

はじめに

応用情報技術者試験及び高度共通午前Ⅰ試験は、共通キャリア・スキルフレームワークをモデルとして、レベル3の基準を満たすかどうかを判定する試験です。出題範囲は技術全般の分野が含まれるテクノロジ系知識と、プロジェクトマネジメント、サービスマネジメントといったマネジメント系知識の他、システム戦略、経営戦略、企業と法務といったストラテジ系知識も含まれる幅広い分野になっています。

応用情報技術者の午前試験では、この広い範囲から出題される80問の60%以上を正解しなければいけません。また、高度試験でも、それぞれの専門分野に関する試験の他に、午前Ⅰ試験として、応用情報技術者試験と同じ範囲から出題される30問の60%以上に正解しなければなりません。

全ての分野を得意としていることが理想ですが、技術系分野が得意な人、ストラテジ系分野が得意な人など、これだけの広範囲であれば、分野によって得意・不得意が出てくるのが普通です。本書は、各分野の出題ポイントを効率良く理解して、知識を習得できるよう、最新の出題内容を詳細に分析し、今後の試験での出題内容を予想した“午前試験対策書”の最新版として工夫をしています。

短時間で効率的に学習を進めるため、はじめに現状での各分野の理解度を測るために、ダウンロードして利用できる「学習前診断テスト」を用意しています。まず、このテストによって各分野の理解度を知ること、重点的な学習が必要な分野を把握しましょう。次に、この書籍で出題ポイントを確認し、問題を解くことによって、その分野の問題を解くために必要な知識を効率的に学習し、定着させましょう。そのために、学習効果が高い問題を厳選し、関連事項も含めたポイントが理解できるように解説を工夫しています。

平成31年度春期の応用情報技術者の午前試験では、得点が50～59点のため、午後試験の採点をされなかった人が約27%いました。本書は、こうした悔しい思いをしないうために、直前の知識整理にも利用できるように工夫されています。上手に利用して学習に励まれ、合格の栄冠を手に入れられることを心よりお祈りしております。

2019年9月

アイテック IT 人材教育研究部

目次

はじめに	3
------------	---

第1部 本書の学習方法と試験のポイント 7

第1章 本書の学習方法	8
第2章 応用情報・高度 午前（I）試験のポイント	13

第2部 午前試験の出題ポイント 21

第1章 基礎理論	22
1.1 数値表現	24
1.2 論理演算とシフト演算	42
1.3 データ構造	54
1.4 基本アルゴリズム	72
1.5 BNF と逆ポーランド記法	91
1.6 待ち行列とその他理論	100
第2章 コンピュータ構成要素	114
2.1 CPU アーキテクチャ	115
2.2 メモリアーキテクチャ	132
2.3 ハードウェア	151
第3章 システム構成要素	162
3.1 システムの信頼性	164
3.2 システムの性能	174
3.3 システム構成	184
第4章 ソフトウェア	197
4.1 プロセス制御	198
4.2 主記憶管理	214
4.3 OSS（オープンソースソフトウェア）	226

第 5 章	ヒューマンインタフェースとマルチメディア	233
5.1	ヒューマンインタフェース	235
5.2	マルチメディア	245
第 6 章	データベース	254
6.1	データモデル	257
6.2	正規化	270
6.3	SQL	285
6.4	DBMS の制御	301
第 7 章	ネットワーク	324
7.1	OSI 基本参照モデルと TCP/IP	326
7.2	LAN	348
7.3	通信サービスと性能計算	365
第 8 章	セキュリティ	385
8.1	暗号化技術	387
8.2	認証技術	396
8.3	セキュリティ管理・評価・対策	406
第 9 章	開発技術	444
9.1	開発プロセスと開発手法	446
9.2	要求分析・設計技法	462
9.3	テストとソフトウェアの品質	477
9.4	ソフトウェア開発管理技術	492
第 10 章	IT マネジメント	496
10.1	プロジェクトマネジメント	498
10.2	サービスマネジメント	521
10.3	システム監査	540

目次

第11章 IT戦略	559
11.1 システム戦略・経営戦略	561
11.2 オペレーションズリサーチ (OR) と インダストリアルエンジニアリング (IE)	600
11.3 企業会計・財務	622
11.4 関連法規・ガイドライン	635
11.5 標準化	651

