

情報科教育法b

第9回

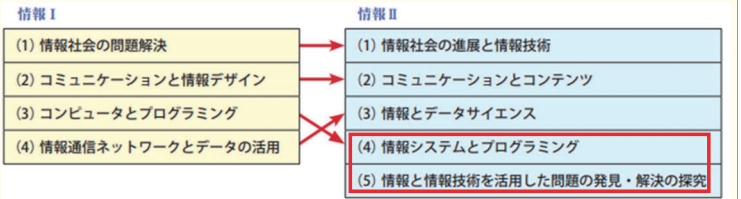
スケジュール 2024

授業	Date	開催	内容
1	8/19	梅田C	ガイダンス、様々な模擬授業を見る
2	8/19	梅田C	情報Ⅰ～情報社会と問題解決
3	8/19	梅田C	情報Ⅰ～コミュニケーションと情報デザイン
4	8/19	梅田C	アクティブラーニングとチーム・ティーチングについて
5	8/20	梅田C	情報Ⅰ～コンピュータとプログラミング
6	8/20	梅田C	情報Ⅰ～情報通信ネットワークの活用
7	8/20	梅田C	チーム・ティーチングの模擬授業 (10分×13人)
8	8/21	梅田C	情報Ⅱについて
9	8/21	梅田C	情報Ⅱについて+指導案
10	8/21	梅田C	模擬授業 (15分×6人)
11	8/21	梅田C	模擬授業 (15分×7人)
12	8/22	梅田C	模擬授業 (20分×5)
13	8/22	梅田C	模擬授業 (20分×5)
14	8/22	梅田C	模擬授業&総括 (20分×3)

本日の内容

- 前回の演習のについて発表
- 情報Ⅱについて (後半)
- 午後は、教採の模擬授業

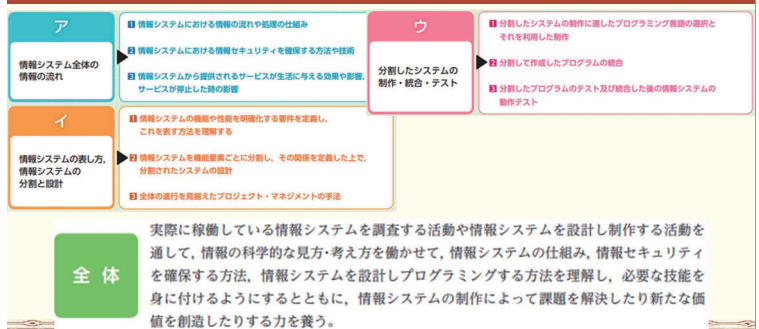
高等学校の情報教育について



4. 情報システムとプログラミング

- 実際に稼働している情報システムを調査する活動や情報システムを設計し制作する活動を通して
 - 情報の科学的な見方・考え方を働かせ
 - 情報システムの仕組み、情報セキュリティを確保する方法、情報システムを設計しプログラミングする方法を理解
 - 情報システムの制作によって課題を解決したり新たな価値を創造したりする力を養う
- こうした活動を通して、以下を養う
 - 情報システムの設計とプログラミングに関わろうとする態度
 - 自分なりの新しい考え方や捉え方によって解決策を構想しようとする態度
 - 自らの問題解決の過程を振り返り、改善・修正しようとする態度
 - 情報セキュリティなどに配慮して安全で適切な情報システムの制作を通して情報社会に主体的に参画しその発展に寄与しようとする態度

4. 情報システムとプログラミング



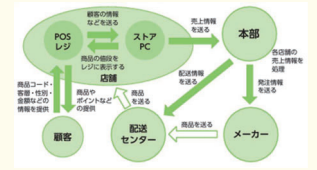
4. 情報システムとプログラミング

1. 情報システム全体の情報の流れ
2. 情報システムの情報セキュリティ
3. 情報システムの表し方
4. 情報システムの分割と設計
5. 分割したシステムの制作とテスト
6. 分割したシステムの結合とテスト
7. 情報システムの評価・改善
8. 情報システムの集中処理と分散処理
9. クライアントサーバシステムとP2Pシステム

