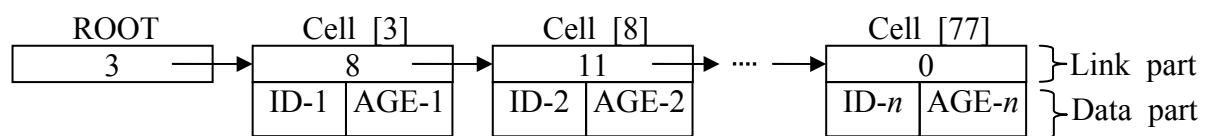


Fig. 1 An example of a singly-linked list

- (1) The cell area is created as a structure array. A cell is referred by an index of a structure array, 1 - 100. -1 is stored in the link part of available cells.
- (2) A subprogram `searchCell()` searches available cells. The return value of this program is the position of the cell when an available cell is found, or 0 when an available cell is not found.
- (3) `add (id, age)` is a subprogram, which searches available cells, stores the data and links to the head of the list. The return value of this program is the position of the added cell when the operation is completed, or 0 when an available cell is not found.
- (4) `delete (id)` is a subprogram, which searches a cell with the employee ID specified by `id`, and deletes the cell found first. The return value of this program is the position of the deleted cell when the operation is completed, or 0 when the cell with the `id` is not found. The deleted cell becomes available again in the cell area.
- (5) `exchange(ptr)` is a subprogram, which exchanges the order of the cells referred by `ptr` for the next cell. The return value of the program is `ptr` when the operation is completed, or -1 when the cells could not be exchanged, because the cell referred by `ptr` was at the last of the list.

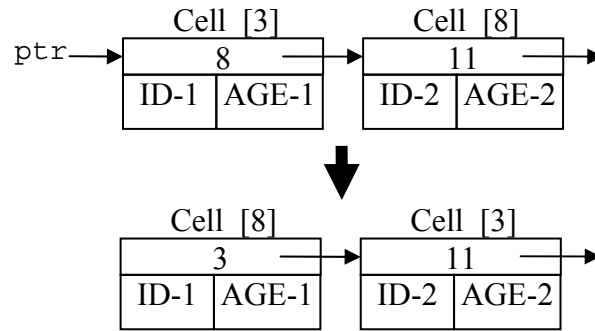


Fig. 2 An example of exchange

(6) The arguments of the subprograms are shown in the tables below.

Table 1 Specification of the arguments of `add`

Variable	Type	Input / output	Meaning
<code>id</code>	integer	input	Employee ID
<code>age</code>	integer	input	Age

Table 2 Specification of the argument of `delete`

Variable	Type	Input / output	Meaning
<code>id</code>	integer	input	Employee ID

Table 3 Specification of the argument of `exchange`

Variable	Type	Input/Output	Meaning
<code>ptr</code>	integer	input	The original position of the cell before exchange

[Program]

```
o structure: litem{integer:chain, id, age} Cell[100]
/* Index of structure array begins with 1. */
```

```
o integer: ROOT
```

```
o searchCell()
```

The program which searches available cells is provided.

```
o add(integer: id, integer: age)
```

```
o integer: ptr
```

```
▪ ptr ← searchCell()
```

```
  ↑ ptr > 0
```

```
  ▪ Cell[ptr].chain ← ROOT
```

```
  ▪ Cell[ptr].id ← id /* stores the employee ID */
```

```
  ▪ Cell[ptr].age ← age /* stores the age */
```

```
  ↓ ▪ A
```

```
▪ return ptr /* ptr is the return value of the function */
```

```
o delete(integer: id)
```

```
o integer: find, i, pi, ptr
```

```
▪ find ← 0
```

```
▪ i ← ROOT
```

```
▪ pi ← 0
```

```
■ i > 0 and find = 0
```

```
  ↑ Cell[i].id ≠ id
```

```
  ▪ pi ← i
```

```
  ▪ B
```

```
  ▪ find ← i
```

```
  ▪ ptr ← Cell[find].chain
```

```
  ▪ Cell[find].chain ← -1
```

```
  ↑ pi = 0
```

```
  ▪ ROOT ← ptr
```

```
  ↓ ▪ C
```

```
▪ return find /* find is the return value of the function */
```

```
o exchange(integer: ptr)
```

```
o integer: rtnv, ptr1, ptr2
```

```
▪ rtnv ← -1
```

```
▪ ptr1 ← Cell[ptr].chain
```

```
  ↑ ptr1 > 0
```

```
  ▪ ptr2 ← Cell[ptr1].chain
```

```
  ▪ Cell[ptr].chain ← ptr2
```

```
  ▪ D
```

```
  ▪ rtnv ← ptr
```

```
▪ return rtnv /* rtnv is the return value of the function */
```