巻末資料

出題範囲

索引

商標表示

各社の登録商標及び商標、製品名に対しては、特に注記のない場合でも、これを十分に尊重いたします。

第1章

part 1

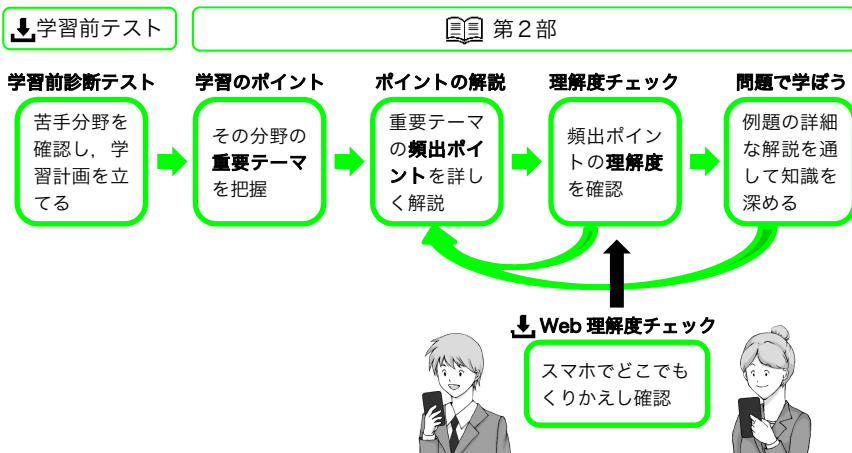
本書の学習方法

本書は、受験者の方が**短時間で効率良く**試験対策できるように構成されています。

【本書の特長】

- ・「学習前診断テスト」で苦手分野を確認してから効率的に学習を進める構成
- ・幅広い分野から出題される午前問題のレベル3の知識について、以下の流れで解説します。
 - ①重要なテーマの頻出ポイントを解説
 - ②厳選された、学習効果が高い問題を例題として掲載
 - ③基礎概念をより深く理解できるよう、オリジナルに書き下ろした詳細な例題解説
- ・節ごとの「理解度チェック」で確実に理解を深める構成

本書の学習手順は次のようになります。



注記 📖 は書籍、↓ はダウンロードコンテンツ/Webコンテンツ。

第2章

part 1

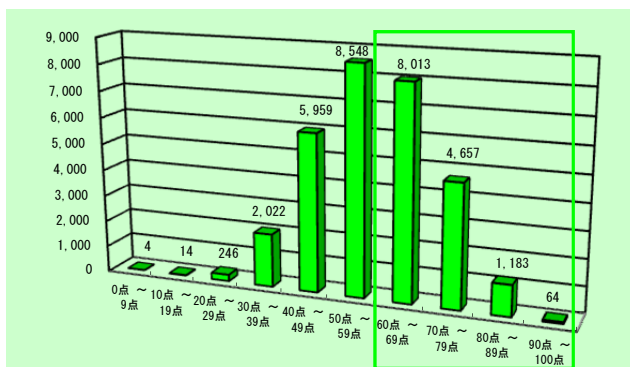
応用情報・高度 午前(I)試験のポイント

応用情報技術者（以下、応用情報）試験の午前試験と高度情報処理技術者（以下、高度）試験の午前I試験について、出題傾向と学習方法を中心に、八つのポイントを説明します。

(ポイント1) 合格に必要な正答率 60%以上は、簡単に取れない

応用情報と高度は、午前、午後試験（論文を除く）ともに満点が100点で、両方の試験で60点以上得点できれば合格です。午前の問題は各問同じ配点で60%以上（応用情報は48問、午前Iは18問）正解できれば合格になります。

次のグラフ（図表1）は平成31年度春期応用情報の午前試験の得点分布を表しています。合格点の60点以上を取った人は13,917人（受験者の約45.3%）で、午後試験の答案が採点されない人が約54%と半数以上を占めています。正答率60%以上と聞くと簡単そうですが、問題は幅広い分野から出題されるので、応用情報の48問は意外にハードルが高いといえます。ただし、午前試験の不合格者のうち、合格まであと10点以内（50～59点）の人は8,548人おり、午前不合格者の約50%も占めています。このように「あと1問」で悔しい思いをしないためにも、**経験のない分野や苦手分野については、早めに学習を始め、正答率としては70%を目標に学習を進める**ことをお勧めします。