4.1.情報システム全体の情報の流れ

1. 情報システムが集めるデータと生み出す利便性

- 情報システムのデータの流れを図解化などで表現
 - POS、処理の流れ
- 情報システム全体の流れを把握
 - CSモデル



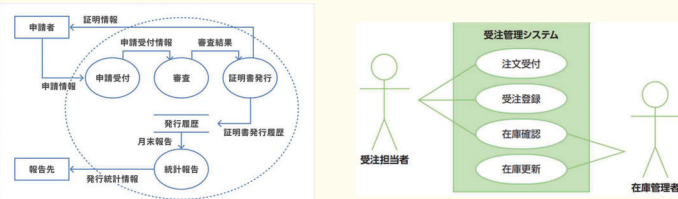
情報システム	集める情報	提供する利便性
電子決済システム	購入した商品 店舗情報 購入日時 支払先の決済システムへの送金情報	支払いの簡便化
SNS	ユーザーから投稿された 文字・写真・動画	ユーザー間の交流の促進
ITS (Intelligent Transport Systems)	位置情報 目的地	最適な交通手段の提供
eラーニング	学習履歴 学習進捗	次に学習する内容の提案

図3 情報システムが集める情報と提供する利便性の例

4.1.情報システム全体の情報の流れ

2. 情報システムの情報の流れや処理の仕組み

- 情報システムの情報の流れや処理の仕組みを表現する方法として、データフロー図、UMLといった表現方法



4.1.情報システム全体の情報の流れ

3. 情報システムの処理形態

- 1人で仕事をする場合と複数人でする場合の違い
- コンピュータの処理の種類と違い

処理名	利用されている情報システム	処理内容
バッチ処理	給与計算システム	情報を一括処理する形態。一定期間あるいは一定量データをためてから、一括して処理をする。データを投入してから処理結果を得られるまでの処理手順が確立しており、その手順に沿って処理を行う。
オンラインリアルタイム処理	産学予約システム 予約システム	データの発生と同時に処理するが、高い信頼性と即時性が要求され、多数の端末からの同時要求に対して、応答時間を短縮できるようにすることが重要である。
リアルタイム制御処理	自動運転システム	極めて厳しい即時性を要求される形態。カメラやセンサーなどを使って常に状態を監視することが要求される。
対話型処理	プログラム開発	人の判断を加えながら情報処理を進める形態。ユーザーがディスプレイ上に表示されたアイコンなどをクリックし、キーボードからコマンドを入力することによって、コンピュータとの情報のやり取りを行う。

図5 情報システムの処理形態

4.1.情報システム全体の情報の流れ

4. 情報システムの集中処理と分散処理

- 集中処理
 - 保守や運用管理、セキュリティの管理が容易ではあるが、ホストコンピュータが故障するとE XERCISEシステム全体が停止
- 分散処理
 - 故障しても処理を継続できるため信頼性が高く、機能の拡張が容易ではあるが、保守やセキュリティ管理、運用管理が複雑

4.1.情報システム全体の情報の流れ

5. クライアントサーバシステムとP2Pシステム

- 分散型システム→CSシステム
- 様々な役割のサーバ
- P2P→暗号資産のブロックチェーンや一部のSNSアプリ

サーバ	役割
ファイルサーバ	ファイルを集約管理し、ユーザーのアクセスに応じてファイルを提供する。
Webサーバ	Webページを管理し、クライアントの要求に応じて情報や機能を提供する。
データベースサーバ	データベースを管理し、アクセス処理、データの取得・挿入などを行う。
コミュニケーションサーバ	通信制御を行う。プロトコルを定めるネットワークを構築する機能を持つ。
プリンターサーバ	プリンタを共有する機能を提供する。
メールサーバ	電子メールの配信を行う。