Subquestion 1

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

Answer group for A:

- | | |
|--------------------------|----------------------------|
| a) <code>ptr ← 0</code> | b) <code>ptr ← ROOT</code> |
| c) <code>ROOT ← 0</code> | d) <code>ROOT ← ptr</code> |

Answer group for B:

- | | |
|---|--------------------------------------|
| a) <code>find ← Cell[find].chain</code> | b) <code>find ← Cell[i].chain</code> |
| c) <code>i ← Cell[find].chain</code> | d) <code>i ← Cell[i].chain</code> |

Answer group for C:

- | | |
|-------------------------------------|--------------------------------------|
| a) <code>Cell[i].chain ← i</code> | b) <code>Cell[pi].chain ← i</code> |
| c) <code>Cell[i].chain ← pi</code> | d) <code>Cell[pi].chain ← pi</code> |
| e) <code>Cell[i].chain ← ptr</code> | f) <code>Cell[pi].chain ← ptr</code> |

Answer group for D:

- | | |
|--|---|
| a) <code>Cell[ptr].chain ← ptr</code> | b) <code>Cell[ptr].chain ← ptr1</code> |
| c) <code>Cell[ptr1].chain ← ptr</code> | d) <code>Cell[ptr1].chain ← ptr1</code> |
| e) <code>Cell[ptr2].chain ← ptr</code> | f) <code>Cell[ptr2].chain ← ptr1</code> |

Subquestion 2

From the answer groups below, select the correct answers to be inserted into the blanks in the description below.

When the program below is executed with the programs of the list, the number of the cell in the list is E. And when the data part is represented as (employee ID, age), the data part of the first cell to which ROOT links is F.

[Program]

```
o integer: ROOT
o integer: i
  ■ i:1, i<5, 1
    ■ add(3×i, i+30)
    ■ add(7×i, i+40)
  ■
  ■ i:1, i<3, 1
    ■ delete(3×i)
    ■ delete(7×i)
  ■
  ■ exchange(ROOT)
```

Answer group for E:

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

Answer group for F:

- a) (3, 31) b) (6, 32) c) (7, 41) d) (9, 33)
e) (12, 34) f) (14, 42) g) (21, 43) h) (28, 44)

- Q5.** Read the following descriptions of a program design, and then answer Subquestions 1 and 2.

One program is developed to register the subscription, to deliver the journals, magazines and newspapers (hereafter referred as journals) to subscribers at home and to calculate the charge of subscribers for their subscriptions.

[Program Design Description]

The underlined column's name denotes the primary key.

- (1) Table **Type_of_journals**

Row	1	2	3	← Column
↓	<u>Type_ID</u>	Journal_type	No_issues_per_year	
1	Y	Yearly	1	
2	HY	Half-Yearly	2	
3	M	Monthly	12	
4	SM	Semi-Monthly	24	
5	BW	Biweekly	26	
6	W	Weekly	52	
7	D	Daily	365*	

* 366 for leap year

- (2) Table **List_of_subscribers**

<u>Subscriber_ID</u>	Subscriber_name	Address	Phone
----------------------	-----------------	---------	-------

- (3) Table **Journals**

Journals table lists all the journals that the distributing center deals with. For clearing purpose, this table is given with some records:

<u>Journal_ID</u>	Journal_name	Type_ID	Unit_Price
W1	The World	W	10
W2	CNTT	W	15
M1	ATGT	M	20
SM1	TCNN	SM	25
...

- (4) People can subscribe to one or more journals. First, the program creates all the empty tables, which have the name taken from first column of the **Journals** table. Each of these table (hereafter, Subscription Tables) have the structure like this, for example, the following structure for the weekly journal “The World”:

Table **W1**

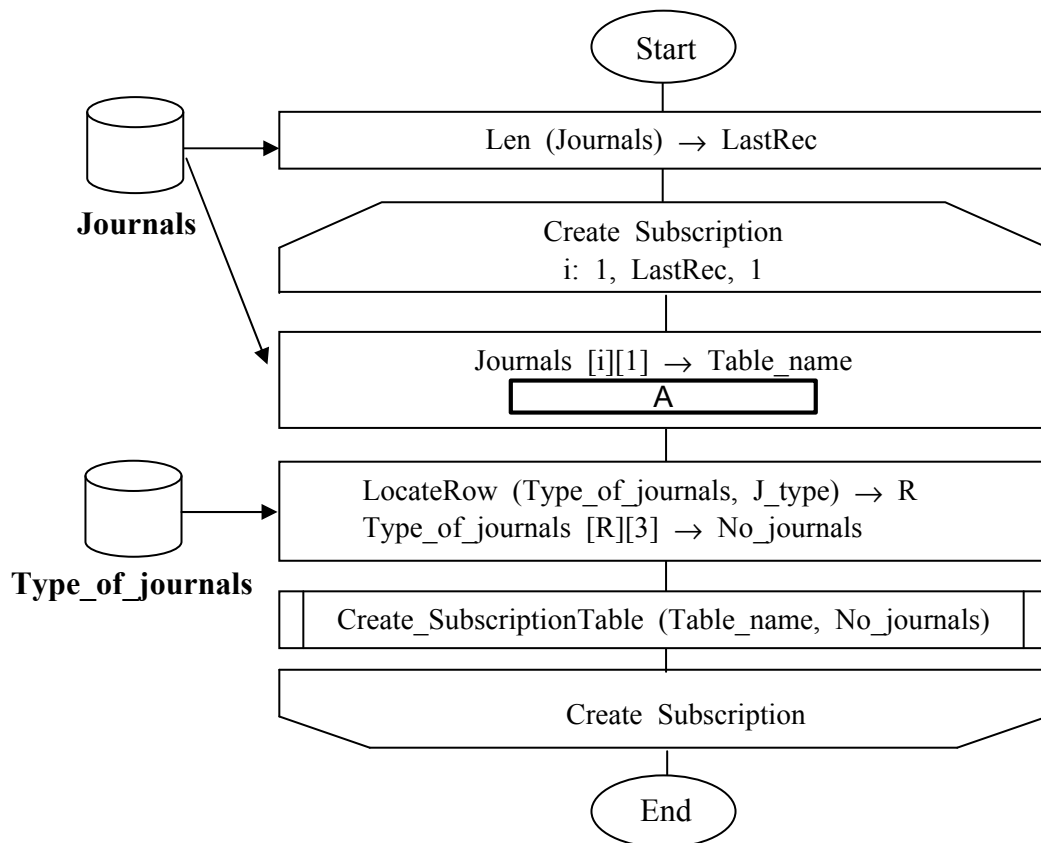
<u>Subscriber_ID</u>	N1	N2	N3	...		N52	Charge
1	1	*	*			*	520
2	1	*	*				
...							

- (i) The **W1** (name of the table) is taken from the first column of the **Journals** table.
The type of journal is taken from the third column of the **Journals** table.
- (ii) The first column always is named **Subscriber_ID**. The last column always is named **Charge**: The total payment for the subscription of this journal. The number of the remaining columns varies, depending on the specific journal and equals the number of issues per year for that journal. This number is defined as the third column of the **Type_of_journals** table. In this case, there are 52 columns, representing 52 issues per year (N1, N2, N3... N52) of the weekly journal “The World”.

(5) Functions Description:

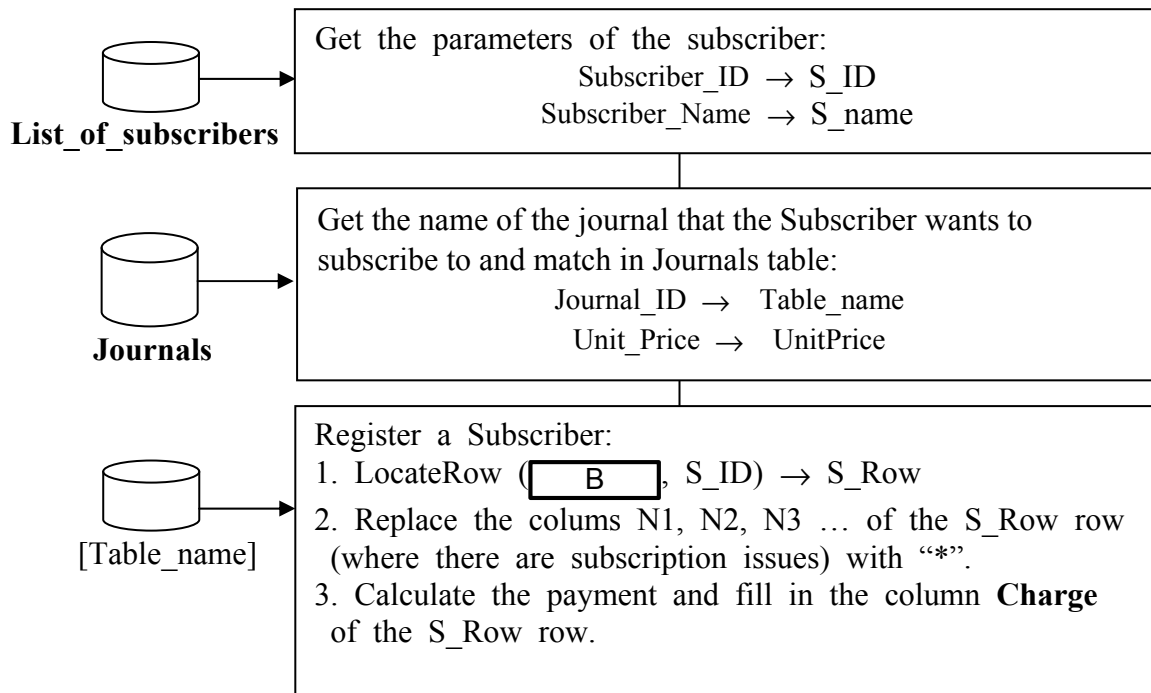
Function Name	Description
Len (table_name)	Len () returns the number of rows of the table_name table.
LocateRow(table_name,value)	LocateRow() returns the row number of the table_name table, where value is a specific value of the primary key. If the row does not exist, it returns 0.
Create_SubscriptionTable (Table_name, No_journals)	Create_SubscriptionTable creates an empty table as shown in item (4) in the [Program Design Description] .

