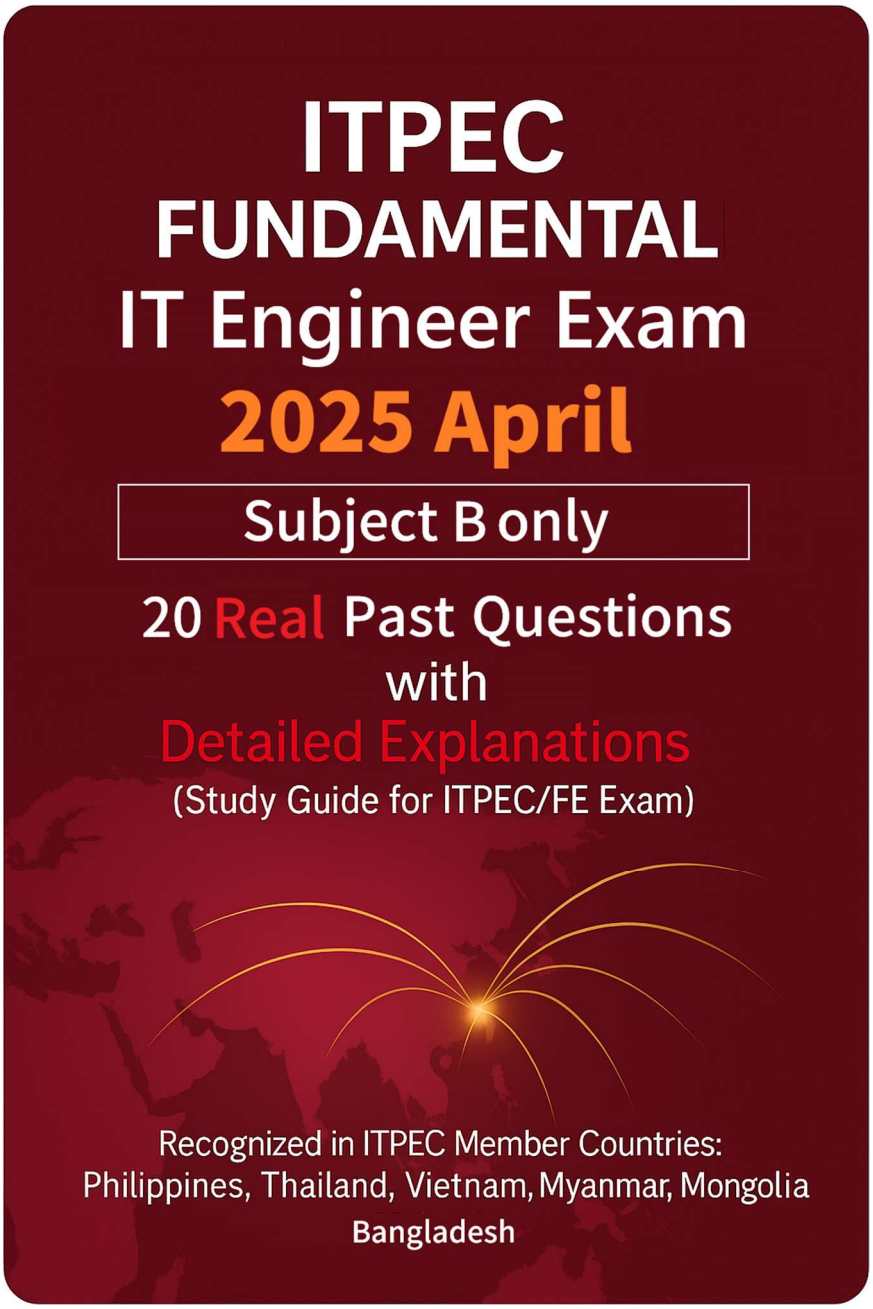
Sample Edition



ITPEC Fundamentals of Information Technology Engineer Examination (FE)

– English Edition Spring 2025 Subject B Version

Author: Takashi Narita

Based on past **FE Examination** questions published by ITPEC.