図7 サーバの種類

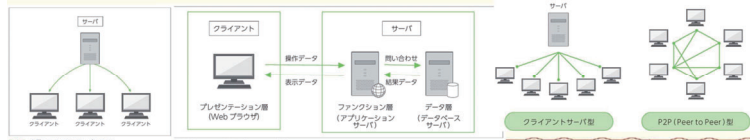


図6 2層クライアントサーバシステム

図8 クライアントサーバシステムの仕組み

4.1.情報システム全体の情報の流れ

6. 組み込みシステム

- 特定の機能を実現するために、専用化されたコンピュータハードウェア
- それを制御するソフトウェアから構成されている組み込みシステム

機器	ハードウェア
通信機器	携帯電話、スマートフォン、FAX
運輸機器	自動車
家電機器	テレビ、洗濯機、冷蔵庫、電子レンジ
医療機器	X線断層撮影装置、磁気共鳴画像診断装置、陽電子放出断層撮影装置
産業機器	エレベーター、信号機、ATM

図表10 組み込みシステムの種類の例

4.2.情報システムの情報セキュリティ

1. 情報システムにおける情報セキュリティ

- 「情報Ⅰ」では個人の情報セキュリティ対策について扱ったが、「情報Ⅱ」では組織や情報システムとしての情報セキュリティ対策

情報システムのトラブル	トラブルの影響
機密情報の漏洩	ウイルスへの感染や不正な情報の持ち出し、記録媒体の紛失は、組織の競争力や信頼を大きく損なう可能性がある。
個人情報の流出	保有する個人情報を流出させてしまった場合は、賠償や訴訟など大きな問題にまで発展する可能性がある。
Web ページ改ざん	組織の顔ともいえる Web ページの改ざんは、イメージ損失につながる。また、Web サーバがウイルスに感染していると、訪問者にウイルスを感染させてしまう。
システム停止	システムが停止してしまうと、最悪の場合は業務そのものが停止してしまう。
ウイルス感染	ウイルスが既に感染したパソコン内で複製され、他のパソコンに感染を広げ、上記で述べた全てのトラブルを引き起こす可能性がある。

図表11 情報システムのトラブルとその影響の例

4.2.情報システムの情報セキュリティ

2. データの暗号化による情報流出の防止

- 悪意のある第三者にデータを盗み見られる「盗聴」
- 送信者になりすまし、受信者にデータを送信する「なりすまし」
- 送信されたデータを不正に書き換えて受信者に送信する「改ざん」

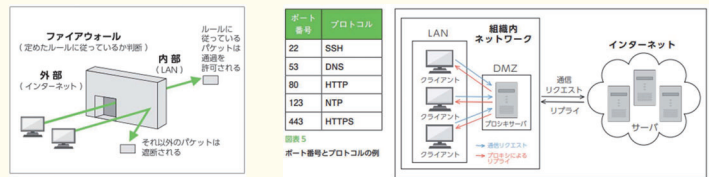
暗号方式	内容
共通鍵暗号方式	暗号化の鍵も復号の鍵も同じ鍵を使用する暗号方式。鍵を盗まれないように大切に管理する必要がある。代表的な方式に AES などがある。盗聴対策に効果的である。
公開鍵暗号方式	暗号化する公開鍵と復号する秘密鍵の2種類の鍵を使用した暗号方式。公開鍵だけを広く公開し、秘密鍵は公開しないので情報を完全にやり取りできる。不特定多数の人と情報をやり取りする場合に用いられ、代表的な方式に RSA や楕円曲線暗号がある。盗聴対策に効果的である。
デジタル署名	送信者の秘密鍵で電子文書の署名データを生成し、送信者の公開鍵で署名データの作成者が電子文書の送信者と同じであることを検証する仕組み。改ざん対策に効果的である。
SSL (Secure Sockets Layer) / TLS (Transport Layer Security)	インターネット上で個人情報やパスワード、クレジットカードなどの情報を暗号化して送受信するプロトコル。SSL は、共通鍵暗号方式の利点と、公開鍵暗号方式の利点の両方を生かすような暗号方式である。SSL の仕組みは、TLS に引き継がれている。

