txt040313 プログラムテスト

① テスト

ⓐ テストの目的

テストの目的は作成したソフトウェア製品の品質を評価することである。開発したシステムやモジュールが設計仕様書どおりに動作するかなど、ソフトウェアの完成度を検証する一連の作業である。開発者とユーザがそれぞれの観点からテストを実施し、システムやモジュールの潜在的な欠陥を発見することでもある。計画、設計、製作の段階はトップダウンアプローチで進められるが、テスト段階以降はボトムアップアプローチの考え方が用いられる。

ⓑ テスト時の主要評価項目

㋐ 信頼性は要求仕様を満足し、ソフトウェアにエラーがなく、規則違反の使用に対して堅固であることである。

㋑ 操作性は入力しやすいこと、入出力情報やメッセージの内容が読みやすく、理解しやすいことである。

㋒ 性能はソフトウェアの処理能力や所要記憶容量が目標値であることである。

㋓ 保守性は修正・改造が容易であり、拡張性があることである。

② テストの種類

ⓐ テストの種類とその内容

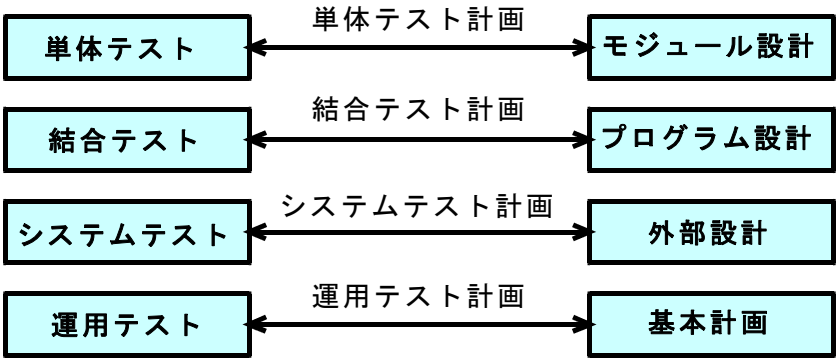
- 単体テスト（モジュールテスト：モジュール内のバグを見つけるテスト）
- 結合テスト
  - モジュール間インタフェーステスト
  - プログラムの入出力を含む一貫テスト
- システムテスト
  - プログラム間の結合テスト
  - サブシステム間の結合テスト
  - サブシステムおよび全体システムの入出力を含む一貫テスト
  - 例外事項テスト
  - 障害テスト
  - 処理速度テスト
- 運用テスト(承認テスト)

ⓑ　テストの順序

テストは、単体テスト、結合テスト、システムテスト、承認テスト、運用テストの順に進められる。単体テストからシステムテストはシステム開発者が主体で進める。承認テスト、運用テストはシステムのユーザが主体で進めるテストになる。

ⓒ　テスト計画の作成時期とテストの関係

テスト計画はテストの種類に応じて、計画、設計段階に作成される。

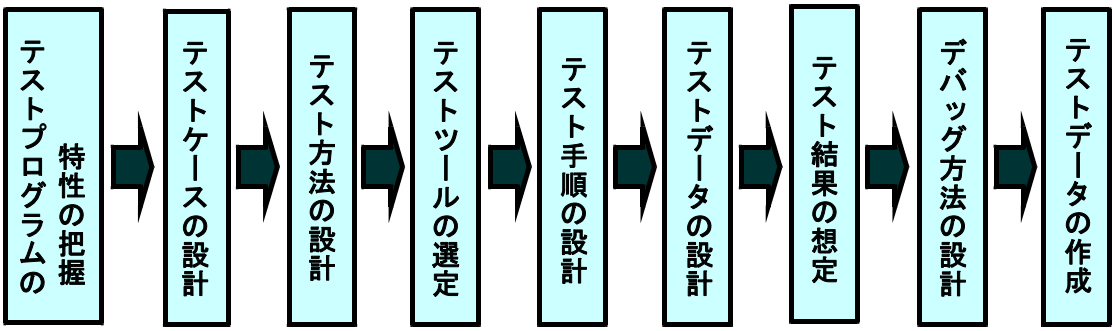


③　テスト設計

ⓐ　テスト設計とは

テストは限られた時間とコストの範囲内で実行する必要がある。単体テスト、結合テスト、システムテスト、運用テストなどのテストの種類に応じて、テストの目的、テストの方法を明確にし、テスト対象のプログラムの特性を吟味して、適切なテスト技法、テストツール、テストデータを選択・設計しなければならない。テストの種類やプログラムの特性に応じて、テスト方法やテスト環境が異なる。

ⓑ　テスト設計の手順



ⓒ　テスト対象プログラムの特性の把握

オンラインリアルタイム処理、バッチ処理など、プログラムの特性によって、テスト方法、テスト環境が異なる。オンラインリアルタイム処理では通信回線関係のハードウェアやソフトウェアの準備が必要であり、大規模システムではテストシミュレーションを含むテスト環境の整備が必要になる。テストに必要なハードウェア構成を検討するために、被テストプログラム、同時使用の共通プログラムを明確にし、テストに使用するライブラリの大きさを見積もる必要がある。

ⓓ　テスト設計の検討内容および留意点

㋐　テストケースの設計

テストの目的に合わせて、必要なテストケースを洗い出す。テストケースには、有効なテスト条件、システムに求められている機能、その機能の範囲外、エラーケースも含める。データの有効範囲と無効範囲の境界に関するエラーが多いので、境界に関するテストケースを必ず含める。

㋑　テスト方法の設計

テストケースの内容を吟味し、それぞれのテストケースを単独でテストするか、組み合わせてテストするかを決定する。テスト順序を決定する。

㋒　テストツールの選定

テストケースを効率よくテストするために必要なテストツールを選定する。テストドライバ、テストデータジェネレータ、スタブ、デバッガ等を選定する。

㋓　テスト手順の設計

テストケースを吟味して、テスト実行の手順を決める。テストの手順によって、効率が大きく異なる。結合テスト、システムテスト、運用テストの場合には熟慮が必要である。

㋔　テストデータの設計

テストの目的を明瞭にし、最も効率よく実行できるテストデータを設計する。少ないデータで効率よくテストするには、テストデータジェネレータで作成する方法、テストチームが作成する方法、ユーザが作成する方法のいずれかの方法を選択することが望ましい。負荷テストの場合は現在使用中のファイルから抽出する方法を採用することが多い。テストの目的に合わせたテストデータの作り方を選択し、テーストケースの選定は、モジュール設計書に基づいて，すべてのロジックパスを一度は通るようなテストケースによって検証を行う。

㋕　テスト結果の想定

テストに使用するファイルやデータベース、出力データについて、テスト前の内容とテスト後の内容を明確にする。

㊦　デバッグ方法の設計

デバッグ方法を検討し、効率よくバグを見つけることができるようにする。デバッグ方法もテストの種類に合わせて検討する。

㊧　テストデータの作成方法

❶　テストデータジェネレータで作成する。
❷　テストチームが作成する。
❸　ユーザが作成する。
❹　現在使用中のファイルから抽出する。
❺　現在使用中の帳票から抽出する。

ⓔ　テスト環境の検討内容および留意点

㋐　ハードウェア

テストの種類に対応させて、テストに必要なハードウェア構成を明確にする。必要な時期を明確にし、前もって準備する。

㋑　ソフトウェア

基本ソフトウェアはハードウェアと同様に、必要な時期を明確にして、前もって準備する。テスト支援ツールは必要なテスト支援ツールを明確にし、導入する。モジュールやプログラム単位のテストツールは、共通に使用できるツールの他に、スタブなどのように、モジュールやプログラム単位のテストツールが必要になる。

㋒　テスト体制

テストチームに対する教育、訓練を考慮して早めに手配する。ユーザの参画は、テストケースの設計、テストデータの作成、テストの実施、テストの結果の検証において適宜必要となる。システムテストの場合、ユーザの参画が重要である。テスト終了の承認手続きと承認者を明確にする。

㋓　消耗品の準備

テストに必要な用紙や事務用品等を準備する。

④　テスト支援ツール

ⓐ　テストデータジェネレータ

被テストプログラムが使う入力ファイルやデータベースをテストケースに合わせて作るときに使われる。作成した内容をリストする機能を持っている。プログラムテスト用のデータを簡

単なパラメータを与えることで自動的に生成する。正常なケースのデータ、異常なケースのデータ、エラーデータが生成できる。オンラインシステムで、対話型で作成できるものもある。

テストデータの作成方法は対話型でデータを１件ずつ作成する。ある条件を与えて（初期値と増分などの条件）、データの入力なしに、規則に従ってデータを大量に作る方法がある。作成したデータをファイルに格納するため、ファイルの定義や編成法、フィールドの定義、属性の定義を行っておく。データチェック機能はテストデータ作成時にデータの属性をチェックする。データの再作成は作成済みのデータに条件を与えることによって、目的にあったデータを再作成する。必要項目を指定することで、ファイルのレイアウトの変更も可能である。

