第 1 章

OS としての Linux

オペレーティングシステム (Operationg System)，略して OS は，日本語では「基本ソフトウェア（基本ソフト)」と呼ばれるように，コンピューターを使用する際には不可欠のソフトウェアである。現在，我々の身の回りにあるコンピューターといえばパソコン (PC)，スマートフォンやケータイ，ゲームマシン・・・等が頭に浮かぶが，電化製品の中にコンピュータが入っていないものはないと言ってよい。天気予報はスーパーコンピュータを使ったシミュレーションに基づいて行われるし，Google や Yahoo といったサーチエンジンは日々の情報検索には欠かせないツールとなっている。OS はこれらのコンピュータ全てに搭載されているものである。現在の OS は Windows^{TM} 系統のものが特に PC では殆どのシェアを握っているが，Linux(リナックス) と呼ばれる UNIX 互換 OS は Windows より種類が豊富で，特に Internet の基幹サーバ，スーパーコンピュータでは圧倒的なシェアを誇っていることは余り知られていない。本章では，コンピュータの仕組みと OS の役割，その上で動作するネイティブアプリケーションと Web アプリケーションの違い，そして Linux について，大ざっぱに必要なところだけ解説する。

1.1　PC(ハードウェア) と OS(基本ソフトウェア)

コンピューターは，電子計算機とも呼ばれる通り，全てのデータを電子的に保持して 2 進数 (ビット列) として表現し，それを 2 進データのまま処理する（計算する）機械である。処理すべきデータは半導体で作られたデジタル回路の中を電気信号の高低で表現されて通過する。単純に，電圧の高い所を 1，低いところを 0 と表現することにすると，例えば 00101010 というビット列は図 1.1 左のようになる。

この信号は水晶発振器によって規則的に発せられ，一秒間に膨大な周期でビット列が大量に送られる。1 秒間に送信される電気信号の周期 (波形) の数を動作周波数と呼び，Hz (ヘルツ) を単位として表現する。これがコンピュータの処理速度を決める一つの要素となる。例えば 3.3GHz のマシンは単純計算で一秒間に 33 億ビットものデータを運ぶことが出来ることになる。

デジタル回路では，一本の回線に 1 つのビット列が割り当てられて送られる。この回線をデータバス (bus) と呼ぶ。図 1.1 右図に示すように，データバスは 1 バイト (8bit) を最小単位として束ねられ，平行して複数の bit 列を送れるようになっている。この束ねられた回線数が，コンピュータが一括処理できる bit 数ということになるので，このデータバスの回線数を称してコンピュータのビット数と呼ぶ。現在の PC は 32 ビットマシン，64

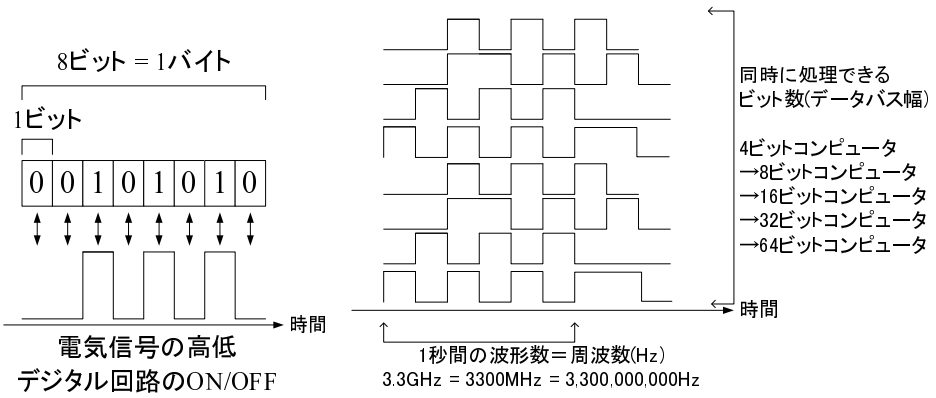

図 1.1 ビット (bit) 列 (左図) とデータバスを流れるデータ・周波数 (右図)

ビットマシンが主流であるが，これはデータバスの回線数だと思えばよい[*1]。データバスの本数（太さ）が多ければ多いほどコンピュータは高速になる。

従って，コンピュータは動作周波数×ビット数（データバスの太さ）に比例して高速になることが分かる。同じ動作周波数でも 32 ビットマシンより，64 ビットマシンが高速であるのはこの理由による。

1.2 OS の役割=コンピューター資源の管理

コンピュータの例として，PC を構成する主要パーツを模式図にしたものと，その上で動作するアプリケーションソフトウェアの例を図 1.2 に示す。全てのパーツはデータバスで結ばれ，データはビット列 (2 進表現) として流れている。