図表13 データの暗号化の仕組みの例

4.2.情報システムの情報セキュリティ

3. 情報システムのファイアウォール

- パケットフィルタリング方式やアプリケーションゲートウェイ方式などを用いて、外部からの不正なアクセスを防止



図表14 パケットフィルタリング方式

図表15 アプリケーションゲートウェイ方式

4.2.情報システムの情報セキュリティ

4. 情報システムの個人認証

ユーザ認証	内容
パスワード認証	ユーザーだけが知っている ID とパスワードにより認証する方式。
ワンタイムパスワード	一度しか使用することができない使い捨てのパスワードをその都度発行する方式。
コールバック	アクセス権を持つ端末であることを確認するため、回線をいったん切り、システム側から再発信して通信を開始する方式。
バイオメトリクス認証	身体的特徴や行動的特徴を抽出して認証する方式。身体的特徴としては、指紋や静脈認証がある。行動的特徴としては、署名するときの速度や筆圧から特徴を抽出する方式などがある。
多要素認証	ユーザーだけが知っている知識情報、ユーザーだけが持っている所持情報、ユーザーだけの生体情報などのうち、2つ以上を組み合わせて認証する方式。

図表17 代表的なユーザ認証の例

4.2.情報システムの情報セキュリティ

5. アクセス制御とアクセス権

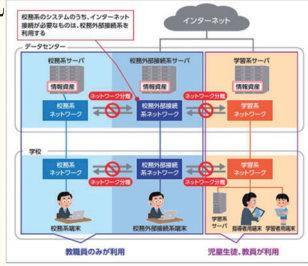
- アクセス制御の方法は、「認証・認可・監査」という3つのモデルから成り立つ

- 認証：対象の正当性や真正性を確かめるもの、コンピュータにログインするユーザーが本人かどうかを識別するユーザ認証などがある
- 認可：管理者があらかじめ定めた条件を満たしたユーザーのみアクセスを許可する制御方法
- 監査：認証や認可でとった処理の過程で発生したログを記録し、検証した結果、条件を見直すことによってより高い精度のアクセス制御を行う制御方法

4.2.情報システムの情報セキュリティ

6. ネットワークのセグメント化

- 「教育情報セキュリティポリシーに関するガイドライン」（令和元年12月版）が公開されており、セキュリティ対策
- セグメントを分割することで、学習者用端末や指導者用端末から、校務系ネットワークや校務外部接続系ネットワークへのアクセスを防ぐことができる



図表10 教育現場におけるセグメント化
出典：「教育情報セキュリティポリシーに関するガイドライン」(令和元年12月版) 文部科学省
http://www.mext.go.jp/content/20200225main_image02-100003137_003.pdf

4.3.情報システムの表し方

1. 要件定義

- 業務上実現すべき要件である業務要件定義
- 業務要件を実現するために必要な情報システムの機能を明らかにする機能要件定義

業務要件定義	機能要件定義
<ul style="list-style-type: none"> 利用者がシステムを使う目的を明確にする。 利用者がシステムを使うことで得られる利益を明確にする。 システムを使うことで利益を得るためのプロセスを定義する。 	<ul style="list-style-type: none"> 目的を達成するために、利用者がシステムを使って行う仕事を明確にする。 ユーザーの視点で、システムは何を行う必要があるかを定義する。
<p>例)</p> <ul style="list-style-type: none"> 煩雑な在庫管理を情報システムに置き換えることで管理コストをカットする。 利用者はシステムを通して、常に正しい在庫数を確認することができる。 	<p>例)</p> <ul style="list-style-type: none"> 利用者はシステムを使って、「在庫確認」「追加発注」「在庫履歴印刷」ができる。 システムは在庫のリストをデータとして保持し、納品後はデータを更新し、在庫確認時は在庫の数を正しく表示する。