ⓑ　カバレージモニタ

テスト対象プログラムのうち、既テスト分と未テスト分を識別するツールである。あるテストデータがプログラムのどの経路を通ったかを調べながら、プログラム全体の経路のうち約何％をカバーしたかを求めることができる。プログラムの経路は選択や繰返しなどの分岐のある文がでてくることによって増えていく。すべての経路をテストすることは不可能なので、カバレージを見ながら判断する。

ⓒ　ドライバとスタブ

㋐　ドライバ

ドライバはボトムアップテストで下位モジュールのテスト時に上位モジュールの代わりに用いるテストプログラムである。

㋑　スタブ

スタブはトップダウンテストで上位モジュールのテスト時に下位モジュールの代わりに用いるテストプログラムである。

ⓓ　デバッガ

デバッガはプログラムのバグを見つけるために使用するテストツールである。プログラム実行の中断、パフォーマンスの分析、実行の継続、コード部のスキップ、エラーの訂正、変数の表示及びセット、期待される入力の設定、出力の表示等の機能がある。

ⓔ　インスペクタ

インスペクタはプログラムを実行してエラー検出やデータ構造の内容を確認するためのデバッギングツールで、プログラムを途中で中断し、トレース対象データの閲覧や更新を対話形式で処理する。

ⓕ　トレーサ

トレーサはプログラムの実行時に制御の流れを追跡するときや実行過程を時系列的に１ステ

ップずつモニタリングするときに使用する。あるデータを処理するときのプログラムの動きを命令単位で調べられる。特定区間の命令を実行する毎に、その所在や命令自身、実行直後のレジスタなどの内容を書き出す。誤りの箇所が特定できない時に有効である。区間の指定を的確にしないと、出力量が膨大になり、処理時間がかかる。ワンステップずつ追跡調査できる。

ⓖ　ダンプツール

ダンプツールはダンプ命令を組み込んで、特定の領域をリストするツールである。特定の命令が実行される前後にダンプ命令を組み込んでプログラムの実行状況を把握するときに使用する。ダンプ命令には、そのプログラムを使っている全ての領域をダンプするものと、指定した部分だけダンプするものがあり、プログラムを実行しながら出力する動的ダンプとプログラムの実行終了後に出力する静的ダンプがある。

ⓗ　スナップショットダンプ

スナップショットダンプはプログラムにデバッグ命令を組み込んでおき、デバッグ命令を実行する都度、主記憶装置の一部やレジスタの内容を書き出す。指定した条件のときだけ主記憶やレジスタの内容などの追跡データを出力するダンプである。

動的ダンプのときに使われ、プログラムにエラーがあり原因不明で分からないとき、プログラムの特定の要所にダンプの条件を設定しておき、そのプログラムの記憶装置よりメモリの一部または全部の内容を出力して、エラーを調べることをいう。

プログラムの誤りの箇所がほぼ見当がついていて、どのような状態で異常が発生しているかを知るため、誤りと思われる命令群の処理前のデータの状態と処理後のデータの状態を比較するときに使用するデバッグツールである。

ⓘ　クロスリファレンス

クロスリファレンスはプログラムで使用している変数や関数の名前を相互参照できるデバッギングツールである。

ⓙ　プリティプリンタ

プリティプリンタは字下げや予約語のフォント変更などプログラムの整形機能をもつデバッギングツールである。

ⓚ　プルーフリスト

プルーフリストは入力したデータが誤りなく入力されているかをチェックするために、入力したデータをそのまま出力して正しく校正する目的で使用するリストである。

⑤　ユニットテスト（単体テスト）

ⓐ　ユニットテストとは

㋐　ユニットテスト

ユニットテストはプログラムの最小単位であるモジュールの品質をテストすることであり、その目的は結合テスト前にモジュール内のエラーを発見することである。テストは機能テストと構造テストの２つの観点から行う。モジュールはプログラムを構成する要素であるから、単体では動作しない。ドライバとスタブというテスト支援ツールを使用してテストを行う。

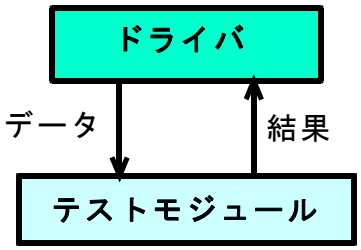
㋑　機能テスト

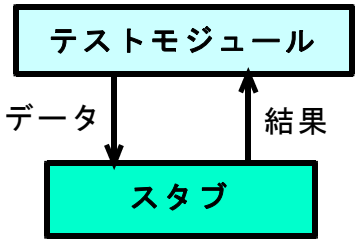
機能テストはモジュールの機能仕様をもとに、すべての入力条件や出力条件、エラー処理など、モジュールの機能が満足されているかどうかを検証する。機能テストは外部に見える機能の検証である。

㋒　構造テスト

構造テストはモジュールの詳細仕様や原始プログラムをもとに、モジュールの論理が正しいかどうかの検証を行う。プログラム内部の論理の検証である。

ⓑ　ドライバとスタブ





㋐　ドライバ

複数のソフトウェア部品（モジュール）の結合テストを行なう際に、呼び出し側のモジュールが未完成であったりテストのために実行するのが面倒だったりする場合に、その代用となるテスト用の呼び出しプログラムをテストドライバ、あるいは単にドライバという

ドライバはテストモジュールの上位モジュールの機能をシミュレートする。テストモジュールから見て主プログラムの役割を果たす。結合テストのボトムアップテストに利用される。

㋑　スタブ

モジュールの動作をテストする際、呼び出し先の下位モジュールの代わりをする空のモジュールのことをスタブという。下位モジュールが未完成なうちに上位モジュールのテストをしたい場合に用意される。スタブはテストモジュールの下位モジュールの機能をシミュレー

トするもので、テストモジュールから見て副プログラムの役割を果たす。結合テストのトップダウンテストに利用される。

スタブは下位モジュールと同じ名称や引数、返り値の型などを持ち、上位モジュールから同じコードで呼び出される。内部は空で何も処理を行わないが、本物と同じような値を返さなければならない場合には、想定される返り値の一つをあらかじめ算出しておき、これを定数として機械的に返却するといった処理を行う。

⑥　ブラックボックステスト

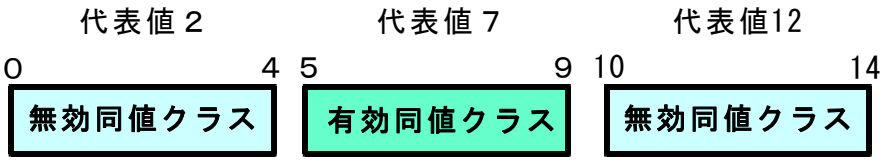
ⓐ　ブラックボックステストとは

ブラックボックステストはプログラムの外部仕様をもとにテストケースを設計するための技法で、プログラムの詳細なアルゴリズムの仕様は参照しないで、プログラムの機能仕様やインタフェース仕様だけを用いて設計する。テストケースの設計はエラーを検出する可能性の高いデータを効果的に組み合わせて、効率的にテストが行えるようにすることである。稼働後のシステムの品質はテスト時に使用したテストケースの良否に左右される。プログラムの設計内容から、プログラムの機能とデータの関係を考慮して、テストデータを作成し、プログラムのテストを行う。技法として、同値分割、限界値分析、因果グラフ、実験計画法などがある。

ⓑ　因果グラフ

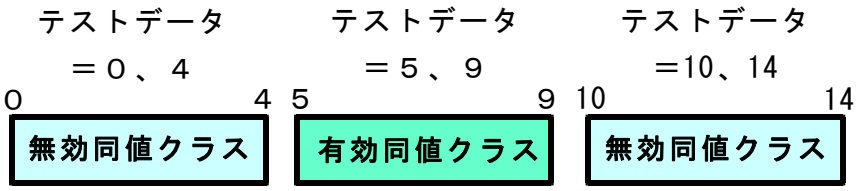
因果グラフは、テスト対象の入力が明確にクラス分けできないようなときに、入力・出力、原因・結果の関係をグラフで表現し、デシジョンテーブルに展開して、テスト項目を設計する。基本要素として、同値、否定、和、積、必要とする、排他的、包含するなどの論理関係を利用してグラフ展開する。

ⓒ　同値分割



同値分割は、テスト対象の入力データの取り得る値の範囲の中から、同じ意味を持つ範囲を１つのクラスとして、いくつかのクラスに分割する。分割したクラスの中から、各クラスを代表する値をテストデータとして選択する。クラス分割する場合、入力データの正しいものおよび誤っているものについてもいくつかのクラスに分割する。正しい範囲のテストデータを有効同値クラス、誤ったデータの範囲を無効同値クラスという。

ⓓ　限界値分析



限界値分析は、入力データ、出力データを同値クラスに分割し、それぞれのクラスの境界条件をテストの対象データになるように値を選ぶ方法である。同じクラス内の最大値または最小値、あるいは両方の値を選ぶ方法である。テストデータは入力条件だけでなく出力も意識して作成する必要がある。