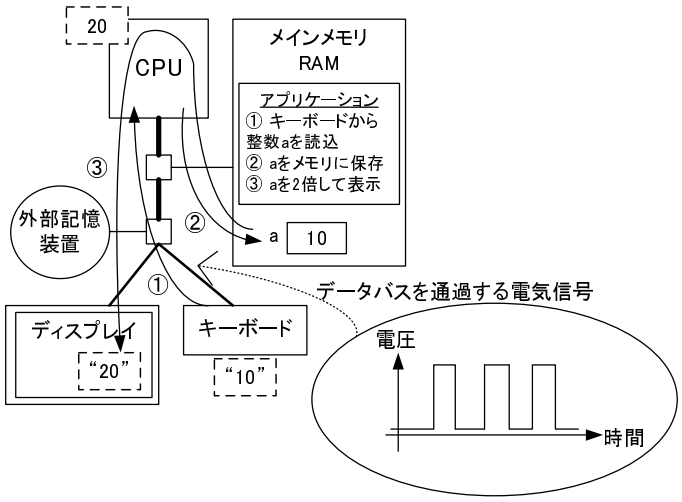

図 1.2 PC の構成とデータの流れ

コンピューターを使用する人間（ユーザ）が直接使う装置がディスプレイ（出力装置,

[*1] 複雑化した現在の PC 内部では必ずしも全てのデータバス幅は一致していない。

Output）とキーボード・マウス（入力装置，Input) である。これらは標準的な入出力装置であるため，標準出力 (standard output, stdout)，標準入力 (standard input, stdin) とも呼ばれる。図 1.2 では「10」という文字列を標準入力から入力し，アプリケーションソフトウェアがそれを 2 倍にし，標準出力に「20」を表示している。

このような一連の処理内容を記述し，コンピューターの頭脳に当たる CPU(中央演算処理装置) が順次その内容を解釈して処理を行うための指令書をプログラム (program) あるいはアプリケーションソフトウェア (アプリ，アプリケーション，application software, application) と呼ぶ。基本的にはアプリケーションソフトウェアは半導体で作られている高速な記憶装置，メインメモリに読み込まれ，CPU はメインメモリを読み出してそこで指示された処理を順次行う。読み込まれた $a = 10$ というデータに 2 を乗じて標準出力に表示するまで，すべてデータバスを通じてデータがやりとりされる。

起動されていないアプリケーションソフトや処理結果のデータを長期保存するために，ハードディスクや光ディスク (CD, DVD, Blu-ray 等) 等の外部記憶装置 (storage) がある。ここにファイル (file) という形で，ビット列の塊を磁気データ等に変換して保存する。

これら一連の操作を全てアプリケーションから直接各パーツを制御することは現実的ではない。ことに近年の複雑なハードウェア機構，即ち，コンピュータ資源 (computer resource) を一つのソフトウェアだけで把握することは不可能に近い。

そこで，どんなアプリケーションソフトウェアでも不可欠な，以下のような基本機能を提供してくれるソフトウェアを用意し，それを介してアプリケーションソフトが動作するようにしておくのが，現在のコンピュータの基本的な設計思想である。

ファイル管理　外部記憶装置への適切なファイル配置
プロセス管理　複数のアプリケーションソフトウェアの起動・終了の管理
ユーザ管理　コンピュータ資源を使用する複数ユーザの管理・セキュリティ確保
ユーザインターフェースの提供　コンピュータを動かすための CUI と GUI
ネットワーク基盤の提供　インターネット上でデータをやりとりするための TCP/IP ネットワークプロトコルの基本機能を提供

これらの基本機能を提供するソフトウェアがオペレーティングシステム (Operating System)，OS(オーエス) である。OS の機能を階層 (レイヤー, layer) に分けて示したのが図 1.3 である。

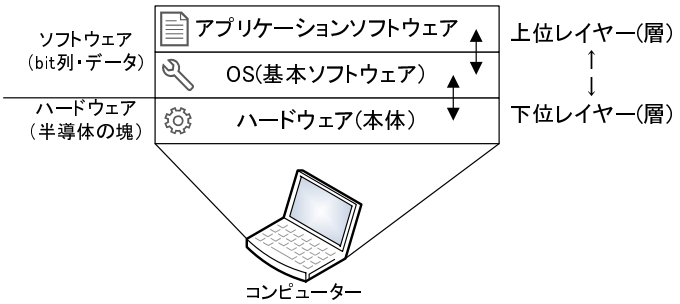

図 1.3　OS とソフトウェア階層

アプリケーションソフトウェアは，データをファイルに保存したり読み込んだり，イ

ンターネットに接続したりという処理を下の階層にある OS に要求する。OS は，ドライバー (driver) というハードウェアを直接制御するソフトウェアを介してコンピュータを操作する。家電やスマートフォンから，ゲームマシン，PC に至るまで，現在のコンピュータはこの OS なしでユーザが満足に動作させることは事実上無理である。