Disclaimer

This book contains past exam questions from the Fundamental Information Technology Engineer Examination (FE), administered by the Information Technology Professionals Examination Council (ITPEC), specifically from the Spring 2025 Examination. The questions are reproduced with permission for educational purposes, and the copyright of the original questions remains with ITPEC.

All explanations, translations, and analyses are original work by the author, Takashi Narita.

Preface

The **Fundamental Information Technology Engineer Examination (FE)** is a core examination that demonstrates your ability to apply fundamental knowledge of information technology, programming, algorithms, and system design. Originally developed in Japan, it is now recognized internationally through the ITPEC (Information Technology Professionals Examination Council) mutual recognition framework. Member countries include the Philippines, Thailand, Vietnam, Myanmar, Mongolia, and Bangladesh.

In many of these countries, the FE serves as a gateway for those aiming to work in Japan or for Japanese companies abroad. Where Japanese-owned IT firms operate, holding this certification can also provide a strong advantage in hiring and career advancement, as it reflects both technical competence and familiarity with Japanese corporate culture and IT standards.

This book is designed for learners who wish to prepare for the **FE Examination in English**. It offers clear explanations, answer analyses, and study tips to help you build genuine understanding rather than relying on rote memorization. Whether you are a student, a working professional, or someone seeking to advance your IT career, this book will support efficient, practical, and motivating study.

Important Notice

The ITPEC Fundamentals of Information Technology Engineer Examination (FE) consists of two parts: Subject A and Subject B.

Each subject has different characteristics and types of questions.

This book covers only Subject B.

➞ For Subject A, please purchase the separate volume.

➞ A combined edition (Subjects A & B) is also available.

About the Author

Takashi Narita is an IT instructor with more than 20 years of industry experience in system design, software development, programming, and database management. Now based in the Philippines, he teaches programming, system development, and system design remotely to students in Japan.

He began creating this English edition of **Fundamental Information Technology Engineer Examination (FE)** practice questions to support IT human resource development in the Philippines. Over time, he recognized that it could also serve as a valuable resource for learners across Asia seeking to succeed in Japanese-owned companies or in Japan itself.

Through this book, he aims to help readers build the skills and confidence needed to contribute meaningfully in Japanese and Japan-affiliated workplaces. He is passionate about making complex IT concepts accessible to learners of all backgrounds and believes that studying in English opens doors to international opportunities.

How to Use This Book

The fastest way to pass the **Fundamental Information Technology Engineer Examination (FE)** is to practice past questions repeatedly and understand the logic behind each answer.

This book provides past FE questions in English, along with clear explanations and answer analyses. Always review the explanations for any questions you miss, make sure you understand the reasoning, and then try again.

Recommended study flow:

1. Solve one set of questions under timed conditions, just like the actual exam.
2. Check which answers were incorrect and study the explanations carefully.
3. Reattempt the same set a few days later until you can achieve a high score with confidence.
4. Move on to the next set and repeat the process.

Study Tips

• **Do more than one set** – One set of past questions is not enough for thorough preparation. Aim for at least three sets to build both knowledge and confidence.

• **Consistency is key** – Even 10 minutes a day makes a difference. Use your phone, PC, or tablet to open this book and solve just one question. From my own experience earning multiple IT certifications, daily habit is the single most important factor in success. For me, solving one question before bed soon grew naturally to two or three without feeling forced. Short, consistent study sessions lead to long-term retention.

• **Set a study schedule** – Work backward from your exam date, create a plan, and track your progress.

• **Get used to English IT terms** — many are 3–4-letter abbreviations (e.g., CPU, LAN, DNS) and can feel intimidating at first, but with repeated exposure they'll become second nature.

Exam Overview

• **Format: Two parts** — **Subject A** (formerly “Morning”) and **Subject B** (formerly “Afternoon”).

• **Exam Areas (Subject A):** 60 multiple-choice questions in 90 minutes.

Subject A consists of **60 questions** drawn from the three domains below.

1) Technology Domain

IT fundamentals, hardware, software, databases, networks, security, etc.

2) Management Domain

project management, service management, system audit, etc.

3) Strategy Domain

management strategy, business law, system strategy, corporate activities, etc.

You should aim to spend about 1.5 minutes (90 seconds) per question.