[Flow chart 1: Creating Subscription Tables]



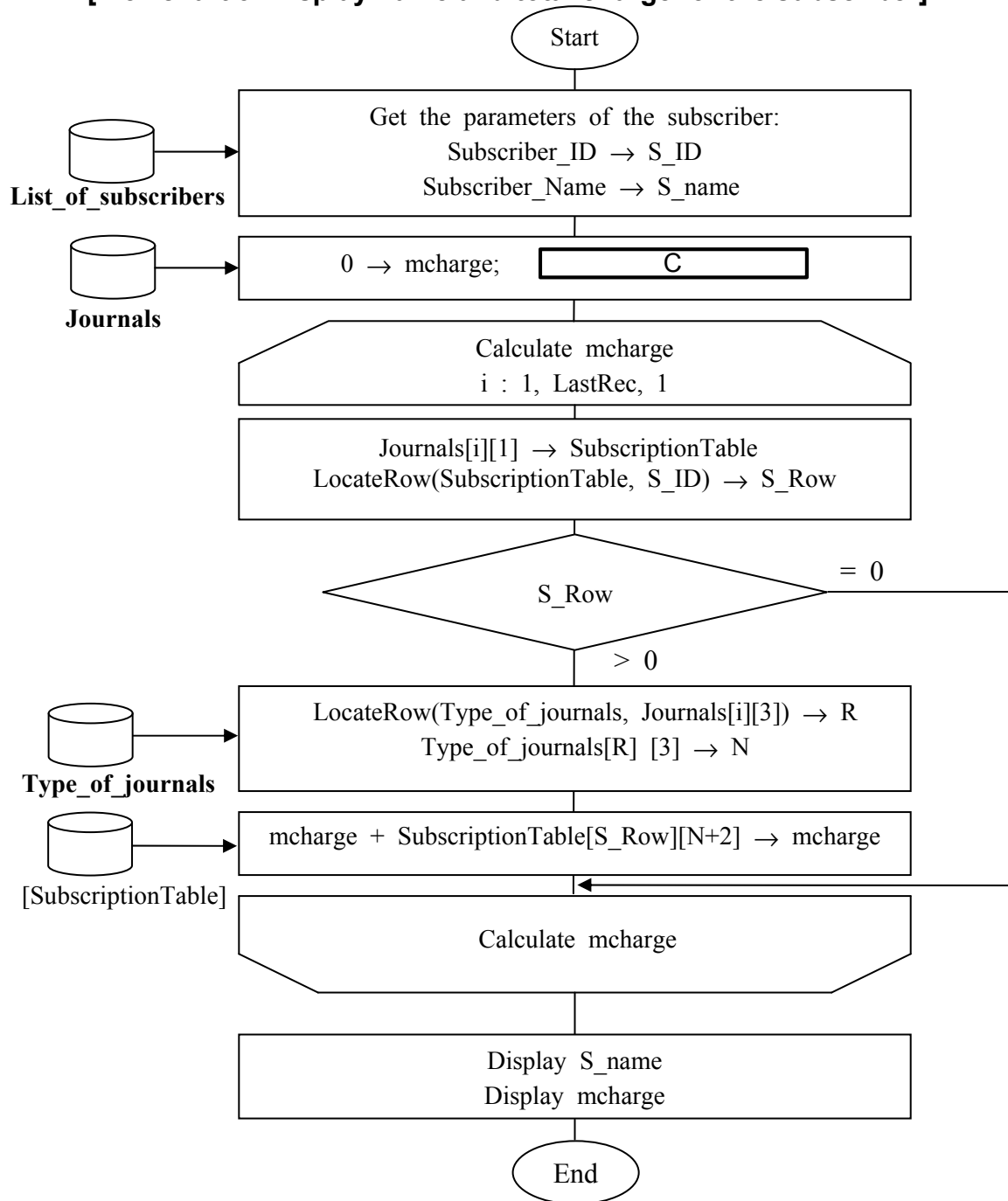
Note: The specification of the loop conditions is as follows:
variable name: initial value, terminal value, increment

[Flowchart 2: System flowchart of Registration process (for a Journal)]



The **Distribution process** is similar to the Registration process, except that it replaces “*” with “1” for the each delivered issue and it does not calculate the payment, since it was already calculated in the Registration process.