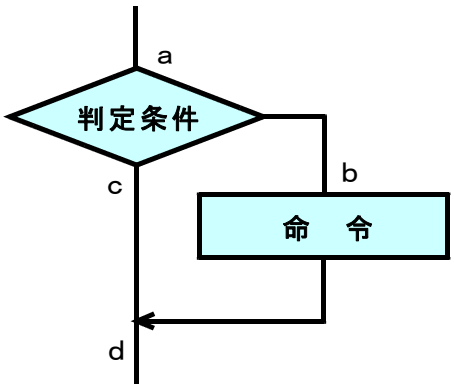
⑦　ホワイトボックステスト

ⓐ　ホワイトボックステストとは

ホワイトボックステストはプログラムの制御の流れに着目し、プログラムのステップの重要な部分を通るようなテストデータを作成し、テストする方法である。プログラムの内部仕様をもとにして、テストケースを設計する技法で、プログラムの内部構造や論理を詳細に調べるため、プログラマの立場から見た詳細な機能テストは行えるが、仕様にはあるがプログラムに実現されていない機能のエラーを発見できない問題がある。

規模の大きいプログラムでは、代表的な正常処理の経路と異常処理や例外処理の経路を中心にテストケースを設計する。すべてのステップを網羅するテストデータは膨大になるため、命令網羅、判定条件網羅、条件網羅、複数条件網羅などの簡略化した方法を利用してテストケースを設計する。

ⓑ　命令網羅



命令網羅は条件式の真偽に関係なく、すべての命令を少なくとも１回は実行するようにテストケースを設計する。繰り返しの対象のブロック内の命令も最低１回は実行させる。流れ図で

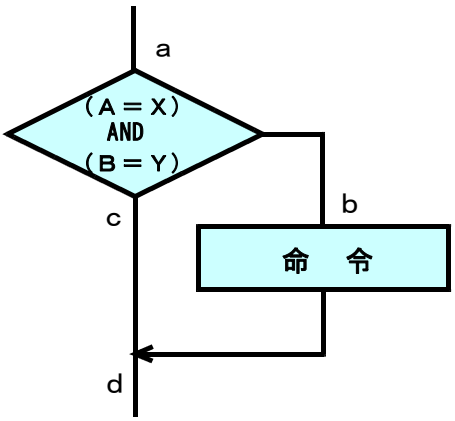
ａ→ｂ→ｄの経路のテストを実行すれば、命令を網羅したことになる。

ⓒ　判定条件網羅

判定条件網羅は条件式の真偽ではなく、判定結果の真偽または判定結果の種類の数の経路を網羅し、かつすべての命令を少なくとも１回は実行するようにテストケースを設計する。流れ図で、ａ→ｂ→ｄ、ａ→ｃ→ｄの２つの経路のテストを実行すれば、命令を網羅したことになる。

ⓓ　条件網羅

条件網羅は、判定結果ではなくすべての条件式において真偽を判定し、かつすべての命令を１回は実行するようにテストケースを設計する。流れ図で、判定条件(Ａ＝Ｘ)AND(Ｂ＝Ｙ)が真の場合と偽の場合の２通りの組み合わせをテストする。真の場合は、(Ａ＝Ｘ)AND(Ｂ＝Ｙ)が成り立つ場合であるが、偽の場合は、(Ａ≠Ｘ)AND(Ｂ＝Ｙ)、(Ａ＝Ｘ)AND(Ｂ≠Ｙ)、(Ａ≠Ｘ)AND(Ｂ≠Ｙ)の３ケースあるが、このうち条件式が偽となる１ケースについて実行すればよいことになる。



ⓔ　複数条件網羅

複数条件網羅は、それぞれの判定における条件付きの可能なすべての組み合わせ、かつすべての命令を少なくとも１回は実行するようにテストケースを設計する。流れ図において、判定条件は、真の場合は、(Ａ＝Ｘ)AND(Ｂ＝Ｙ)が成り立ち、偽の場合は、(Ａ≠Ｘ)AND(Ｂ＝Ｙ)、(Ａ＝Ｘ)AND(Ｂ≠Ｙ)、(Ａ≠Ｘ)AND(Ｂ≠Ｙ)の３ケースが成り立つ。これらの４ケースすべてについてテストする。

⑧　結合テスト

ⓐ　結合テストとは

結合テストはモジュールを結合したテストで、モジュール間のインタフェースの検証とプロ

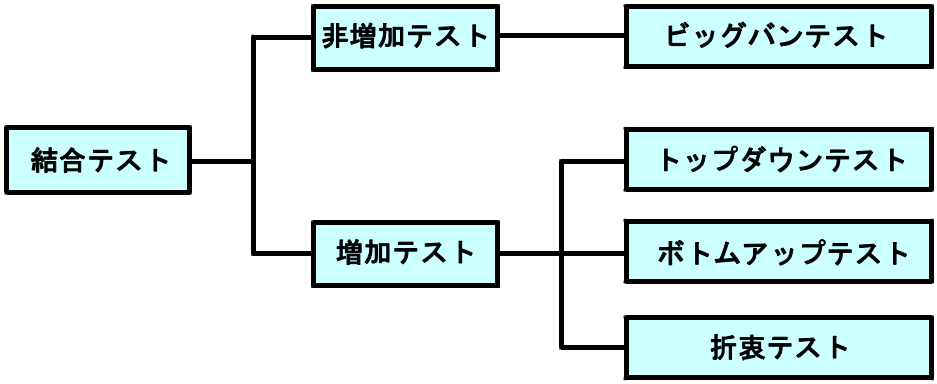
グラムの入出力を含むテストである。プログラムが外部仕様で定められた機能どおりに、実現されているかを検証する。信頼性や操作性、保守性などの検証も行う。

ⓑ　結合テストの手法

結合テストの手法には、すべてのモジュールを一斉に結合して行う非増加テストのビッグバンテストとテストの進行につれて結合するモジュールを逐次増加させながら行う増加テストの方法がある。

増加テストには、上位のモジュールからスタートして、逐次下位モジュールを結合しながらテストを進めるトップダウンテストと、逆に、下位モジュールからスタートして、逐次上位モジュールを結合しながらテストを進めるボトムアップテストがある。また、トップダウンテストとボトムアップテストの両方の手法を組み合わせてテストする折衷テストがある。

どの方式を用いてテストを行うかは、開発しているソフトウェアの特徴や要求される信頼性、開発の進捗状況など各種の条件を考慮して決定する。



⑨　ビッグバンテスト

ⓐ　ビッグバンテストとは

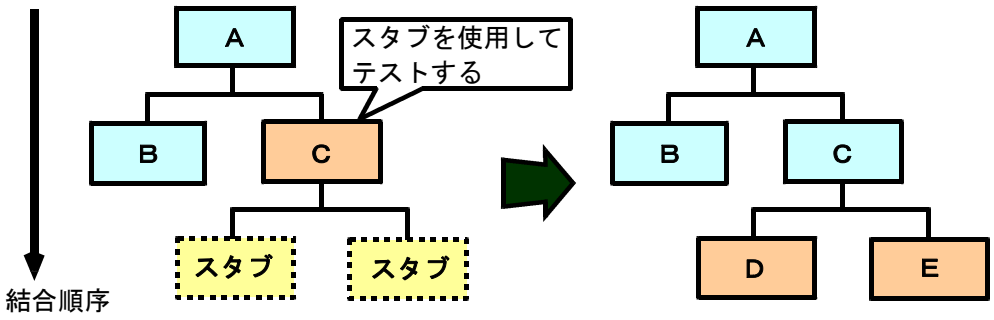
ビッグバンテストはモジュール単体毎にドライバまたはスタブを用意し、すべてのモジュールテストを行い、最後に、各モジュールを結合して一斉にプログラムテストを行う方法である。ドライバやスタブの作成量が多いため、経済性の面で増加テストより劣る。モジュールテストが十分に行われるため、信頼性は高い。

ⓑ　ビッグバンテストの特徴

㋐　結合テスト時に、ドライバやスタブが不要である。
㋑　骨組みになる部分も含めて、すべてのモジュールが、最初からテストされる。
㋒　特定の経路をテストすることが難しい。
㋓　エラーが発生したときのデバッグが容易でない。
㋔　単体テストが終了していないモジュールが１個でもあると、結合テストを開始できない。

⑩　トップダウンテスト

ⓐ　トップダウンテストとは



トップダウンテストは最上位のモジュールからモジュールテストを行う方式で、順次、下位モジュールを結合しながらモジュールテストを繰り返す。上位モジュールからテストを行うため、下位モジュールが未完の場合、スタブが必要である。スタブは、上位モジュールから制御をもらい、引数を設定してから制御を戻す簡単な機能を持つ。トップダウンテストはモジュール間インタフェースを仮定する必要が無く、引数の受渡をテストすることができる。最上位モジュールは最初からテストが繰り返されており、バグの発見も効率よく行え、プログラム全体の制御を司る重要な部分がきちんとテストできる。