1.3 Linux とは？

現在，多種多様な OS が提供されているが，PC 用としては次の 2 系統のもので占められている。

Windows(TM) 系統 マイクロソフト社が独占的に開発・販売している OS。CUI のみの簡素な MS-DOS から出発し, その後，Windows95/98/ME/NT/2000/XP/Vista/7 が一般コンシューマ向けに提供されてきた。現在では 3 年おきに新しい Windows OS がリリースされている。サーバ用として Windows Server 2003/2008 もある。

UNIX 互換 OS 系統 Net/OpenBSD, Linux, Solaris, MacOS X などの系統が今に残る UNIX 互換の OS と言える。UNIX「互換 (Compatible)」と呼ぶのは，UNIX™ が商標登録されて使えない名前であること，UNIX と同じ機能を持っているということを明示するためである。

Linux は UNIX 互換 OS の中でも最も有力な OS の一派である。Linus Trovalds が 1991 年に OS としての基本機能 (カーネル, Kernel) を公開して以来，カーネルは共通ながら，様々な付加アプリケーションソフトウェアをまとった，Linux ディストリビューション (Linux distribution) が開発されている。本書ではそのうち，商用に販売されている RedHat Enterprise Linux(RHEL) から無料で配布可能なソフトウェアだけを抜粋して作られた CentOS 5 を用いる。

1.3.1 GUI と CUI

現在の OS は，基本的な操作を簡単にできるよう，視覚的に分かりやすい画面表示をユーザに対して提供している。例えば Windows 7 の画面は，使用したいアプリケーションソフトウェアが一つ，あるいは複数の窓 (ウィンドウ, Window) を開き，ユーザから・ユーザへの入出力を受け付ける。このようなユーザと直接接する画面・入出力装置をユーザーインターフェース (user interface)，あるいは単にインターフェースとも呼ぶ。特に図 1.4 左のようにグラフィカルなユーザーインターフェースを GUI(Graphical User Interface の略) と呼ぶ。今や GUI を持っていない OS は多数のユーザを獲得することは不可能であろう。

しかし，グラフィカルな処理には多大な CPU パワーを有する。画面の 1 点ごとに 1672 万色 (24bit フルカラー) の配色が可能だとすると，ハイビジョンの画素数 (1920 × 1080) の画像を表示するためには 49766400bit = 6220800 Byte ≈ 5.9MB ものデータが必要となる。

これに対して，文字データのやりとりだけで OS を操作する方式もある。現在でも Windows ではコマンドプロンプトという名前で，OS に対して命令を文字列（コマンド）で与える機能が残っている (図 1.4 右)。このようにコマンドだけで OS を操作するユー

ザーインターフェースを CUI(Character User Interface) と呼ぶ。システムトラブルなどで GUI が使用できない場合，OS 起動時に一時的にファイルを退避したり，システム情報を書き換えたりする必要がある場合，今でも CUI が使用されるのは，GUI よりコンピューターへの負担が少ないことによる。

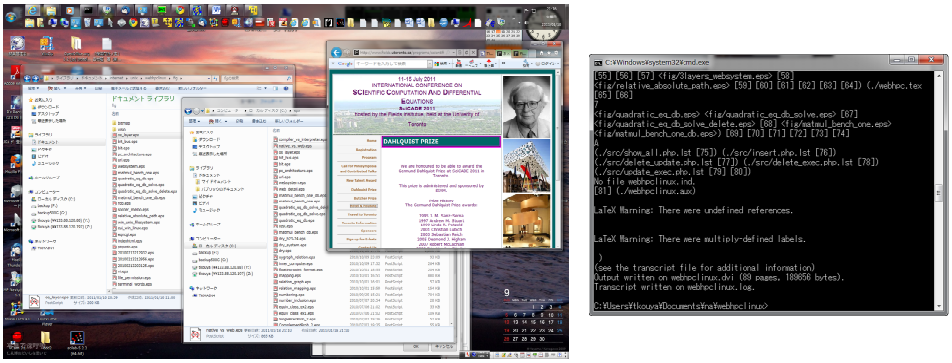

図 1.4 Windows7 の GUI 画面 (左) と CUI 画面 (右)