図表1 H31 春応用情報処理技術者試験 午前得点分布

第 1 章

part 2

基礎理論

▶▶▶ Point

学習のポイント

基礎理論, アルゴリズムとプログラミングという二つの中分類から構成され, 内容は非常に広範囲にわたります。試験での出題数は応用情報で 7 問程度, 高度共通で 3 問程度と, いずれも全問題数の 1 割程度となっています。問題は, 分野全体から出題されていますが, ここでは出題頻度の高い, 数値表現, 論理演算とシフト演算, データ構造, 基本アルゴリズム, BNF (Backus-Naur Form; バックナース記法) とポーランド記法, 待ち行列とその他理論について取り上げます。

(1) 数値表現

数値表現には, 基数法, 基数変換, 補数表現, 2 進データ (固定小数点形式, 浮動小数点形式), 10 進データ (パック 10 進数, ゾーン 10 進数) などが含まれますが, 中でも出題率の高いものは基数変換, 補数表現, 浮動小数点形式です。さらに, 誤差に関する問題の出題も予想されます。

(2) 論理演算とシフト演算

論理演算とシフト演算に関する問題は, 必ずと言ってよいほど出題されています。基本事項を中心に理解しておきましょう。また, 命題や集合についても扱います。

(3) データ構造

データ構造に関しては, リスト, スタック, キュー, 木などの特徴やこれらに対するデータの追加・削除などの問題が多く出題されています。

1.1 数値表現

▶▶▶ Explanation

ポイントの解説

よく出題される問題のパターンは、次のように分類できます。

- ・ m 進整数を n 進整数に変換する最も基本的なもの
- ・ n 進数の数値同士の演算
- ・ n 進小数を扱うもの
- ・ 番号の割当て問題に基数の考え方をを用いるもの

どの問題も基数の考え方を理解しておけば解ける問題ですが、ここではより速く、より確実に得点するためのポイントをまとめ、演習問題を通して実際に出題されたときにどのような手順で解いていけばよいかをマスターしましょう。

(1) 基数変換

① 基数法

基数と重みに掛ける各桁の数を使って表現する方法を基数法といいます。2 を基数とする 2 進数は次のように表現できます。

$$(101.01)_2 = 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2}$$

② 基数変換の計算

小数部分を含む基数変換、2 進数と 8 進数、16 進数の相互変換が特に重要です。

- ・ 2 進数、8 進数、16 進数から 10 進数への変換
基数法に従い、10 進数に書き直して数値を求めます。

$$(F5A)_{16} = 15 \times 16^2 + 5 \times 16^1 + 10 \times 16^0 = 3,930$$

- ・ 10 進数から 2 進数、8 進数、16 進数への変換
10 進数から n 進数へ変換する場合、基数法の表現に注目してみると、整数部は n で割る操作で下位の桁から順に上位の桁を求めることができ、小数部は n を掛ける操作で小数第 1 位から順に下位の桁を求めることができます。

▶▶▶ Check

理解度チェック ▶ 1.1 数値表現

次の文中の に適切な用語を入れてください。

- (1) 2進数の $(1.11)_2$ を10進数で表現すると ア , 16進数の $(A5.C)_{16}$ は イ になります。また, 10進数の $(21.25)_{10}$ を2進数で表現すると ウ , 16進数では エ になります。
- (2) 8桁で表現される2進数では, $(11110011)_2$ の2の補数は オ なので, 負数を2の補数で表す場合には, $(11110011)_2$ は10進数で カ になります。
- (3) 浮動小数点形式では, $\pm (\text{キ}) \times (\text{ク})^{\text{ケ}}$ で表現される数値を, コ 部, ケ 部, キ 部の三つの部分に分けて記録します。 コ 部は, 数値の正負を1ビットで示し, ケ 部は, 表現する数値が ク の何乗であるかを示します。また, 有効桁数を確保するために キ 部の1ビット目が1になるように調整されますが, この操作のことを サ と呼びます。
- (4) 数値の演算結果には誤差が含まれることがあります。代表的な誤差には, 絶対値の大きな数値に, 絶対値の小さな数値を加減算するとき生じる シ , 値のほぼ等しい数値の減算によって生じる ス , 四捨五入や切捨て, 切上げによって生じる セ , 計算を途中で打ち切ることによって生じる ソ があります。