ⓑ　トップダウンテストの特徴

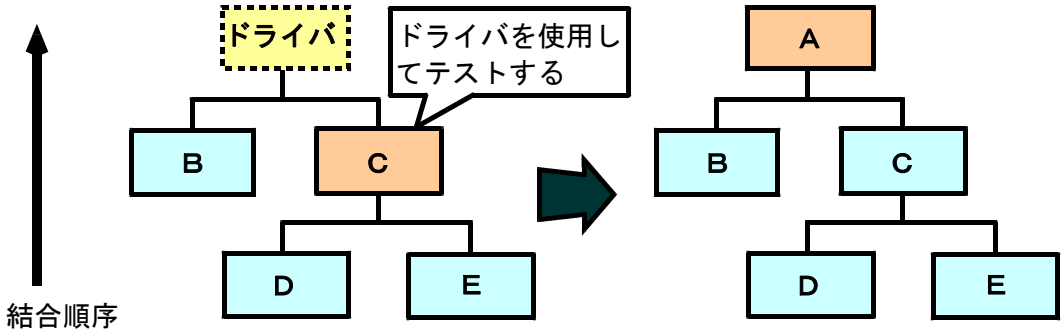
㋐　最初は並行作業が困難であり、テストするまで時間が空いてしまう可能性がある。
㋑　上位のインタフェースが早い時期にテストできる。
㋒　重要度の高い上位モジュールが繰返し実行されるので、信頼性が高い。
㋓　モジュール単体の機能や、個々の論理を十分にテストできないことがある。
㋔　一般に、新規開発システムに適用すると効果がある。
㋕　トップダウンでモジュールの集積を進めるため、並行した開発がし難い。
㋖　全体としては工数が多くなる。

⑪　ボトムアップテスト

ⓐ　ボトムアップテストとは

ボトムアップテストはプログラム構造の最下位レベルのモジュールを最初にテストし、次に、それらの上位レベルのモジュールを結合してテストする。これを繰り返しながら最後に最上位のモジュールを結合してテストを行う。上位モジュールが未完の場合、テストモジュールの上位モジュールの機能をシミュレートするドライバが必要である。ドライバは、下位モジュール

に制御と引数を渡すとともに、下位モジュールから戻ってきた引数をもとに、対応する操作を実行する。上位モジュールの役割を代行する機能を持つ。インタフェーステストを何回も行う場合、繰り返し下位モジュールを呼び出せる制御構造を持つ必要があり、ドライバの作成が煩雑になる。



ⓑ　ボトムアップテストの特徴

㋐　開発の初期の段階から並行作業が可能である。
㋑　モジュール単体の機能や論理が十分にテストできる。
㋒　テストの最終段階でインタフェース上の問題が発生しやすい。
㋓　全体の機能を把握するのに時間がかかる。
㋔　既に稼働しているシステムを修正して、システムを開発する場合に有効である。
㋕　問題は、モジュール間のインタフェースについてすべてを結合しない限り、全体の整合性をテストできない点にある。
㋖　最上位モジュールが最後に結合されるため、十分にテストできない。

⑫　折衷テスト

ⓐ　折衷テストとは

折衷テストはトップダウンテストとボトムアップテストを組み合わせてテストを行う方法である。最上位モジュールはトップダウンテスト、最下位に近いモジュールはボトムアップテストを用いる。トップダウンテストのモジュール間のインタフェース活用の利点とボトムアップテストの並行テストの実施の利点を組み合わせて行える。

ⓑ　折衷テストテストの特徴

㋐　並行してテストできる程度が高い。
㋑　骨組みになる部分のテストが早くからできる。
㋒　テストの計画及び管理がやりやすい。
㋓　特定の経路のテストが容易である。

㋓　ドライバ、スタブの両方を用意する必要がある。

㋔　最上位のモジュールと最下位のモジュールの両方からテストを進めることが可能で、中間に位置するモジュールが最後にテストされる。

㋕　全ての下位モジュールをスタブとしてあらかじめ作成しておくことができる。

⑬　システムテスト

ⓐ　システムテストとは

システムテストはソフトウェアやシステムが要求された仕様に合致しているかどうかを検証するために、複数の機能グループを組み合わせて行うテストである。

ⓑ　システムテストの内容

㋐　プログラム間やサブシステム間の結合テスト

㋑　サブシステム・システムの入出力を含むテスト

㋒　例外データを与えたときに正しく例外処理を行うことを確認する例外事項のテスト

㋓　障害テスト

㋔　性能テスト(スループット、レスポンスタイム、ターンアラウンドタイム)、負荷テスト

㋕　機能テスト

ⓒ　主要システムテストの検討内容

㋐　性能テスト

スループット、レスポンスタイム、ターンアラウンドタイムなどを評価する。スループットは単位時間当たりの処理する仕事の量で、同時に稼働している端末数やマルチプログラミングの性能によって変化するため、各種条件での検証が必要になる。レスポンスタイムも同時に稼働している端末装置の台数によって異なるため、各種条件での検証が必要になる。

㋑　負荷テスト

短時間に重い負荷をかけるテストである。同時に実行するプログラム数や同時に稼働する端末台数を増加させて、レスポンスタイムなどをテストする。システムが負荷に耐えられるかどうかをテストする。

㋒　機能テスト

ユーザの要求仕様を満足しているかどうかをテストする。ユーザの要求仕様の機能が詳細に記述されていない場合には、設計書の中から要求仕様を取り出す作業が必要になる。

⑭　運用テスト

ⓐ　運用テストとは

運用テストはユーザ部門の運用グループが実際の運用と同一の条件を作り出し、機能面および操作面の双方のテストを行う。承認テスト、導入テスト、フィールドテストなどがある。ある程度の期間を費やし、担当者がシステムの操作に習熟する研修を兼ねて行われることもある。

運用テストでは、本番移行基準の確認、移行テスト、保守性テスト、運用効率や業務効率の測定、ユーザビリティテストなどが行われる。移行テストでは、安全性・効率性の観点で、既存システムから新システムへの切り替え手順、切り替えに伴う問題点を確認する。

仕様書通りの論理が実装されていることを確認するだけではなく、操作に対する応答時間や単位時間当たりの処理性能を計測したり、高い負荷をかけたときの反応や耐久性を見たり、入力ミスや誤操作、ハードウェア障害などを故意に発生させてエラー処理や復旧などの手順を確認したりする場合もある。

ⓑ　承認テスト

開発部門がユーザ部門の承認を得るテストである。ユーザ部門がデータを用意し、要求仕様どおりの結果が得られるかどうかをユーザ部門に承認してもらうために行う。ユーザ部門が管理、実行する。

ⓒ　導入テスト

システムが実際の動作環境で、誤りなく稼働するかどうかを確認するテストで、実際の業務に即した利用の仕方をしてみて問題なく動作するかを試すテストである。承認テスト、検収テストを兼ねる場合もある。

ⓓ　フィールドテスト

フィールドテストは特定の利用者に実際にシステムを使ってもらう実地テストである。開発した製品を、開発部門以外の部門で、製品の仕様書と照らし合わせて評価・確認するテストである。特定の利用者に、実際の仕事に使ってもらう実地テストである。

⑮　その他のテスト

ⓐ　移行テスト

移行テストは、現行システムから新システムへ、ハードウェアやソフトウェア、通信回線、各種ファイルを円滑に移し変えるために、移行方法や移行手順、移行体制、移行日程計画、移行タイムチャート、移行対象データ項目、データの移行方法、ＯＳやミドルウェアの入替などの問題点を検討するために行うテストである。

ⓑ　コードインスペクション

コードインスペクションは机上でプログラムを詳細に読んで検査することをいう。エラーをタイプ別に分類、集計し、次に発生する同種のエラーの発見を容易にするために利用する。ＯＳのように、高い信頼性を要求されるソフトウェアの開発に利用する。

ⓒ　リグレッションテスト（退行テスト）

レグレッションテストはシステムの保守段階に行うテストで、プログラムの変更により、既存の正しい範囲に新しい誤りが発生しないかどうかを検証する。コンピュータプログラムに手を加えたことによる影響を確認するテスト操作である。

プログラムが大規模化で複雑化すると、何も関係がないかのように見えるプログラムが相互に関係しあっているのを見落とす場合も少なくない。ある箇所を改善しようとして加えた修正が、思いもよらない部分に影響してバグを呼び起こしてしまう。

ⓓ　ペネトレーションテスト

ペネトレーションテストは、コンピュータやネットワークのセキュリティ上の弱点を発見するテスト手法の一つで、システムを実際に攻撃して侵入を試みる手法である。

ネットワーク接続された情報システムが外部からの攻撃に対して安全かどうか、実際に攻撃手法を試しながら安全性の検証を行う。不正に侵入できるかどうかだけでなく、ＤｏＳ攻撃にどれくらい耐えられるかを調べたり、侵入された際にそこを踏み台にして他のネットワークを攻撃できるかどうかなどを調べる場合もある。