UNIX 互換系統の OS も，ベース部分は未だに CUI だけで完結しているが，X Window という GUI を提供するソフトウェアを搭載してからは，Windows OS と共通の操作方法を多く持つ GUI も使うのが普通になっている。現在の Linux では KDE か GNOME という GUI のどちらかを使うのが一般的である。

1.3.2 CentOS とは？

Linux の初期の頃は様々なディストリビューションが存在していたが，今では RedHat 社が開発している RHEL から派生した RPM 系統のものと，Debian GNU/Linux 系統のものが主流になっている。カーネルはもちろん，アプリケーションソフトウェアも共通のものが多いが，アップデートやインストール・削除といった管理機構に最大の違いがある。Linux のカーネル自体は配布自由，ソースコード（後述）は公開されているオープンソフトウェア (Open Software) で，使用の際にライセンス料は一切課されないものであるが，RHEL はカーネル以外のアプリケーションソフトウェアや管理機構に有償のものを含み，サポートも有償で提供している商用 OS である。Debian GNU/Linux は完全フリーで，ディストリビューションごと自由に配布できる無償 OS である。

CentOS は前述したように，RHEL から有償のソフトウェアを外して Debian GNU/Linux と同様に自由に配布，無償で使用できるようにしたディストリビューションである。そのため，RHEL のアップデートと同じタイミングでバージョンアップがなされる。配布元の Web サイトの運営は寄付で賄っているとのことである。

CentOS も GUI を標準搭載しており，KDE か GNOME もどちらかを選択できる。図 1.5 左は GNOME の画面であるが，マウス操作，ツールバー，ウィンドウ操作は Windows OS の GUI によく似ており，Windows の使用経験があればまごつくことはないであろう。

しかし，本書ではこの華やかな GUI は一切使用しない。その代り，無味乾燥な CUI(図 1.5 右) のみを使用する。コマンドの使用方法を記憶しないとスムーズに使えない上に，すべてのコマンドが英語由来の文字列になっているため，最初は困難が伴うであろう。しか

図 1.5　CentOS5 の GUI(GNOME) 画面 (左) と CUI 画面 (右)

し，一度慣れてしまえば，ネットワーク越しに CUI を使ってもストレスが殆どないはずである。文字データのやり取りしかしないため送受信データ量が小さくなるからである。もちろん Windows におけるリモートデスクトップのような機能もあるが，本書では扱わない。

後述するが，CentOS を始めとする PC 用 Linux ディストリビューションは，もっぱらインターネット上で 24 時間サービスを提供しているサーバ（Server）として使用されている。現在でも Web サーバに占める Linux のシェアは Windows Server よりも高いと言われている。本書の後半では Web アプリケーション（後述）を作成していくことになるが，CentOS 上で Web アプリケーションを動作させることができるようになれば，他の Linux ディストリビューションを使っている Web サーバでも同じように動かすことが可能になる。ネイティブアプリ（後述）より，Web アプリケーションを開発するスキルを磨くためにも，CentOS に親しんでおくことは無駄ではない。

1.3.3　Android は Linux の一種

余談だが，2010 年代に入って急激にスマートフォン市場でシェアを伸ばしつつある Android(TM) についても触れておく。図 1.6 は，シャープ/au から 2010 年 11 月下旬に発売された IS03 の GUI と CUI である。

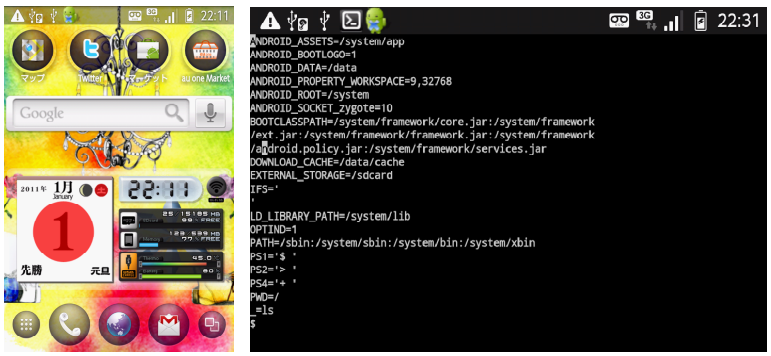

図 1.6　Android(au/Sharp IS03) の GUI 画面 (左) と CUI 画面 (右)

本書の第 2 章以降で CentOS の CUI に親しんでみれば分かるが，この Android はスマートフォン・パッドタイプデバイス向けに Google 社が開発した，Linux ディストリビューションの一種であり，CUI はまるっきり CentOS のそれと同じである。ディレクトリ（フォルダ）構成に相違点があるものの，代表的な UNIX コマンドも共通で使用できる。