• **Exam Areas (Subject B):** 20 multiple-choice questions in 100 minutes.

Algorithms & Programming (pseudo-code); Information Security.

(**20 questions** total: 16 Algorithms & Programming, 4 Information Security)

You should aim to spend about 5 minutes per question.

• **Passing Criteria (FE):**

You must pass both Subject A and Subject B. The passing mark for each subject is 60 out of 100.

Links

The test mode (CBT or paper), schedule, application method, and venue vary by country. Details are subject to change; always check your country's official website for the latest information.

Official Information Links:

- ITPEC Official Site: https://itpec.org/
- Philippines (DICT): https://dict.gov.ph/itpec/
- Thailand (NECTEC): https://www.nectec.or.th/itpec/
- Vietnam (VITC): https://www.vitc.vn/
- Myanmar (ITPEC Myanmar): https://www.myanmaritpec.org/
- Bangladesh (BCC): https://www.bcc.gov.bd/
- Mongolia (MITC): https://www.mitc.gov.mn/

Author's Note: A quick word on Subject B

Subject B tests **programming & algorithms**.

Unlike Subject A—where past-like items can be solved by "recognition"—you'll **rarely** see the exact same algorithm. What matters is **understanding and application**.

When it feels hard — do this

1. Peek if needed, fill the blanks, then **trace the finished code** yourself.
2. Use **tiny inputs** and a quick table (track i, j, sum, max, …).
3. Check **stopping conditions & boundaries** (loop end, indices, empty/one/many).
4. Learn the **shape** of formulas (e.g., relative error, Hamming distance)—deep math not required.

With these habits, fill-in questions get much easier. Unfamiliar names won't stop you if you can **trace correctly**.

Bottom line: Subject B is often the hurdle. **Master basics, repeat patterns, move one steady step at a time.**

Note: Styles and coverage can change—review the latest syllabus and past papers.

*ITPEC provides an official **Pseudo-code Specification** in the downloadable past exams. For authoritative syntax/semantics, refer to those documents.*

Exams are always a race against time. If you glance at a question and feel a strong sense of dread, move on to another one. The best strategy is to start with the questions that make you think, "I can do this!"

B-Q1

From the answer group below, select the correct combination of answers to be inserted into ____A____ through ____C____ in the program.

Centurial years refer to the years that are divisible by 100. The centurial years are not leap years except for years that are exactly divisible by 400. The non-centurial years, that is, years that are not divisible by 100, refer to leap years that are divisible by 4. The function isLeapYear receives an integer number year and returns true if a given year is a leap year or false otherwise.

[Program]

```
O boolean: isLeapYear(integer: year) // returns true if the variable year
                                     // is a leap year; otherwise, returns false
  if ( A )
    return B
  elseif ( C )
    return false
  elseif (year mod 4 = 0)
    return true
  else
    return false
  endif
```

Answer group

	A	B	C
a)	year mod 100 = 0	false	year mod 400 = 0
b)	year mod 100 = 0	true	year mod 400 = 0
c)	year mod 100 ≠ 0	false	year mod 400 ≠ 0
d)	year mod 100 ≠ 0	true	year mod 400 ≠ 0
e)	year mod 400 = 0	false	year mod 100 = 0
f)	year mod 400 = 0	true	year mod 100 = 0
g)	year mod 400 ≠ 0	false	year mod 100 ≠ 0
h)	year mod 400 ≠ 0	true	year mod 100 ≠ 0

(Source:2025S,FE,Subject-B,Q1)

Answer: f

A = year mod 400 = 0

B = true

C = year mod 100 = 0

Explanation:

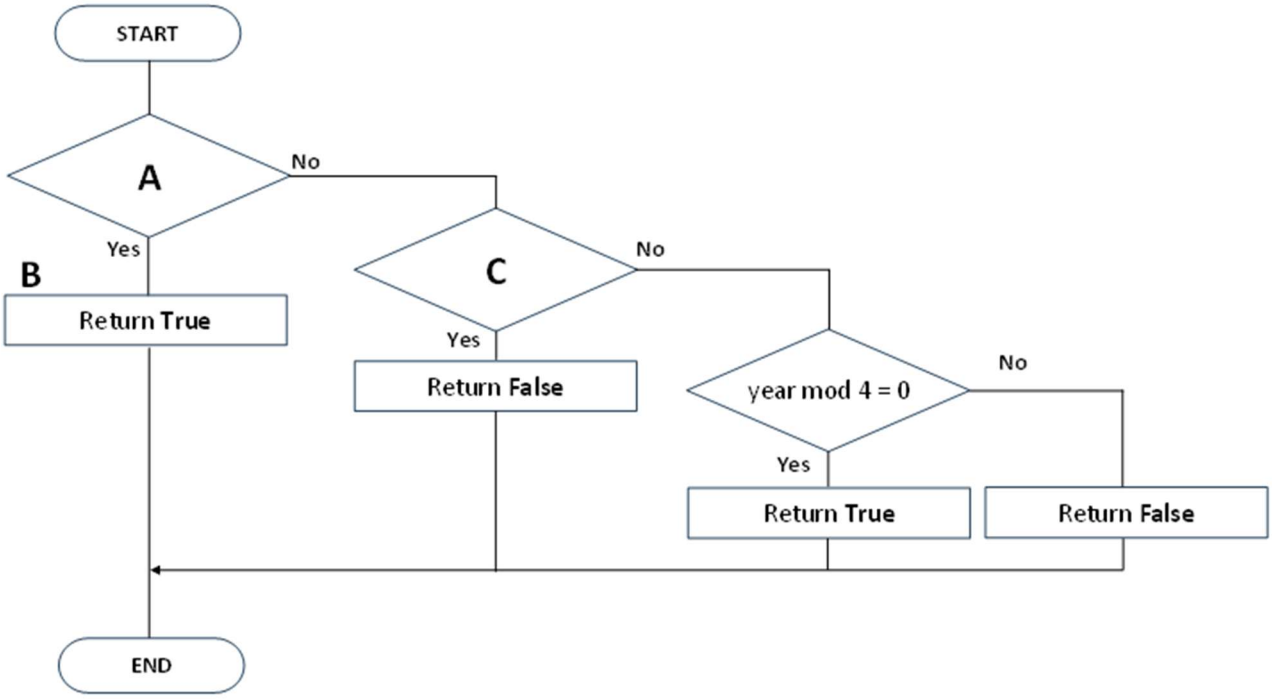
There are three rules for determining leap years:

Rule 1: If a year is divisible by 4, it is a leap year.

Rule 2: However, if it is divisible by 100, it is not a leap year.

Rule 3: However, if it is divisible by 400, it is a leap year.

This logic is implemented in the program, which evaluates a given year and returns **True** if it is a leap year, or **False** otherwise.



I have created a flowchart for review. In the flowchart, **A** and **B** should be considered as a pair: if the condition at **A** is satisfied, then the program returns the value specified at **B**.
An important principle in such processing is to **check the larger numbers first**.
If you begin by checking divisibility by 4, you must then add additional checks for the exceptions at 100 and 400, which makes the logic more complicated.

Therefore, the first check should be whether the year is divisible by 400.
Looking at the answer choices, this corresponds to "**year mod 400 = 0**" and "**true**"
In other words, if the year is divisible by 400, the program returns **true**.

This already determines the correct answer, but let us also confirm the next step, represented by **C** in the flowchart. At **C**, when the condition is satisfied, the program returns **false** (indicating **not a leap year**).
This matches **Rule 2(if it is divisible by 100, it is not a leap year.)**, and the correct expression is "**year mod 100 = 0**"

Author's Comments
Finally, the most important skill in analyzing programs is the ability to **visualize the flowchart from the code**. Notice how constructs like **elif** represent conditional branches.
If you can understand that the program and the flowchart perform the "same logic," you will be well prepared. Without this fundamental skill, it will be very difficult to pass Subject B of the FE examination. Please make sure you understand it thoroughly.

B-Q2

From the answer group below, select the correct combination of answers to be inserted into ___A___ and ___B___ in the program.

The function **`isPerfect`** receives positive number n, and returns whether n is a perfect number. Here, a number is a “perfect number” if the sum of its positive divisors (excluding the number itself) is equal to the number itself. For instance, 28 is a perfect number because 28 has divisors 1, 2, 4, 7, and 14, and 1 + 2 + 4 + 7 + 14 = 28.