例題演習

プログラムのテストの目的として，最も重要なものはどれか。

ア　バグがないことを示すこと　　イ　バグの原因を究明すること
ウ　バグを修正すること　　エ　バグを見つけること

解答解説

テストの目的に関する問題である。

アのバグがないことを示すことは、テストの結果から諸条件を考えて論理的に展開することは可能であるが、現状ではバグが０であると断定することは難しい。テストを実施することの目的とは異なる。

イのバグの原因追及は、バッグの再発生を防止するために考える内容であり、テスト結果に基づいて次のステップで考える内容である。

ウのバグの修正は、テストの結果に基づいて、改善の対策を考える段階の問題であり、テストの直接の目的ではない。

エのバグを見つけることがテストの最も重要な目的である。求める答えはエとなる。

例題演習

プログラムモジュールの単体テストに関して、正しい記述はどれか。

ア　トップダウンテストでは、テスト対象のプログラムモジュールが呼び出す下位モジュールの代わりをするスタブが必要である。

イ　入力条件のテストでは、プログラム設計で規定された最大値・最小値のケースが重要であり、明らかに誤った条件の入力ケースを実施する必要がない。

ウ　プログラムモジュール１本ごとの論理上の正しさを証明するものであるから、コンパイルでエラーが発生しなければ単体テスト完了とする。

エ　プログラムモジュールのコーディングが全て完了していなくても、単体テストを開始することができる。

解答解説

単体テストに関する問題である。

アのトップダウンテストでは下位モジュールの代わりをするスタブが必要であり、アの記述は正しい。求める答えはアとなる。

イの入力条件に関するテストはモジュールの機能仕様に基づいて、すべての入力条件、出力条件、エラー処理等、モジュールの持つ機能が満足しているかどうかを検証する。従って、誤った条件の入力ケースを実施する必要がないという記述は誤りである。

ウは、プログラム内部の論理の検証が必要である。コンパイルでエラーが発生しないだけでは論理上十分であるとはいえない。

エは、コーディングが完了していないとコンパイルできないため単体テストができない。

例題演習

入力データと出力結果の関係に注目してテストデータを作成し、プログラムの機能をテストする手法はどれか。

ア　トップダウンテスト　　イ　ブラックボックステスト

ウ　ボトムアップテスト　　エ　ホワイトボックステスト

解答解説

プログラムの機能をテストするブラックボックステストに関する問題である。

アのトップダウンテストは、上位のモジュールから下位のモジュールへと順次結合して行う結合テストである。

イのブラックボックステストは、モジュールをブラックボックスと見なし、機能仕様書に基づき作成したテストデータでテストを行う手法である。入力データと出力結果の関係に注目して、プログラムの機能をテストするものである。求める答えはイである。

ウのボトムアップテストは、下位のモジュールから上位のモジュールへと順次結合して行う結合テストである。

エのホワイトボックステストは、モジュールの制御構造を詳細に検討するテストである。

例題演習

ソフトウェア開発におけるテスト技法のうち，ブラックボックステストに関する記述として，適切なものはどれか。

ア　原始プログラムを解析し，プログラムの制御の流れと変数などのデータの流れをテストするものであり，主にプログラム開発者以外の第三者が実施する。
イ　ブログラムが設計者の意図した機能を実現しているかどうかのテストであり，主にプログラム開発者以外の第三者が実施する。
ウ　プログラムのすべての命令が最低１回は実行されることを目的とするテストであり，主にプログラム開発者自身が実施する。
エ　プログラムの内部構造や論理が記述された内部仕様書に基づくテストであり，主にプログラム開発者自身が実施する。

解答解説

ブラックボックステストに関する問題である。

ブラックボックステストは、プログラムの外部仕様をもとにテストケースを設計するための技法で、プログラムの詳細なアルゴリズムの仕様は参照しないで、プログラムの機能仕様やインタフェース仕様だけを用いて設計する。プログラムの設計内容から、プログラムの機能とデータの関係を考慮して、テストデータを作成し、プログラムのテストを行う。技法として、同値分割、限界値分析、因果グラフ、実験計画法などがある。

アは机上デバック、イはブラックボックステスト、ウ、エはホワイトボックステストの考え方である。求める答えはイとなる。

例題演習

プログラムテストにおける限界値分析で設定するテストデータとして、適切なものはどれか。ここで、“Ａの直前の値”とは“Ａより小さくてＡに近い値”を指し、“Ａの直後の値”とは“Ａより大きくてＡに近い値”を指す。

ア　最小値、最小値の直後の値、最大値の直前の値、最大値
イ　最小値、最大値
ウ　最小値の直前の値、最小値、最大値、最大値の直後の値
エ　最小値の直前の値、最小値の直後の値、最大値の直前の値、最大値の直後の値

解答解説

限界値分析におけるテストデータに関する問題である。

限界値分析は、ブラックボックステスト法の一手法であり、入力データと出力データを同値クラスに分割し、それぞれのクラスの端がテストの対象になるように値を選ぶ方法である。通常は有効同値クラスの最大・最小とそれぞれを一つ超えた値を用いる。従って、最小値の直前の値、最小値、最大値、最大値の直後の値となる。求める答えはウとなる。

例題演習

表は，あるプログラムの入力データを，有効同値クラスと無効同値クラスに分けたものである。同値分割法によってテストケースを設計する場合，最小限のテストデータの組合せとして，適切なものはどれか。

ア　－２，０，１，５，６，８
イ　０，１，５，６
ウ　－１，３，６
エ　１，５

同値クラス	データ
無効同値クラス	−2, −1, 0
有効同値クラス	1, 2, 3, 4, 5
無効同値クラス	6, 7, 8

解答解説

同値分割法に関する問題である。

同値分割法は、テスト対象の入力データの取り得る値の範囲の中から、同じ意味を持つ範囲を１つのクラスとして、いくつかのクラスに分割する。分割したクラスの中から、各クラスを代表する値をテストデータとして選択する。

無効同値クラス－２～０、６～８の中からそれぞれ１つずつ選択し、有効同値クラス１～５の中から１つ選択しテストデータとする。－１、３、６のテストデータの組み合わせが適切である。求める答えがウとなる。

例題演習

ホワイトボックステストのテストデータの作成方法に関する記述として、適切なものはどれか。

ア　同値分割の技法を使用してテストデータを作成する。
イ　プログラムの外部仕様に基づいてテストデータを作成する。
ウ　プログラムの内部構造に基づいてテストデータを作成する。
エ　プログラムの入力と出力の関係からテストデータを作成する。

解答解説

ホワイトボックステストに関する問題である。

ホワイトボックステストは、プログラムの制御の流れに着目し、プログラムのステップの重要な部分を通るようなテストデータを作成し、テストする方法である。プログラムの内部構造や論理を詳細に調べるため、プログラマの立場から見た詳細な機能テストは行えるが、仕様にはあるがプログラムに実現されていない機能のエラーを発見できない問題がある。規模の大きいプログラムでは、代表的な正常処理の経路と異常処理や例外処理の経路を中心にテストケースを設計する。すべてのステップを網羅するテストデータは膨大になるため、命令網羅、判定条件網羅、条件網羅、複数条件網羅などの簡略化した方法を利用してテストケースを設計する。

ア、イ、エはブラックボックステストに関する内容であり、ウがホワイトボックステストに関する内容である。求める答えはウとなる。

例題演習

ホワイトボックス法に属するテストケースの作成方法はどれか。

ア　原因－結果グラフ　　イ　限界値分析

ウ　条件網羅　　エ　同値分割

解答解説

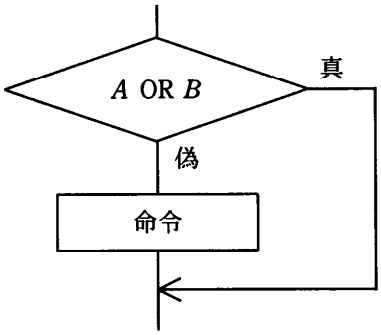
ホワイトボックステストのテストケースに関する問題である。

ホワイトボックステストは、ログラムのテストデータ選択方法の１つで、プログラムのモジュールの実行経路を詳細に確認するためにテストデータを選択する。すべての命令をテストすることは難しく、要求されている機能の確認も見出しにくい問題がある。テストデータの作成方法に、命令網羅、判定条件網羅、条件網羅、複数条件網羅等がある。

ア、イ、エはブラックボックス法のテストケース作成方法で、ウの条件網羅がホワイトボックス法のテストケース作成方法である。求める答えはウとなる。

例題演習

図の論理を判定条件網羅（分岐網羅）でテストするときのテストケースとして，適切なものはどれか。



ア

A	B
偽	真

イ

A	B
偽	真
真	偽

ウ

A	B
偽	偽
真	真

エ

A	B
偽	真
真	偽
真	真

解答解説

ホワイトボックステストに関する問題である。

判定条件網羅は、プログラムの全ての判定条件で、真と偽を少なくとも１回以上実行するようにテストケースを設計する。

アの場合は真の場合のテストのみである。

イの場合は、Ａが偽Ｂが真の場合も、Ｂが偽Ａが真の場合も共にＯＲは真であるから、真の場合のテストのみである。

ウの場合は、Ａが偽Ｂが偽の場合はＯＲは偽、Ａが真Ｂが真の場合はＯＲは真であるから、真と偽を少なくとも１回実行していることになる。真、偽を少なくても１回行うテストの判定条件網羅はウとなる。求める答えはウとなる。