[Flowchart 3: Display name and total charge for the subscriber]



Subquestion 1

From the answer group below, select the correct answer to be inserted into the blanks in the flowcharts.

Answer group for A:

- a) Journals [i][1] → J_type
- b) Journals [i][2] → J_type
- c) Journals [i][3] → J_type
- d) Type_of_journals[i][1] → J_type

Answer group for B:

- a) Journals
- b) S_name
- c) Table_name
- d) [Table_name]
- e) Type_of_journals

Answer group for C :

- a) Len (Journals) → LastRec
- b) Len (Journals) + 1 → LastRec
- c) Len (Journals) - 1 → LastRec
- d) Len (Type_of_journals) → LastRec
- e) Len (Type_of_journals) + 1 → LastRec
- f) Len (Type_of_journals) - 1 → LastRec

Subquestion 2

In the case that a subscriber discontinues and annuls his (or her) unfinished subscription, he (or she) will be reimbursed and the program needs to recalculate the total reimbursed money for that journal. Which of the following statements is correct?

Answer group:

- a) Reimbursed Money = UnitPrice * Total number of the columns that contain "1"
- b) Reimbursed Money = UnitPrice * Total number of the columns that contain "*"
- c) Reimbursed Money = UnitPrice * Total number of the columns that contain "*" and "1"
- d) Reimbursed Money = UnitPrice * Total number of the columns that contain "*" or "1"

Select one question from **Q6** or **Q7**, mark (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.

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

[Program Description]

A magic square is an arrangement of the numbers from 1 to n^2 in an $n \times n$ matrix, with each number occurring exactly once, and such that the sum of the entries of any row, any column, or any main diagonal is the same. It is not hard to show that this sum must be $n(n^2+1)/2$.

$n = 4$			
1	15	14	4
12	6	7	9
8	10	11	5
13	3	2	16

n (n is an integer which is multiple of four) ordering magic square (multiple of 4 series) will be created by following rule. First, all the numbers from one to $n \times n$ are written in order from left to right across each row in turn, starting from the top left hand corner. Second, the central square with sides of length $n/2$ should be retained. Also retain the squares with sides of length $n/4$ in each of four corners. The others are interchanged their diametrically opposite number.

function tofill()

numbers from one to $n \times n$ are written in order from left to right, starting the top left hand corner.

function change_rows_diagonally()

value of top and bottom rows of array are changed diagonally.

function change_cols_diagonally()

value of right and left columns of array are changed diagonally.

[Program]

```
#define MAX_SIZE 160
int magic_square[MAX_SIZE][MAX_SIZE];

void tofill(int arr_size );
void change_rows_diagonally(int arr_size, int unchanged_cells, int row);
void change_cols_diagonally(int arr_size, int unchanged_cells, int row);

void main()
{
    int arr_size, unchanged_cells, row;
    // code section to get size of an array to be built.
    scanf("%d",&arr_size);
    if(arr_size%4 != 0 || arr_size > MAX_SIZE){
        printf("arr_size must be multiple of 4 and not more than MAX_SIZE");
        exit(1);
    }
    tofill(arr_size);
    A
    // Find square ordering size whose values remain without changing for each corner of array.
    for(row = 0; row < arr_size /2; row++) {
        if(row < unchanged_cells)
            change_rows_diagonally(arr_size, unchanged_cells, row);
        else
            change_cols_diagonally(arr_size, unchanged_cells, row);
    }
} // end of main

//***** change rows diagonally*****
// function – value of top and bottom rows of array are changed diagonally

void change_rows_diagonally(int arr_size, int unchanged_cells, int row)
{
    int col, temp;
    for(col=unchanged_cells; col<(arr_size-unchanged_cells); col++) {
        temp = magic_square[row][col];
        magic_square[row][col] = B;
        B = temp;
        // Interchange numbers with their diametrically opposite numbers.
    }
}
```



```

//***** change cols diagonally*****
// function – value of right and left columns of array are changed diagonally.

void change_cols_diagonally(int arr_size, int unchanged_cells, int row)
{
    int col, temp;
    for(col=0; col<unchanged_cells; col++) {
        temp = magic_square[row][col];
        magic_square[row][col] = B;
        B = temp;
        temp = magic_square[row][arr_size-col-1];
        magic_square[row][arr_size-col-1] = C;
        C = temp;
        // Interchange numbers with their diametrically opposite numbers.
    }
}

// function – numbers from one to  $n \times n$  are written in order from left to right, starting the top left hand corner.

void tofill(int arr_size)
{
    int row, col, count = 1;
    D
    // Change row, column numbers when filling row in order from left to right.

    magic_square[row][col] = count++;
}

```


Subquestion

From the answer groups below, select the correct answers to be inserted into the blanks

in the above program.