解答

- (1) ア: $(1.75)_{10} \times (1.11)_2 = 2^0 \times 1 + 2^{-1} \times 1 + 2^{-2} \times 1 = 1 + 0.5 + 0.25$
イ: $(165.75)_{10} \times (A5.C)_{16} = 16^1 \times 10 + 16^0 \times 5 + 16^{-1} \times 12$
 $= 160 + 5 + 12/16$
ウ: $(10101.01)_2 \times (21.25)_{10} = 16 + 4 + 1 + 1/4$
 $= 2^4 \times 1 + 2^2 \times 1 + 2^0 \times 1 + 2^{-2} \times 1$
エ: $(15.4)_{16} \times (21.25)_{10} = (10101.01)_2 = (0001\ 0101.0100)_2$
- (2) オ: $(00001101)_2 \times (00001100)_2 + 1$ ビットを反転して+1
カ: $(-13)_{10} \times (11110011)_2$ の2の補数 $(00001101)_2$ は10進数で $(13)_{10}$
- (3) キ: 仮数 ク: 基数 ケ: 指数 コ: 符号 サ: 正規化
- (4) シ: 情報落ち ス: 桁落ち セ: 丸め誤差 ソ: 打ち切り誤差

▶▶▶ Question

問題で学ぼう

問1 2進数で表現すると無限小数になる10進小数はどれか。

(H26春・AP 問1)

ア 0.375

イ 0.45

ウ 0.625

エ 0.75

解説

2進数を10進数に変換するとき、1桁目に1 ($=2^0$)、2桁目に2 ($=2^1$)、3桁目に4 ($=2^2$)と、それぞれの桁の数字に重みを掛けて計算します。これは2進小数でも同じで小数第1位が $0.5 (=2^{-1}=1/2)$ 、第2位が $0.25 (=2^{-2}=1/4)$ 、第3位が $0.125 (=2^{-3}=1/8)$ となります。そして、第4位は $2^{-4}=1/16=0.0625$ になりますが、ここまでくると電卓なしの計算は困難なので、小数第3位までを考えればよいでしょう。つまり、2進数の小数第3位までの有限小数を10進数に変換した値は0.5、0.25、0.125の組合せで表現できるので、選択肢の10進小数値がこの三つの値の組合せになっているかという観点で調べてみます。

(ア) $0.375=0.25+0.125$ 、(ウ) $0.625=0.5+0.125$ 、(エ) $0.75=0.5+0.25$ なので、それぞれ2進数で(ア) 0.011、(ウ) 0.101、(エ) 0.110と表現できる有限小数です。したがって、無限小数になるのは(イ)の0.45ですが、これは2進小数では $0.01110011\dots$ という無限小数になります。

有限桁の2進数になる10進小数 → 0.5, 0.25, 0.125の組合せ

解答 イ

第6章

part 2

データベース

▶▶▶ Point

学習のポイント

データベースを理解するためのキーワードは、データの共有化です。情報システムが出現してしばらくは、特定の業務だけを支援するために、部門ごとに導入されてきました。そして、データもそれぞれの業務、さらには、プログラム、利用者ごとに必要な内容だけを集めたファイルという形で管理されてきました。しかし、組織内の多くの部門に情報システムの導入が進むと、同じ内容であるはずなのに、部門やファイルによって内容（値）が違うということが、問題になりました。これを防ぐためには、データを一元管理するしかありません。そのための仕組みとして、導入されたのがデータベースです。

データベース出現前には、データが部門や個人によって管理され、必要な内容だけが扱われていましたから、その内容は担当者レベルで把握可能な範囲でした。しかし、一元管理となると組織全体としての整理が必要です。このためには、「データモデル」や「正規化」といった考え方が必要になります。また、利用者が多くなり障害の影響なども大きくなったため、同時実行制御や回復処理などの DBMS による制御も必要になります。この分野からの出題は、こうしたデータの共有化のために必要な考え方や機能に、現在のデータベースの事実上の標準である関係データベースを扱うための SQL を加えた内容がほとんどです。この章では、これらの出題内容を「データモデル」、「正規化」、「SQL」、「DBMS の制御」の四つに分けて学習していきます。データベースのキーワードである「データの共有化」ということを念頭に置いて学習していきましょう。