[Program]

```
O boolean: isPerfect(integer: n)
  integer: k
  integer: sum ← 0
  integer: half ← integer part of (n ÷ 2)
  for (increase k from 1 to half by 1)
    if ( A )
      B
    endif
  endfor
  if (sum = n)
    return true
  else
    return false
  endif
```

Answer group

	A	B
a)	n mod k ≠ 0	sum ← sum + 1
b)	n mod k ≠ 0	sum ← sum + k
c)	n mod k = 0	sum ← sum + 1
d)	n mod k = 0	sum ← sum + k

(Source:2025S,FE,Subject-B,Q2)

Answer: d

A = n mod k = 0

B = sum ← sum + k

Explanation

At first glance, the concept of a "perfect number" may seem difficult, but the program itself is not very complicated.

There are two main points:

1. Find the numbers that divide evenly (where the remainder is zero).
2. Add those numbers to the total one by one.

Finally, the program checks whether the total sum of divisors is equal to the original number.

Now, let's walk through the program.

The variable sum is initialized to zero so that we can accumulate the divisors.

The variable half is set to half of the number being tested.

This is because, when searching for divisors, any value greater than half does not need to be checked—only the number itself could divide in that range.

A for loop is used here. Let's examine the loop condition.

The loop variable k runs from 1 to half, increasing by 1 at each step.

Now, let's think about the blank A inside the if statement.

Looking at the answer choices, we see expressions using n (the number being tested) and k (the loop variable) with the modulo operator.

Since we want to find numbers that divide evenly, the correct condition is n mod k = 0.

Next, what should we do if the remainder is zero?

In that case, the number k is a possible component of the perfect number, so we add it to the running total sum.

In other words, we add k to sum.

Recall again that the loop variable k runs from 1 to half, increasing by 1.

For example, if the given number is 28, then the loop runs from 1 to 14.

This means the code inside the for … endfor block is executed 14 times.

Understanding this execution flow is very basic, yet extremely important.

By this point, the overall structure of the program should be clear.

We are dividing the number 28 by every integer from 1 through 14.

The divisors that divide evenly (where n mod k = 0) are 1, 2, 4, 7, and 14.

These numbers are successively added to the variable sum.

Finally, if the value of sum is equal to the original number, it is identified as a perfect number, and the function returns **True**.

Author's Comments

How was that? You also encountered the for loop here.

When learning about loop processing, the most important thing is to understand the loop condition precisely: Where does it start (the initial value), where does it end (the final value), and by how much does it increase (the step size)?

In this explanation, I emphasized "increasing by 1."

Since most loops increment by 1, you don't always have to think about it too deeply, but it is still important to read the loop definition carefully.

Take your time to fully understand this.

The difficulty level here is still relatively low.

However, if you cannot grasp problems at this level, it will be very difficult to solve the ones that follow.

Notes and Disclaimers

Source Note:

The questions in this book are based on the past questions from the Spring 2025 FE (Fundamentals of Engineering) Examination published on the official ITPEC website. The author translated them into English and added supplementary explanations.

Source: ITPEC – FE Examination Past Questions (https://itpec.org/)

Relationship with ITPEC:

This book is an independent publication. It is created in accordance with the usage policy for past exam questions published by ITPEC, but it is not officially affiliated with, authorized, or endorsed by ITPEC.

General Disclaimer:

The content of this book is based on information available at the time of writing. The examination format, scope, or content may change. Please refer to the official ITPEC website for the most up-to-date **information.**

License and Usage:

This book is intended for personal study purposes. Redistribution or commercial use of the explanations and translations by the author without permission is prohibited.

Feedback Welcome:

If you find any errors or have suggestions for improvement, please feel free to contact the author. Your input will help improve future editions of this book.

Copyright & Source Information

This book contains past exam questions from the **FE (Fundamentals of Information Technology Engineer) Examination** published by ITPEC, reproduced in accordance with ITPEC's public usage policy for educational purposes. The copyright of the original questions remains with ITPEC.

Source: https://itpec.org/