図表1 業務要件定義と機能要件定義の違い

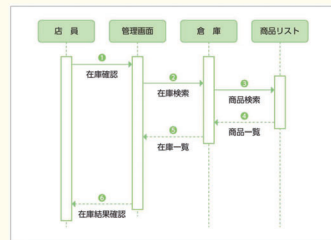
4.3.情報システムの表し方

2. 利用者とシステムのやり取りの図表化

- ユースケース図
- シーケンス図



図表1 ユースケース図のイメージ

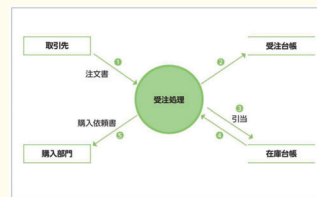


図表2 シーケンス図

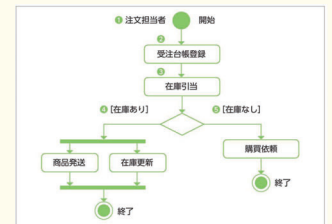
4.3.情報システムの表し方

3. データの流れに着目した図表化

- DFD (Data Flow Diagram)
- アクティビティ図



図表3 DFDの例

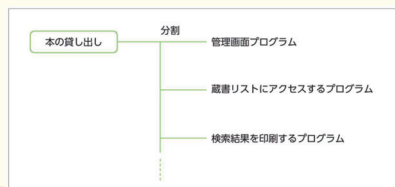


図表4 アクティビティ図

4.4.情報システムの分割と設計

1. ソフトウェア方式設計

- 外部設計：ユースケース図やシーケンス図などでは利用者の視点からシステムに何が求められるかを整理
- 内部設計：整理した内容をどのようにプログラムとして記述するかを考えるもの

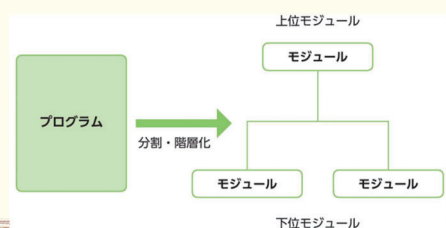


図表1 ソフトウェア方式設計のイメージ

4.4.情報システムの分割と設計

2. ソフトウェア詳細設計

- ソフトウェア詳細設計ではプログラムを更に細かく、モジュールと呼ばれる単位に分割・階層化し、モジュール間のインタフェースを明確にする

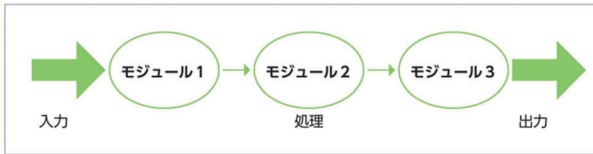


4.4.情報システムの分割と設計

3. モジュール分割技法

• モジュール分割

- ①最上位のモジュールの定義
- ②モジュールの機能分析
- ③モジュールの分割技法の選択
- ④モジュールの分割
- ⑤インタフェースの定義
- ⑥分割すべきほかのモジュールの検討



図表3 データの流れに着目したモジュール分割のイメージ

4.4.情報システムの分割と設計

4. モジュール結合とカプセル化

• モジュール分割の留意点

- 一つのモジュールの大きさが適切になるようにする
- 一つのモジュールから呼び出す他のモジュールの数に制限を付ける
- モジュールからモジュールを呼び出す階層構造があまり深くないようにする
- モジュール間のインタフェースが単純になるようにする



図表4 独立性の高いモジュール

図表5 カプセル化のイメージ

4.5.分割したシステムの制作とテスト

1. プログラミング言語の選択

- プログラミング言語には、データの扱いに適したもの、機械の制御に適したもの、グラフィックスに強いものなどの特性

• コンパイル方式

- コンピュータが直接実行可能な形式を生成する：C言語など

• インタプリタ方式

- 記述したプログラムを実行時に逐次コンピュータが直接実行可能な形式に翻訳する：Pythonなど

言語	用途の例
C	OS や組み込みプログラムの開発など
C#	Windows のアプリケーション開発など
Java	情報システム開発、Android のアプリケーション開発など
JavaScript	Web 画面で動作するプログラムなど
PHP	Web サーバで動作するプログラムなど
Python	機械学習のプログラムなど
R	統計処理のプログラムなど
Ruby	Web サーバで動作するプログラムなど
Swift	Mac や iOS のアプリケーション開発など

4.5.分割したシステムの制作とテスト

2. 単体プログラムの作成

- 単体プログラムのデータの入力及び出力としては次のものが一般的である

- 関数への引数や戻り値として渡す
- ネットワーク通信のメッセージとして渡す (WebAPI)
- Pythonを使って実際にプログラムをする

変数	説明
url	図書館検索を行うサーバの URL
以下は、図表3 の params で指定するパラメータ	
token	Web API を利用するためのアクセスキー
table	使用するテーブルの名前
column	検索する列の名前
value	検索する文字列
format	出力形式に json を指定する