6.4 DBMS の制御

▶▶▶ Explanation

ポイントの解説

(1) ACID 特性

トランザクションとは、本来、取引というような意味ですが、データベース処理では、1 件の入力データに対する一連の処理や、その範囲のことを指します。つまり、トランザクションとは、分けることのできない一連の処理単位のことです。次の四つの特性を備えていることが必要とされます。なお、この四つの特性をまとめて、それぞれの英語の頭文字を取り **ACID 特性** といいます。

①Atomicity (原子性)

トランザクションは、全ての処理が完了するか、何も行われなかったかのどちらかの状態で終了することで、All or Nothing の原則と呼ばれます。

②Consistency (一貫性)

処理の状態に関わらず、データベースの内容に矛盾がないことです。トランザクション実行中には、更新したデータ内容を、他のトランザクションや利用者から見えないようにします。

③Isolation (独立性)

トランザクションの処理結果は、他のトランザクション処理の影響を受けないことです。結果として、複数のトランザクションを同時実行させた場合と、順番に一つずつ実行した場合とで、処理結果が一致することになります。なお、処理結果が一致するとは、二つのトランザクション X、Y を実行した場合、実行順番が違う X→Y と Y→X のどちらかに一致するという意味です。

④Durability (耐久性)

トランザクションの実行が終了すれば、更新内容などの処理結果が、ソフトウェアやハードウェアのエラーによって損なわれることがないことです。

(2) トランザクション制御

ACID 特性のうち、耐久性については、トランザクション処理が終了してから発生する障害が原因となりますから、後述の回復処理で実現されます。他の三つ、つまり、原子性、一貫性、独立性については、トランザクション処理の実行中に行われる **排他制御** と **コミットメント制御** によって実現されます。そし

(10) 分散処理フレームワーク

① Apache Hadoop

ビッグデータを扱う処理は分散並列処理を行うものが大半です。それを実現するシステム技術である Apache Hadoop は、Apache Software Foundation がオープンソフトウェアとして公開している複数のソフトウェアを含むフレームワークです。Apache Hadoop は、バッチ処理向けに高いスループットを実現する分散ファイルシステムである HDFS (Hadoop Distributed File System) と、複数のサーバを用いて多量のデータを並列分散処理するプログラミングのためのソフトウェアである Hadoop MapReduce を中心に構成されています。HDFS は、ファイルのブロックを格納しクライアントとの受渡しを行うデータノードと、メタデータやデータノードを管理するネームノードからなります。障害に備えて、データノードは複数の複製を作成し、ネームノードはバックアップノードをもちます。Hadoop MapReduce は、Map 処理と Reduce 処理の2段階に分けて行います。Map 処理は、データを小さな単位に分けて相当数のサーバに割り振り、中間結果を出力します。Reduce 処理では、出力された中間結果を集約し、一塊のデータとして出力します。Hadoop MapReduce 以外の、リアルタイム処理向けの Apache Spark、ストリーム処理向けの Apache Storm など他の並列処理フレームワークも利用できます。

② Apache Spark

Apache Spark は、リアルタイム処理向けのフレームワークです。インメモリでデータを処理するため、MapReduce に比べて高速処理ができ、Java や Python などの言語に、多種類の API や目的別のライブラリーを提供してアプリケーションの開発を支援しています。

③ Apache Storm

Apache Storm は、無制限のストリーミングデータを処理するストリーミング処理のフレームワークです。リアルタイム解析、連続問合せ処理、オンライン機械学習などに利用できます。

(11) CAP 定理

2000 年に Eric Brewer が発表した、「分散システムにおいて C：整合性 (Consistency), A：可用性 (Availability), P：分断耐性 (Partition Tolerance) の3つのうち最大2つしか満たすことができない」という定理を **CAP 定理**と いいます。