エの場合は３回のテストは全て真の場合である。

例題演習

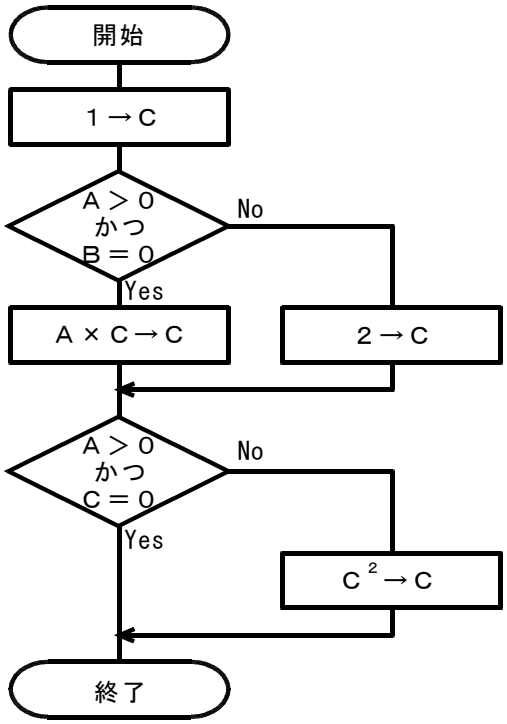
次の流れ図において、判定条件網羅（分岐網羅）を満たす最少のテストケースはどれか。

ア　（１）Ａ＝０，Ｂ＝０
　　（２）Ａ＝１，Ｂ＝１

イ　（１）Ａ＝１，Ｂ＝０
　　（２）Ａ＝１，Ｂ＝１

ウ　（１）Ａ＝０，Ｂ＝０
　　（２）Ａ＝１，Ｂ＝１
　　（３）Ａ＝１，Ｂ＝０

エ　（１）Ａ＝０，Ｂ＝０
　　（２）Ａ＝０，Ｂ＝１
　　（３）Ａ＝１，Ｂ＝０
　　（４）Ａ＝１，Ｂ＝１



解答解説

ホワイトボックステストの判定条件網羅に関する問題である。

判定条件網羅は、プログラムのすべての判定条件で真と偽を少なくとも１回以上実行するようにテストケースを設計する。

この流れ図では次の条件の内容に分けることができる。

①　最初の判定条件の真の場合はＡ＝１かつＢ＝０

②　最初の判定条件の偽の場合はＡ＝１かつＢ＝１、Ａ＝０かつＢ＝１、Ａ＝０かつＢ＝０のいずれか。

③　二つ目の判定条件の真の場合はＡ＝１かつＣ＝０

④　二つ目の判定条件の偽の場合はＡ＝１かつＣ＝１、Ａ＝０かつＣ＝１、Ａ＝０かつＣ＝０のいずれか。

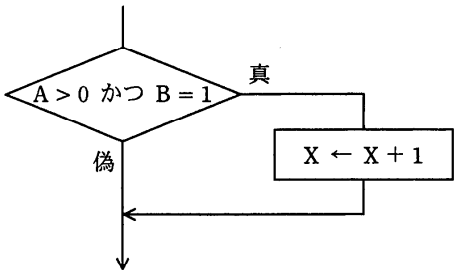
この流れ図ではＣ＝１であるから、２番目の判定条件はすべて偽となる。従って、判定条件網羅では最初の判定条件の真の場合と偽の場合の内の１つを実行すればテスト条件は十分になる。

肯定のＡ＞０かつＢ＝０はＡ＝１かつＢ＝０で１ケース、否定はＡ＝１かつＢ＝１で否定の条件判定を行う。求める答えはイとなる。

例題演習

図の構造をもつプログラムに対して，ホワイトボックステストのテストケースを設計するとき，少なくとも実施しなければならないテストケース数が最大になるテスト技法はどれか。

ア　条件網羅
イ　判定条件網羅
ウ　複数条件網羅
エ　命令網羅



解答解説

ホワイトボックステストのテストケースに関する問題である。

アの条件網羅は、処理の分岐条件で条件式に着目し、真偽について少なくとも１回は実行するようにテストケースを作成する。判定結果よりも条件式に着目する。最低２回テストする。

イの判定条件網羅は、分岐条件の判定結果である真偽の両方の判定を少なくとも１回実行するようにテストケースを設計する。最低２回テストする。

ウの複数条件網羅は、分岐条件の判定結果の真偽を通過するあらゆる組合せを網羅するテストケースを作成する。最低４回テストする。

エの命令網羅は、プログラム中のすべての命令を１回は実行するようにテストケースを作成する。最低１回はテストする。

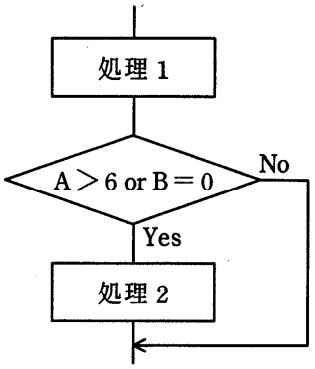
テストケースが最大になるのは複数条件網羅である。求める答えはウとなる。

例題演習

プログラムの流れ図で示される部分に関するテストデータを，判定条件網羅（分岐網羅）によって設定した。このテストデータを複数条件網羅による設定に変更したとき，加えるべきテストデータのうち，適切なものはどれか。ここで，（　）で囲んだ部分は，一組のテストデータを表すものとする。

・判定条件網羅（分岐網羅）によるテストデータ
　（Ａ＝４，Ｂ＝１），（Ａ＝５，Ｂ＝０）

ア　（Ａ＝３，Ｂ＝０），（Ａ＝７，Ｂ＝２）
イ　（Ａ＝３，Ｂ＝２），（Ａ＝８，Ｂ＝０）
ウ　（Ａ＝４，Ｂ＝０），（Ａ＝８，Ｂ＝０）
エ　（Ａ＝７，Ｂ＝０），（Ａ＝８，Ｂ＝２）



解答解説

ホワイトボックステストの条件網羅に関する問題である。

判定条件網羅の場合、（Ａ＝４、Ｂ＝１）はＡ、Ｂ共に偽で、流れ図のＮｏの処理になる。（Ａ＝５、Ｂ＝０）はＡは偽、Ｂは真であり、ＯＲであるから流れ図の処理はＹｅｓになる。

複数条件網羅にする場合、Ａ、Ｂ共に真、Ａは真、Ｂは偽の２つのテストケースを追加する必要がある。

アの場合、（Ａ＝３，Ｂ＝０）はＡは偽、Ｂは真であり、（Ａ＝７，Ｂ＝２）はＡは真、Ｂは偽となる。

イの場合、（Ａ＝３，Ｂ＝２）はＡは偽、Ｂは偽であり、（Ａ＝８，Ｂ＝０）はＡは真、Ｂは真となる。

ウの場合、（Ａ＝４，Ｂ＝０）はＡは偽、Ｂは真であり、（Ａ＝８，Ｂ＝０）はＡは真、Ｂは真となる。

エの場合、（Ａ＝７，Ｂ＝０）はＡは真、Ｂは真であり、（Ａ＝８，Ｂ＝２）はＡは真、Ｂは偽となる。求める答えはエとなる。

例題演習

プログラム中に次の複合判定がある。

条件１　ＯＲ　（条件２　ＡＮＤ　条件３）

判定条件網羅（分岐網羅）に基づいてテストする場合，追加するテスト項目として，適切なものはどれか。

〔終了したテスト項目〕

（１）　条件１が真，条件２が偽，条件３が偽

（２）　条件１が偽，条件２が真，条件３が真

	条件１	条件２	条件３
ア	偽	偽	真
イ	真	偽	真
ウ	真	真	偽
エ	真	真	真

解答解説

判定条件網羅に関する関する問題である。

判定条件網羅は、判定結果が真または偽となるすべての経路を網羅し、かつすべての命令を少なくとも１回は実行するようにテストケースを設計する。

このテスト内容で判定条件網羅の条件を満たすのは、真となるのは次の３つの内の１つが成り立てばよい。

１．条件１または条件２と条件３が同時に真となる。

２．条件１が真となる。

３．条件２と条件３が同時に真となる。

偽となる条件は次のいずれかが成り立てばよいことになる。

４．条件１が偽、条件２または条件３のどちらかが偽になる。

５．条件１、条件２、条件３いずれもが偽になる。

終了したテスト項目では(1)が2の場合、(2)が3の場合で、偽となる条件のテストが行われていない。従って、条件1が偽、条件2が偽、条件3が真のテストが必要である。求める答えはアとなる。