図表2 使用する図書館検索 Web API の仕様

4.5.分割したシステムの制作とテスト

3. 単体テスト

- 内部設計の仕様を満たしているかどうか、また正常に機能するかどうかを確認するテスト

• ホワイトボックステスト

- 「ソースコードのこの条件はテストしているか」「ソースコードのEXERCISEこの条件分岐の実行は両方ともテストされているか」のように、プログラムのソースコードを見ながらテストを行う

• ブラックボックステスト

- ソースコードを見ずに仕様書の機能を満たしているかを考えながらテストを行う

4.5.分割したシステムの制作とテスト

4. デバッグ

- デバッグの工程は大きく次の2つに分けられる

- エラーの性質と場所を特定すること
- エラーを修正すること
- エラーの場所を特定するためには、原始的な手法として、次のようなものがある
 - プログラム中に表示命令を散りばめて、デバッグ用メッセージを出力する
 - デバッガでの情報表示

4.6. 分割したシステムの結合とテスト

1. 結合テスト

- はモジュールを結合して動かし、全体として正しい機能を備えていることを検証するテスト
- 単体テストが終わっていることが前提

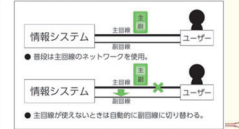
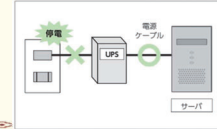
結合テストの方式	特徴	欠点
トップダウンテスト	<ul style="list-style-type: none"> • インタフェースエラーなど重大な欠陥を早期に検出できる。 • 重要度の高い上位モジュールを繰り返しテストすることになり、全体の信頼性が高まる。 • テストドライバを作成する必要がない。 	<ul style="list-style-type: none"> • 作業の分散が難しい。 • スタブを作成する必要がある。
ボトムアップテスト	<ul style="list-style-type: none"> • 分担しての並行作業が容易であり、早期に多くのモジュールをテストできる。 • 欠陥の影響が大きい共通モジュールの品質を先行して高めることができる。 • スタブを作成する必要がない。 	<ul style="list-style-type: none"> • 重要な欠陥が後になって検出される。 • テストドライバを作成する必要がある。

31

4.6. 分割したシステムの結合とテスト

2. 総合テスト

- 総合テストは、機能テストと非機能テストに大きく分けられる
- 機能テスト：要求仕様に規定されている機能がシステムに盛り込まれているかどうかを確認するテスト
- 総合テスト：実際にユーザーが利用するのと同じように、手作業で入力したり操作したりして試験
全ての機能の動作を網羅するには作業工数が多い



32

4.6. 分割したシステムの結合とテスト

3. セキュリティテスト

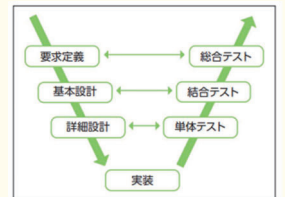
- (1) ユーザ認証に対する攻撃
 - ブルートフォース攻撃、リバースブルートフォース攻撃、リスト型攻撃
- (2) XSS (クロスサイトスクリプティング)
 - HTMLページに、<Script> (スクリプトタグ) を追加して、そのサイトの範囲内で動的なページであるかのように振る舞う攻撃
- (3) CSRF (クロスサイトリクエストフォージェイリ)
 - Webサイトのプログラムに意図しない不正なリクエストを実行させ、サーバの情報を取得したり、コードを実行させたり、などの動作

33

4.7. 情報システムの評価・改善

1. 情報システムの評価・改善

- ウォーターフォール型のシステム開発では、開発工程とテスト工程を対応させたV字モデルという考え方
 - プロトタイプ型の開発：「大まかに動く試作品を短期間で作成し、その動作を確認しながら仕様を修正する」
 - スパイラル型の開発：機能を限定したシステムの作成と評価を行い、徐々に機能を追加しながら繰り返してシステムを作り上げる



34

4.7. 情報システムの評価・改善

2. プログラムの改善・改良

- プログラムの構造化
- テスト駆動開発：テスト駆動開発 (TDD: Test Driven Development) と呼ばれる手法
 - Pythonにおいては、標準ライブラリに unittest モジュールというテストユニットが用意されている。
- リファクタリング
 - 「外部から見たときの振る舞いを保ちつつ、理解や修正が簡単になるように、ソフトウェアの内部構造を変化させること」と定義