Answer group for A:

- a) `unchanged_cells = arr_size ;`
- b) `unchanged_cells = arr_size/2 ;`
- c) `unchanged_cells = arr_size/3 ;`
- d) `unchanged_cells = arr_size/4 ;`

Answer group for B:

- a) `magic_square[arr_size + row + 1][arr_size + col - 1]`
- b) `magic_square[arr_size + row + 1][arr_size - col - 1]`
- c) `magic_square[arr_size + row - 1][arr_size + col - 1]`
- d) `magic_square[arr_size + row - 1][arr_size - col - 1]`
- e) `magic_square[arr_size - row + 1][arr_size + col - 1]`
- f) `magic_square[arr_size - row - 1][arr_size + col - 1]`
- g) `magic_square[arr_size - row - 1][arr_size - col - 1]`

Answer group for C:

- a) `magic_square[arr_size + row + 1][col]`
- b) `magic_square[arr_size + row - 1][col]`
- c) `magic_square[arr_size - row + 1][col]`
- d) `magic_square[arr_size - row - 1][col]`

Answer group for D:

- a) `for(row = 0; row < arr_size; row++)`
`for(col = 0; col < arr_size; col++)`
- b) `for(row = 0; row < arr_size; row++)`
`for(col = arr_size-1; col >=0; col++)`
- c) `for(row = arr_size-1; row >=0; row++)`
`for(col = 0; col < arr_size; col++)`
- d) `for(row = arr_size-1; row >=0; row++)`
`for(col = arr_size-1; col >=0; col++)`

Q7. Read the following description of a Java program and the program itself, and then answer Subquestion.

[Program Description]

The following are classes and test class for a simple expression.

(1) The expression grammar is defined in BNF as follows:

```
expression ::= variable | sequence
sequence  ::= expression + expression | expression - expression
           | expression * expression | expression / expression
```

- (2) Method `eval()` is to evaluate the value of an expression. All subclasses of expression need to implement this method.
- (3) Method `setValue()` is to set an integer to a variable.
- (4) Method `operate()` is to create a new expression by connecting two exist expression with an given operation (e.g. +, -, *, /).

[Program]

```
import java.io.*;

interface Expression {
    int eval();
}

class VarExp implements Expression {
    private int var;
    public VarExp() {};
    public void setValue(int n) {
        var = n;
    }
    public int eval() {
        A
    }
}

class SeqExp implements Expression {
    private int op;
    B
    public SeqExp(Expression e1, Expression e2, int a_op) {
        exp1 = e1;
        exp2 = e2;
        op = a_op;
    }

    public int eval() {
        switch (op) {
            case 0:
                return exp1.eval() + exp2.eval();
        }
    }
}
```



```

        case 1:
            return exp1.eval() - exp2.eval();
        case 2:
            return exp1.eval() * exp2.eval();
        case 3:
            return exp1.eval() / exp2.eval();
    }
    return 0;
}

public SeqExp operate(Expression e, int a_op) {
    C
}

}

public class TestExpression {
    public static void main(String args[]) {

        VarExp a = new VarExp();
        VarExp b = new VarExp();

        SeqExp sum = new SeqExp(a, b, 0);
        SeqExp diff = new SeqExp(a, b, 1);
        SeqExp mul = sum.operate(diff, 2);

        a.setValue(3);
        b.setValue(7);
        System.out.print(mul.eval());
    }
}

```

Subquestion

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

Answer group for A:

- a) return this;
- b) return var;
- c) super.eval();
- d) this.eval();

Answer group for B:

- a) private Expression exp1, exp2;
- b) private int exp1, exp2;
- c) private SeqExp exp1, exp2;
- d) VarExp exp1, exp2;

Answer group for C:

- a) `Expression tmp1 = new SeqExp(exp1, exp2, op);`
`return SeqExp(tmp1, e, a_op);`
- b) `return new SeqExp(exp1, exp2, a_op);`
- c) `return new SeqExp(this, e, a_op);`
- d) `return this.eval;`

Select one question from **Q8** or **Q9**, mark (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.