例題演習

プログラムの構造や制御の流れに着目し、プログラム内のすべての経路を網羅するようなテストを行うのはどれか。

ア　トップダウンテスト　　イ　ブラックボックステスト
ウ　ボトムアップテスト　　エ　ホワイトボックステスト

解答解説

ホワイトボックステストに関する問題である。

アのトップダウンテストは、上位のモジュールから下位のモジュールへと順次結合して行う結合テストである。

イのブラックボックステストは、モジュールをブラックボックスと見なし、機能仕様書に基づき作成したテストデータでテストを行う手法である。

ウのボトムアップテストは、下位のモジュールから上位のモジュールへと順次結合して行う結合テストである。

エのホワイトボックステストは、モジュールの制御構造を詳細に検討するテストである。

プログラムの構造や制御の流れに着目し、プログラム内のすべての経路を網羅するテストはホワイトボックステストで、求める答えはエとなる。

例題演習

モジュール単体テストに関する記述として，最も適切なものはどれか。

ア　通常はコーディングを行ったプログラマではなく，専任のテスト要員がテストケースを作成し，実行する。
イ　モジュール間インタフェースは，モジュール単体ではテストできないので，単体テストの対象外となる。
ウ　モジュール設計書は，正しいことが検証済みであるので，テスト結果に問題があるときは，テストケース又はモジュールに誤りがある。
エ　モジュール設計書を見ながら，原則としてすべてのロジックパスを一度は通るようなテストケースによって，検証を行う。

解答解説

モジュールテストに関する問題である。

アのテストケースの作成は、モジュール設計段階に行うため、プログラマが行う。テストの内容によってユーザが参画することがある。テストデータはテストチームやユーザが作成する。

イのモジュール間インタフェースは、スタブまたはドライバを使用して、行うことができる。

ただし、スタブまたはドライバを使用するので、代替モジュールの結果が適切とはいえない。

ウのモジュール設計書が正しくて、テスト結果に問題がある場合、テストケースやモジュールに誤りがなくても、テストデータやテストの方法、テストツールなどに問題があると、テスト結果に問題が発生する。

エの原則としてすべてのロジックパスを一度通るテストケースで検証を行う記述は適切である。求める答えはエとなる。

例題演習

デバッグツールとして用いるトレーサの説明として、適切なものはどれか。

ア　磁気テープファイルや磁気ディスクファイルなどの内容を出力する。
イ　プログラムの実行中にエラーが発生したとき、メモリの内容を出力する。
ウ　プログラムの特定の命令を実行するごとに、指定されたメモリの内容を出力する。
エ　プログラムの命令の実行順序、実行結果などの履歴情報を出力する。

解答解説

トレーサに関する問題である。

トレーサは、プログラムの実行時に制御の流れを追跡するときや実行過程を時系列的に１ステップずつモニタリングするときに使用する。あるデータを処理するときのプログラムの動きを命令単位で調べたり、特定区間の命令を実行する毎に、その所在や命令自身、実行直後のレジスタなどの内容を書き出したりする場合に用いる。誤りの箇所が特定できない時に有効である。区間の指定を的確にしないと、出力量が膨大になり、処理時間がかかる。ワンステップずつ追跡調査できる。

アはダンプツール、イはデバッガ、ウはスナップショットダンプ、エがトレーサである。求める答えはエとなる。

例題演習

プログラムの動作過程を実行順にモニタリングするデバッギングツールはどれか。

ア　インスペクタ　　イ　クロスリファレンス
ウ　トレーサ　　エ　プリティプリンタ

解答解説

デバッグツールに関する問題である。

アのインスベクタは、プログラムを実行し、エラーを検出する動的デバッグツールで、実行を中断しデータ内容の閲覧や更新を対話形式で処理する。

イのクロスリファレンスは、プログラムで用いられている関数、変数や定数に関する名前を相互に参照する静的デバッギングツールである。

ウのトレーサは、プログラムの実行過程を時系列的に１ステップずつモニタリングする動的デバッギングツールで、プログラムをステップごとにエラーを検証する。プログラムの動作過

程を実行順にモニタリングするのはトレーサである。求める答えはウとなる。

エのプリティプリンタは、字下げや予約語のフォント変更など、プログラムの整形機能をもつ静的デバッギングツールである。

例題演習

プログラム実行中の特定の時点で成立する変数間の関係や条件を記述した論理式を埋め込んで，そのプログラムの正当性を検証する手法はどれか。

ア　アサーションチェック　　イ　コード追跡
ウ　スナップショットダンプ　　エ　テストカバレッジ分析

解答解説

プログラムの正当性を検証する手法に関する問題である。

アのアサーションチェックはエラーのないプログラムを作るためのツールで、実行時にそれが満たされていない場合にエラーや例外を発生させたり、メッセージを表示して処理を中断したりする機能である。プログラム中のバグや不具合、論理の矛盾の発見に使われる。プログラムの正当性検証に使用する。求める答えはアとなる。

イのコード追跡は、トレーサを使用して行う。トレーサ上でプログラムを実行すると、その過程を追跡・記録し、実行された命令を順番に、レジスタや変数の値などの状態と共に表示してくれる。これを見ることで、プログラムが意図したとおりに動作しているかを調べたり、どこに誤りがあるかを発見するのが容易になる。

ウのスナップショットダンプは、プログラムにデバッグ命令を組み込んでおき、デバッグ命令を実行する都度、主記憶装置の一部やレジスタの内容を書き出す。指定した条件のときだけ主記憶やレジスタの内容などの追跡データを出力するダンプである。

エのテストカバレージ分析は、テスト対象プログラムのうち、既テスト分と未テスト分を識別するツールである。あるテストデータがプログラムのどの経路を通ったかを調べながら、プログラム全体の経路のうち約何%をカバーしたかを求めることができる。

例題演習

ホワイトボックステストにおいて，コード中のどれだけの割合の部分を実行できたかを評価するのに使うものはどれか。

ア　アサーションチェッカ　　イ　シミュレータ
ウ　静的コード解析　　エ　テストカバレージ分析

解答解説

ホワイトボックステストの支援ツールに関する問題である。

アのアサーションチェッカは、エラーのないプログラムを作るためのツールで、ある条件が成立していなければならない部分にチェック用のコードを入れ，その条件に違反している場合はエラーを出力することで，プログラムをチェックする仕組みである。

イのシミュレータは、ある計算機で実行可能な目的プログラムの命令を１つずつ解釈し、他の計算機の命令に変換するプログラムである。

ウの静的コード解析は、実行ファイルを実行することなく解析を行う。ソースコードに対して行われることが多く、オブジェクトコードに対して行う場合もある。

エのテストカバレージ分析は、ソフトウェア開発において、出来上がったプログラムのテストをする際に、どの程度をテスト対象とするかまたはテストを実行したかの割合を分析することである。求める答えはエとなる。

例題演習

結合テストの目的に関する記述として、適切なものはどれか。

ア　機能が外部設計書に記されているとおりに実現されているかどうかを検証する。
イ　処理時間や応答時間が目標を満たしているかどうかを検証する。
ウ　プログラムの部品であるモジュール間のインタフェースを検証する。
エ　目標どおりにジョブの多重度や端末の同時接続が実現できるかどうかを検証する。

解答解説

結合テストの目的に関する問題である。

結合テストは、モジュール間のインタフェースとプログラムの機能を検査する。

ア、イ、エはシステムテストである。システムテストでは、インタフェーステスト、機能テスト、性能テスト、障害テストや負荷テストが行われる。

ウは結合テストで、求める答えはウとなる。

例題演習

トップダウンテストに関する記述として，適切なものはどれか。

ア　下位のモジュールを代行するドライバの作成が必要になる。
イ　重要度の高い上位のモジュールがテストで繰り返し使用されるので，上位モジュールの信頼性が高くなる。
ウ　テストの最終段階でモジュール間のインタフェース上の問題が生じやすい。
エ　モジュール数の少ない上位部分から開発していくので，開発の初期段階からプログラミングとテストの並行作業が可能である。

解答解説

トップダウンテストに関する問題である。

トップダウンテストの最上位モジュールは最初からテストが繰り返されており、バグの発見も効率よく行え、プログラム全体の制御を司る重要な部分がきちんとテストできる。

アの下位モジュールを代行するのはドライバではなく、スタブである。

イの内容はトップダウンテストの特徴を表している。求める答えはイとなる。

ウの最終段階でモジュール間のインタフェースに問題が生じやすいのはボトムアップテストで

ある。
エの開発の初期から並行作業が可能なのはボトムアップテストである。

例題演習

ボトムアップテストの特徴として，適切なものはどれか。

ア　開発の初期の段階では，並行作業が困難である。
イ　スタブが必要である。
ウ　テスト済みの上位モジュールが必要である。
エ　ドライバが必要である。

解答解説

ボトムアップテストに関する問題である。
アの並行作業は可能である。
イはスタブではなく、ドライバが必要である。
ウはドライバがあれば、テスト済みの上位モジュールは必要ない。
エのドライバが必要は正しい。求める答はエとなる。