従来の開発順序	テスト駆動開発での開発順序
<ul style="list-style-type: none"> • 内部設計書を作成 • 単体プログラムを作成 • テストを設計 • テストの実施 (成功) 	<ul style="list-style-type: none"> • 仕様の確定 • テストを設計 • テストの実施 (失敗) • 単体プログラムを作成 • テストの実施 (成功)

図表2 開発順序の比較

メリット	デメリット
<ul style="list-style-type: none"> • 開発の遅延を減らす • テストの自動化による効率向上 • コードの品質向上 • 開発者の負担軽減 • 開発プロセスの透明化 • 開発者のモチベーション向上 	<ul style="list-style-type: none"> • 開発初期の学習コスト • テストの実行時間が増える • テストの実行環境の構築 • テストの実行結果の解析 • テストの実行結果の報告 • テストの実行結果の活用 • テストの実行結果の活用 • テストの実行結果の活用

図表3 リファクタリングのメリット

35

4.7. 情報システムの評価・改善

3. チームでの開発の進め方

- コーディングスタイル
- プロジェクト・マネジメント
- ①WBS：作業を分解して構造化する手法
- ②ガントチャート：プロジェクトの進捗を管理するためのツールの一つにガントチャート (Gantt chart)
- ③コミュニケーションツールの利用



作業名	作業コード	担当者	開始日	終了日	進捗
システム分析	A-1	田中	2020-01-10	2020-01-14	100%
設計	A-2	田中	2020-01-10	2020-01-12	100%
開発	A-3	田中	2020-01-10	2020-01-12	100%
テスト	A-4	田中	2020-01-10	2020-01-11	100%
運用	A-5	田中	2020-01-10	2020-01-11	100%



作業名	作業コード	担当者	開始日	終了日	進捗
システム分析	A-1	田中	2020-01-10	2020-01-14	100%
設計	A-2	田中	2020-01-10	2020-01-12	100%
開発	A-3	田中	2020-01-10	2020-01-12	100%
テスト	A-4	田中	2020-01-10	2020-01-11	100%
運用	A-5	田中	2020-01-10	2020-01-11	100%

図表6 ガントチャートの例

36

コンピュータとプログラミングと情報システムとプログラミング



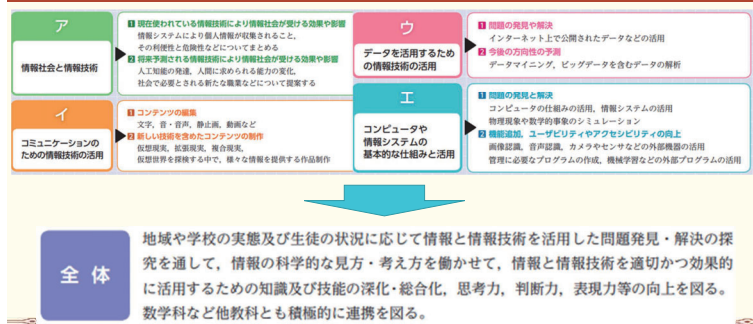
コンピュータとプログラミングと情報システムとプログラミング

コンピュータとプロ...にだけ出現	コンピュータとプロ...によく出る	両方によく出る	情報システムとプロ...によく出る	情報システムとプロ...にだけ出現
シミュレーション アルゴリズム 使いやすい 優しい 少ない 深い 読みやすい 事象 結果 装飾 計算 違い 能力 関係 特徴 オペレーティングシステム メモリ 精度 出る 変える 学ぶ	コンピュータ モデル 表現 プログラミング 活用 適切 学習 行つ データ 解決 問題	よい 分かりやすい 考える 情報 方法 プログラム 応じる 理解 活動 利用 仕組み 使う 踏まえる 改善 機能 身 用いる 技能 処理 評価 触れる 調べる	正しい 扱う 齎す 付ける 取り上げる できる 含む 開発 技術 関数 構成 含める 振り返る	しやすい 新しい 遠い 高い 情報システム 制作 設計 システム 及ぼす 基づく 果たす 示す 届ける 見守る 過す モジュール 流れ プロジェクト 影響 運用 いく きる ける ねらう まとまる ける 与える 供する 取り扱う 取る