- ・整合性：あるデータに対して更新処理がされた後、そのデータに対する全ての読出しが最新のデータを読み出すことができる→更新後の値が全ての複

CPUのリソースと消費電力を使い、高負荷な計算処理を行った仕事量によって正しいブロックの証明をします。一つのブロックに同時に複数のブロックの追加が行われたときは分岐が発生しますが、長く伸びるブロックチェーンが正しいものと判断されます。短いブロックチェーンは無効になり、取引が破棄される可能性があり、取引確定の判断がつかないことをファイナリティのない決済といいます。**ファイナリティ**とは、決済が最終的に完了することです。

・PBFT

コンソーシアム／プライベート型のように閉じたネットワークで、参加者が決まっており、悪意をもった参加者が入りにくいことが前提の場合のコンセンサスアルゴリズムの一つに、分散データベースでレプリケーションプロトコルとして提案されてきたPBFT (Practical Byzantine Fault Tolerance)があります。コンセンサスに参加するノードは、リーダとあらかじめ決められた検証するノードだけで、リーダから検証ノードへ提案し、正しいかどうか判断した検証ノードは投票します。投票結果によって正しいと認められればリーダが新しいブロックを追加します。PBFTにはブロックの分岐は発生せず、追加された時点で取引は確定するので、ファイナリティがあるといわれます。

▶▶▶ Check

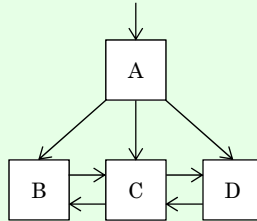
理解度チェック ▶ 6.4 DBMS の制御

次の文中の に適切な用語を入れてください。

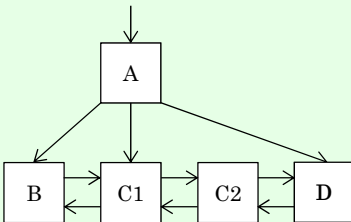
- (1) 同時に更新しなくてはならない表に対する更新など、分けることのできない処理のまとまりのことを **ア** といいます。また、このような性格をもった **ア** 処理に対しては、処理が全て正常に行われるか、全く行われないかのいずれかの状態であるという **イ**、処理の状態に関わらずデータベースの内容に矛盾がないという **ウ**、 **ア** の処理結果は、他のトランザクション処理の影響を受けないという **エ**、終了した **ア** 処理の結果が、障害などによって損なわれないことがないという **オ** という四つの特性が求められ、それぞれの頭文字を並べて **カ** 特性と呼ばれます。なお、この **カ** 特性のうち、 **ウ** と **エ** については、参照や更新に先立って表や行を **キ** する **ク** によって実現され、 **イ** については、複数の更新内容を一括してデータベースに反映する **ケ** によって実現されます。

問4 関係データベースのテーブルにレコードを1件追加したところ、インデックスとして使う、図のB⁺木のリーフノードCがノードC1とC2に分割された。ノード分割後のB⁺木構造はどれか。ここで、矢印はノードへのポインタとする。また、中間ノードAには十分な空きがあるものとする。

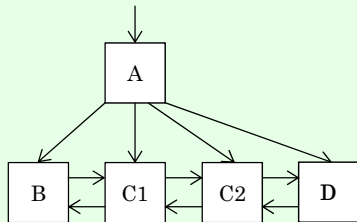
(H30 春-AP 問26)



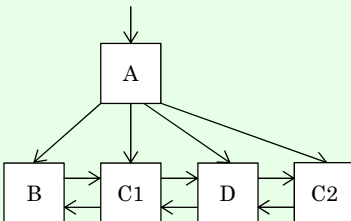
ア



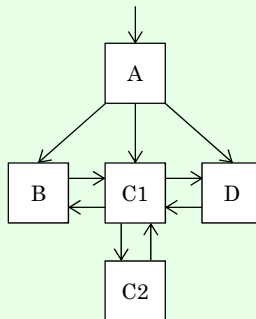
イ



ウ



エ



解説

B⁺木は、多くの関係データベースで利用されているインデックスのデータ構造で以下の性質をもちます。

- ①根ノードから葉ノードまでの木の深さは全て同じ
- ②根以外のノードは、 n 個以上 $2n$ 個以下の要素をもち、要素数+1個の子ノードをもつ