Q8. Read the following description of a C Program and the program itself, and then answer Subquestions 1 and 2.

[Program Description]

This program takes as input a string containing mathematical expressions in postfix order (operand operand operator). The operands and operators are separated by one space. Operands are all numeric. The program then outputs the result of the expression. This is done by the function `evaluate`. The other functions are for stack operations used in the Stack global variable. Input strings are assumed to be in the correct form and syntax. The functions `push` and `popper` are the Stack operators

[Global Variables]

`stack[50]` – represents a Stack where `next` is the top of the stack.

[Sample Input/Output]

Sample Input (space separators)	Sample Output
5 6 10 * +	65
3 7 + 5 / 6 *	12
25 18 - 5 12 + *	119
108 12 / 30 15 / 5 + -	2

[Program]

```
#include <stdio.h>

float stack[50];
int next = 0;

void push (float a) {
    stack[next] = a;
    next++;
}

float popper() {
    next--;
    return stack[next];
}
```



```

char isop (char a) {
    if ( a=='+'||a=='-'||a=='*'||a=='/' ) return 'o';
    else if ( '0'<= a && a <= '9' )         return 'd';
    else                                     return 's';
}

float calcval (char a) {
    switch (a) {
        case '+': return popper() + popper();
        case '-': return A ;
        case '*': return popper() * popper();
        case '/': return B ;
    }
}

float evaluate (char *p) {
    int curr, len, n, num, prev;
    curr = 's';
    len = strlen(p);
    num = 0;
    for ( n=0; n<=len; n++) {
        prev = curr;
        curr = isop(p[n]);
        switch (curr) {
            case 'o': push (calcval (p[n]));
                        break;
            case 'd': C ;
                        break;
            case 's': if ( D ) {
                            push ((float)num);
                            num = 0;
                        }
                        break;
        }
    }
    return popper();
}

void main () {
    float test1, test2, test3, test4;
    test1 = evaluate ("5 6 10 * +");
    test2 = evaluate ("3 7 + 5 / 6 *");
    test3 = evaluate ("25 18 - 5 12 + *");
    test4 = evaluate ("108 12 / 30 15 / 5 + -");
    printf(" test1 %6.2f test2 %6.2f test3 %6.2f test4 %6.2f",
           test1, test2, test3, test4);
}

```


Subquestion 1

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

Answer group for A and B:

- a) `-1.0 * (popper() - popper())`
- b) `1.0 / popper() * popper()`
- c) `popper() * (1.0 / popper())`
- d) `popper() / popper()`
- e) `popper() - popper()`

Answer group for C and D:

- a) `num = 10 * num + (p[n] - '0')`
- b) `num = 10 * num + (p[n-1] - '0')`
- c) `num = num + 10 * p[n]`
- d) `num = num + 10 * p[n-1]`
- e) `num != 0`
- f) `prev == 'd'`
- g) `prev == 'o'`

Subquestion 2

Modify the main program as follows, and then execute the program.

```
void main () {  
    float test;  
    test = evaluate ("3 7 + 5 / 6 *");  
    printf(" %6.2f %6.2f %6.2f %6.2f %6.2f",  
          test, stack[0], stack[1], stack[2], stack[3]);  
}
```

From the answer group below, select the correct answer which shows the output by the `printf` statement in the above main program.

Answer group

- a) 12.00 3.00 7.00 5.00 6.00
- b) 12.00 3.00 7.00 10.00 5.00
- c) 12.00 3.00 10.00 2.00 12.00
- d) 12.00 12.00 2.00 0.00 0.00
- e) 12.00 12.00 2.00 6.00 0.00

- Q9.** Read the following description of a Java program and the program itself, and then answer Subquestion.

[Program Description]

This Java program demonstrates the example of bank operation.

Bank

Bank class has main method. It contract SavingsAccount, CurrentAccount and CurrentPlusAccount. It obtains monthly interest and current balance of each account.

Account

This is an abstract class of account.

It declares the abstract method `monthlyInterest`, which calculate monthly interest. It also performs followings.

- create an account with a given identity number and initial balance
- return the identity number and current balance
- deposit some positive amount into the account, increasing the balance
- decrease the balance by a specified positive amount; if the amount is greater than the balance, throw an `IllegalArgumentException`

SavingsAccount

This calss is inherit from `Account`. It performs followings.

- create a new savings account with a given annual interest rate, as well as the parameters required for all accounts
- withdraw a positive amount that does not exceed the current balance, decreasing the balance by the amount withdrawn
- calculate the monthly interest by multiplying the current balance by the annual interest rate divided by twelve

CurrentAccount

This class is inherit from `Account`. It performs followings.