5. 情報と情報技術を活用した問題発見・解決の探求

- 教科の目標に沿って、地域や学校の実態及び生徒の状況に応じて情報と情報技術を活用して問題発見・解決の探求を通して
 - 情報の科学的な見方・考え方を働かせ
 - 情報と情報技術を適切かつ効果的に活用するための知識及び技能の**深化・総合化、思考力、判断力、表現力等**の向上を図る
- このような活動を通して
 - 情報社会における問題の発見・解決に**情報と情報技術を適切かつ効果的に活用**しようとする態度
 - 新たな価値を創造**しようとする態度
 - 情報社会に参画**しその発展に寄与しようとする態度

5. 情報と情報技術を活用した問題発見・解決の探求



5. 情報と情報技術を活用した問題発見・解決の探求

- 情報社会と情報技術
- コミュニケーションのための情報技術の活用
- データを活用するための情報技術の活用
- コンピュータや情報システムの基本的な仕組みと活用

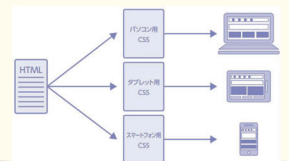
5. 情報と情報技術を活用した問題発見・解決の探求

1. 情報社会と情報技術

- 生徒が主体的に興味のある内容を取り上げる
- テーマ設定→調査→仮説→提案手法→実行→評価、改善
例) AIについて

2. コミュニケーションのための情報技術の活用

- 多様なメディアを組み合わせることでコンテンツを制作
例) 学校の紹介CM作成



5. 情報と情報技術を活用した問題発見・解決の探求

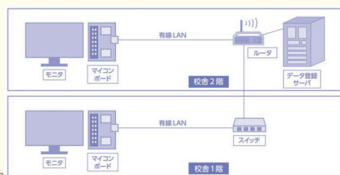
3. データを活用するための情報技術の活用

- (3) だけでなく、他の単元で学んだ内容も総合的に活用でき、かつ探究活動として実施



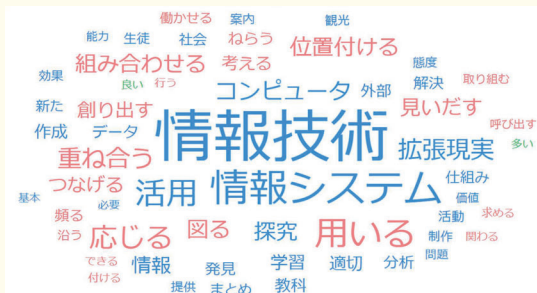
4. コンピュータや情報システムの基本的な仕組みと活用

- 生徒自身が思い描いた情報システムを作成
例) 学校内のIoT化など



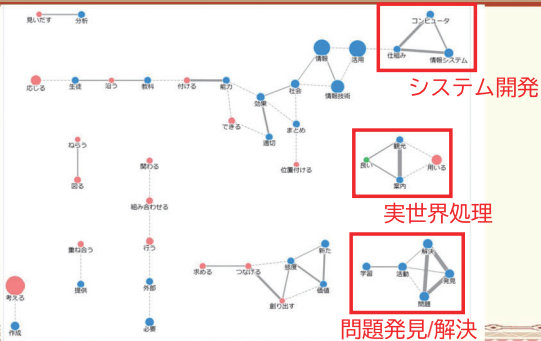
43

どんなことを学ぶのか



44

どんなことを学ぶのか



45

演習

「5. 情報と情報技術を活用した問題発見・解決の探求」において、これまで学んできた内容と生徒の実態に応じて様々なテーマ設定をする必要がある。

教員になったとして、どのようなテーマを設定し、進めていくのかをまとめてみて発表せよ

パワポ1枚にまとめて3分で発表する
名前は記入する

46

課題

1. 情報IIについて、普通科で実施する場合にどのような要件が必要になるか考えよ
また、現状の情報の教員の観点や今後の情報教員に求められる力なども踏まえて書かきましょう

- 提出：Googleフォーム
• 締め切り：7/31 17時まで

47