例題演習

テスト手法の一つであるボトムアップテストの説明として、適切なものはどれか。

ア　下位のモジュールから上位のモジュールへと順に結合しながらテストする方法であり、未完成の上位モジュールの代わりにドライバが必要である。
イ　個々のモジュールを独立にテストし、各々のテストが終了した時点ですべてを結合してテストする方法である。
ウ　上位のモジュールから下位のモジュールへと順に結合しながらテストする方法であり、未完成の下位モジュールの代わりにスタブが必要である。
エ　単体テスト、結合テスト、システムテスト、運用テストの順にテストする方法である。

解答解説

ボトムアップテストに関する問題である。
アはボトムアップテスト、イはビックバンテスト、ウはトップダウンテスト、エは通常の開発工程で順次行われる一連のテスト手順である。求める答えはアとなる。

例題演習

モジュール間やサブシステム間のインタフェースを検証するために行うテストはどれか。

ア　運用テスト　　イ　結合テスト
ウ　システムテスト　　エ　単体テスト

解答解説

結合テストに関する問題である。

アの運用テストは、ユーザ部門の運用グループが実際の運用と同一の条件を作り出し、機能面および操作面の双方のテストを行うことである。

イの結合テストは、サブシステムやモジュールを結合したテストで、サブシステムやモジュール間のインタフェースや入出力の検証テストである。求める答えはイとなる。

ウのシステムテストは、ソフトウェアやシステムが要求された仕様に合致しているかどうかを検証するために、複数の機能グループを組み合わせて行うテストである。

エの単体テストは、プログラムの最小単位であるモジュールの品質をテストすることで、モジュール内のエラーを発見することである。

例題演習

ボトムアップテストにおいて、被テストモジュールの上位のモジュール機能を代行するのはどれか。

ア　インタプリタ　　イ　コンパイラ　　ウ　スタブ　　エ　ドライバ

解答解説

ボトムアップテストにおけるテスト支援ツールであるドライバに関する問題である。

アのインタプリタは、高水準言語で書かれた実行プログラムを１行ずつ解釈しながら実行していくプログラムである。

イのコンパイラは、プログラムをコンピュータに理解できる機械語に変換するプログラムの一種である。

ウのスタブは、トップダウンテストに用いられるツールである。

エのドライバは、ボトムアップテストに用いられるツールである。求める答えはエである。

例題演習

モジュールテストで使用されるドライバ又はスタブの機能に関する記述のうち、適切なものはどれか。

ア　スタブは、テスト対象モジュールからの戻り値を表示・印刷する。
イ　スタブは、テスト対象モジュールを呼び出すモジュールである。
ウ　ドライバは、テスト対象モジュールから呼び出されるモジュールである。
エ　ドライバは、テスト対象モジュールに引数を渡して呼び出す。

解答解説

テスト支援ツールのスタブ、ドライバに関する問題である。

ドライバはテストモジュールの上位モジュールの機能をシミュレートする。テストモジュールから見て主プログラムの役割を果たす。結合テストのボトムアップテストに利用される。スタブはテストモジュールの下位モジュールの機能をシミュレートするものである。テストモジ

ュールから見て副プログラムの役割を果たす。結合テストのトップダウンテストに利用される。

ア、イはドライバの説明、ウはスタブの説明、エはドライバの説明で適切である。求める答えはエとなる。

例題演習

テスト工程におけるスタブの利用方法に関する記述として、適切なものはどれか。

ア　指定した命令が実行されるたびに、レジスタや主記憶の一部の内容を出力することによって、正しく処理が行われていることを確認する。

イ　トップダウン的にプログラムのテストを行うとき、作成したモジュールをテストするために、仮の下位モジュールを用意して動作を確認する。

ウ　プログラムの実行中、必要に応じて変数やレジスタなどの内容を検査し、必要であればその内容を修正した後、後続の処理のテストを行う。

エ　プログラムを構成するモジュールの単体テストを行うとき、そのモジュールを呼び出す仮の上位モジュールを用意して、動作を確認する。

解答解説

テスト支援ツールのスタブに関する問題である。

アはスナップショット、イはスタブの利用方法、ウはデバッガ、エはドライバである。求める答えはイとなる。

例題演習

システム開発におけるテストでは、小さな単位から大きな単位へ、テストを積み上げて行く方法がとられることが多い。このとき、テストの適切な実施順序はどれか。

ア　システムテスト→結合テスト→単体テスト

イ　システムテスト→単体テスト→結合テスト

ウ　単体テスト→結合テスト→システムテスト

エ　単体テスト→システムテスト→結合テスト

解答解説

システム開発におけるテスト順序に関する問題である。

単体テストは、最小単位であるモジュールの検証および品質をテストする。結合テストは、モジュールを結合して行うテストで、モジュール間のインタフェースの検証、プログラムの入出力を含むテストである。システムテストは、ソフトウェアやシステムが、仕様に合致しているかどうかを検証するテストである。

システム開発におけるテスト順序は、単体テスト→結合テスト→システムテストである。求める答えはウである。

例題演習

システムテスト工程で実施するテストはどれか。

ア　負荷テスト
イ　モジュール間のインタフェーステスト
ウ　モジュール仕様書に基づいた動作確認テスト
エ　レグレッションテスト

解答解説

システムテストに関する問題である。

アはシステムテストで確認する内容、イは結合テストで行う内容、ウはブラックボックステストで確認する内容、エのレグレッションテストはソフトウェアの保守時に行うテストである。求める答えはアとなる。

例題演習

運用テストの実施体制や手順に関する記述として，適切なものはどれか。

ア　運用テストは，システムテストの前工程として実施する。
イ　開発部門がテストケースを設定し，ユーザ部門がこれに従いテストをする。
ウ　開発部門の最後の責任として開発部門主導でテストをする。
エ　ユーザ部門が主体となり，実際に運用するときと同じ条件でテストをする。

解答解説

運用テストの実施体制や手順に関する問題である。

運用テストは、運用と同じ環境で実データを使って行うテストで、利用者が新システムへ業務を移行しても問題がないかどうかをテストして確認する。従って、ユーザ部門が主体となり、実際に運用するときと同じ条件でテストする。

アの運用テストの順序はシステムテストの後工程であって、前工程ではない。

イのテストケースの設定は、ユーザ部門が行うもので、開発部門が設定するものではない。

ウのテストの主導はユーザ部門が行うもので、開発部門主導のテストではない。

エのユーザ部門が主体であり、運用の条件でのテストであるという内容は運用テストの記述である。求める答えはエとなる。

例題演習

ソフトウエアのテスト方法のうち，ソフトウエア保守のために変更した箇所が他の部分に影響しないかどうかを確認する目的で行うものはどれか。

ア　運用テスト　　イ　結合テスト
ウ　システムテスト　　エ　レグレッシヨンテスト

解答解説

レグレッションテスト(退行テスト)に関する問題である。

アの運用テストは、運用と同じ環境で実データを使って行うテストで、利用者が新システムへ業務を移行しても問題がないかどうかをテストして確認する。

イの結合テストは、モジュール間のインタフェースとプログラムの機能を検査する。

ウのシステムテストは、プログラム間の結合とシステムの機能を検査する。

エのレグレッションテストは、既存のソフトウェアを修正する場合、その与える影響についてテストすることで、修正内容と関係ない部分に影響を与え、異常が発生していないかどうかを確認するテストである。求める答えはエとなる。

例題演習

システムの移行テストを実施する主要な目的はどれか。

ア　安全性・効率性の観点で，既存システムから新システムへの切替え手順や切替えに伴う問題点を確認する。

イ　既存システムのデータベースのコピーを利用して，新システムでも十分な性能が発揮できることを確認する。

ウ　既存のプログラムと新たに開発したプログラムとのインタフェースの整合性を確認する。

エ　新システムが要求されたすべての機能を満たしていることを確認する。

解答解説

移行テストに関する問題である。

システムの移行は、新システムを稼働させるために、現行システムから新システムへ、ハードウェアやソフトウェア、各種ファイルを円滑に移し変えることである。そのために、移行方法や移行手順、移行体制、移行日程計画、移行タイムチャートなどを作成する。移行作業の中で大きな比重を占めるものがデータの移行である。利用部門と協同して、現行の業務フロー、データベース仕様を元に、現行システムと新システムのデータ項目を比べ、移行対象データ項目を決める。ハードウェア、ソフトウェア、データの移行方法、移行のタイミングも重要な検討項目である。新たなシステム開発に伴って、ハードウェアの入れ替えや増強、ＯＳやミドルウェアの入れ替えや変更、通信回線の新設や増設が行われる。

アの安全性・効率性の観点で、既存システムから新システムへの切り替え手順、切り替えに伴う問題点を確認するのは移行テストの内容に含まれる。求める答はアとなる。

イのデータベースの確認は運用システムの運用効率・業務効率の測定で行う作業である。

ウのテストは結合テスト、システムテストで実施する。

エの機能確認テストはシステムテストで実施する。