- create a new checking account with a given per-check charge, as well as the parameters required for all accounts
- clear a check for a given amount by decreasing the balance by the amount of the check plus the per-check charge
- Bank will not pay any interest for Current Account.

CurrentPlusAccount

This calss is inherit from `Account`. It performs followings.

- create a new special checking account with a given minimum balance and interest rate, as well as the parameters required for a checking account
- clear a check for a given amount according to the rules above
- calculate the monthly interest by multiplying current balance by the annual interest rate divided by twelve if the current balance is above the minimum; otherwise, calculate the interest as it is done for a current account.

[Program 1]

```
public static void main(String[] args) {
    SavingsAccount Sv = new SavingsAccount(1111, 1000.0, 7);
    CurrentAccount Ca = new CurrentAccount(2222, 2000.0, 1.2);
    CurrentPlusAccount Cp =
        new CurrentPlusAccount(3333, 3000.0, 1, 200, 2.5);

    System.out.println(
        "Current Plus Account Balance : >>>>>>>>> " + Cp.currentBalance());
    System.out.println(
        "Savings Account Balance : >>>>>>>>> " + Sv.currentBalance());
    System.out.println(
        "Current Account Balance : >>>>>>>>> " + Ca.currentBalance());
    System.out.println(
        "Current Plus Account Interest : >>>>>>>>> " + Cp.monthlyInterest());
    System.out.println(
        "Savings Account Interest : >>>>>>>>> " + Sv.monthlyInterest());
    System.out.println(
        "Current Account Interest : >>>>>>>>> " + Ca.monthlyInterest());
}
```

[Program 2]

```
public abstract class Account {
    private int idNum;          // identity number for this account
    private double balance;     // current balance for this account

    public Account(int idNumber, double startBal) {
        if (startBal>0.0)
        {
            idNum=idNumber;
            balance=startBal;
        } else
            throw new IllegalArgumentException();
    }

    public int idNumber(){ return idNum; }
    public double currentBalance(){ return balance; }
    public void deposit(double amount) {
        if(amount>0.0)
            A
        else
            throw new IllegalArgumentException();
    }

    public void decreaseBalance(double amount) {
        if(amount>=0.0 && amount<=balance)
            balance-=amount;
        else
            throw new IllegalArgumentException();
    }
}
```



```

    public abstract double monthlyInterest();
    {}
}

```

[Program 3]

```

public class SavingsAccount extends Account {
    private double intRate; // annual interest rate for

    public SavingsAccount(int idNumber, double balance, double rate) {
        B;
        intRate=rate;
    }
    public double monthlyInterest() {
        return (currentBalance() * (intRate / 12.0));
    }
    public void withdraw(double amount) {
        decreaseBalance(amount);
    }
}

```

[Program 4]

```

public class CurrentAccount extends Account {
    private double checkCharge;

    public CurrentAccount( int idNumber, double startBal, double chkCharge)
    {
        super(idNumber, startBal);
        checkCharge = chkCharge;
    }
    public void clearCheck(double amount) {
        C;
    }
    public double monthlyInterest() {
        return 0.0;
    }
}

```


[Program 5]

```
public class CurrentPlusAccount D {  
    private double minBalance;  
    private double intRate;  
  
    public CurrentPlusAccount(int idNumber, double startBal,  
                             double chkCharge, double minBal, double rate) {  
        super(idNumber, startBal, chkCharge);  
        minBalance= minBal;  
        intRate = rate;  
    }  
    public void clearCheck(double amount) {  
        if (currentBalance() >= minBalance)  
            decreaseBalance(amount);  
        else  
            super.clearCheck(amount);  
    }  
    public double monthlyInterest() {  
        if (E)  
            return (currentBalance() * (intRate / 12.0));  
        else  
            return super.monthlyInterest();  
    }  
}
```


Subquestion

From the answer groups below, select the correct answers to be inserted into the blanks

in the above program.

Answer group for A:

- a) `balance+=amount;`
- b) `idNum=0`
- c) `throw IllegalArgumentException()`
- d) `throw new IllegalArgumentException()`

Answer group for B:

- a) `super()`
- b) `super(idNumber)`
- c) `super(idNumber, balance)`
- d) `super(intRate)`

Answer group for C:

- a) `decreaseBalance(amount)`
- b) `decreaseBalance(amount + checkCharge)`
- c) `decreaseBalance(chkCharge)`
- d) `decreaseBalance(checkCharge)`

Answer group for D:

- a) `extends Account`
- b) `extends Account,SavingsAccount`
- c) `extends CurrentAccount`
- d) `implement SavingsAccount`

Answer group for E:

- a) `currentBalance()== 0`
- b) `currentBalance()>= 0`
- c) `currentBalance()>= minBalance`
- d) `minBalance>=0`