Windows の陰に隠れていた Linux であるが，PC より圧倒的な台数が全世界的に普及すると予想されるスマートフォン・パッドタイプデバイスでは Android のシェアが急激に伸びると予想されている。Linux に親しんでおくことは，今後の ICT 開発には欠かせない素養となるであろう。

1.4　アプリケーションソフトウェアの種類と開発方法

本書は CentOS 上で動作するアプリケーションソフトウェアを作成する技法とその動作原理を理解してもらうことを最終目標としている。インターネットが完全に我々の世界のインフラとして存在する昨今では，特定のハードウェア・OS 上で動作するアプリケーションソフトウェアだけではなく，インターネットを介して不特定多数に対してサービスを提供する Web(World Wide Web) をユーザーインターフェースとするアプリケーションソフトウェアも欠かせない。前者をネイティブアプリケーション (Native Application)，後者を Web アプリケーション (Web Application) と呼ぶ。ここではこの両者の仕組み・長所短所を概説し，アプリケーションソフトウェアを開発するためのプログラミング言語 (Programming Langurage) について簡単に触れる。

1.4.1　ネイティブアプリケーション vs. Web アプリケーション

ネイティブアプリケーションと Web アプリケーションの仕組みの違いを示したのが図 1.7 である。

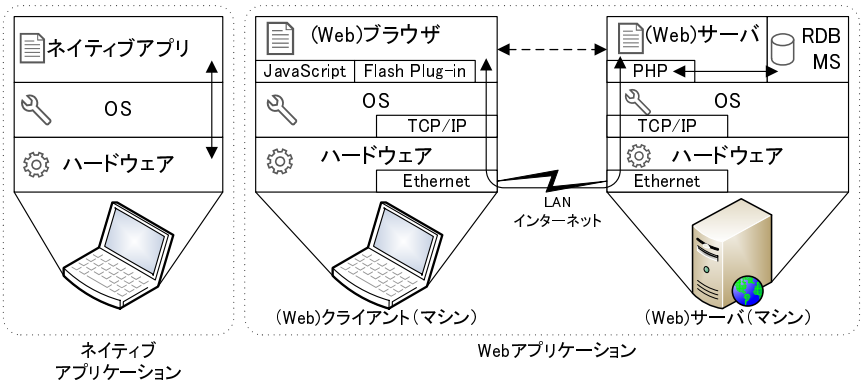

図 1.7　ネイティブアプリケーション (左) と Web アプリケーション (右)

一見して分かる相違点を挙げておこう。

1. ネイティブアプリケーションは，ハードウェアと OS に依存したソフトウェアである。
2. Web アプリケーションは (Web) サーバに設置され，クライアント (ブラウザ）から

のアクセスを受けてサービスを提供する。

3. Web アプリケーションを支える Web サーバソフト，ブラウザは共にネイティブアプリケーションであり，これらと，両者を結ぶインターネット・LAN(Local Area Network) の三者がサービスを支えている。

本書の後半ではネイティブアプリケーションと Web アプリケーションの簡単なサンプルを示し，それを構築していく。ネイティプアプリケーションの例としては，高性能な科学技術計算を目的とする HPC プログラミングの例として，2 次方程式と行列・ベクトル計算を取り上げる。HPC はネイティブアプリケーションを主体とし，ギリギリまでハードウェアの性能を引き出すことを目的とする。

Web アプリケーションは，ブラウザやサーバソフト，ネットワークといった積み重ねの上に成立しているもので，性能は当然ネイティブアプリに劣る。しかしアクセスのしやすさ，ブラウザとサーバでコンピューター資源を分担し合えるというという特徴がある。

本書の最後ではこれらの長所と短所を補い合うベンチマークソフトウェアを作成し，両者が連携して動作するシステムの実例に触れていく。

1.4.2　ソフトウェア開発: コンパイラ言語とスクリプト言語

コンピュータ上で動作するソフトウェアは例外なく何かのプログラミング言語で記述されたソースコード (Source code, Source Program) から生成されたものである。目的や時代要請に応じて多種多様なプログラミング言語が存在するが，ソースコードをどのようにして解釈し，アプリケーションソフトウェアとして動作させるか，という方式で分類するとコンパイラ (compiler) によるものと，インタプリタ (interpreter) によるものの二つが存在する。この 2 つはソースコードをコンピュータに解釈させるネイティブアプリケーションソフトウェアだが，実行の形態が大きく違う (図 1.8)。

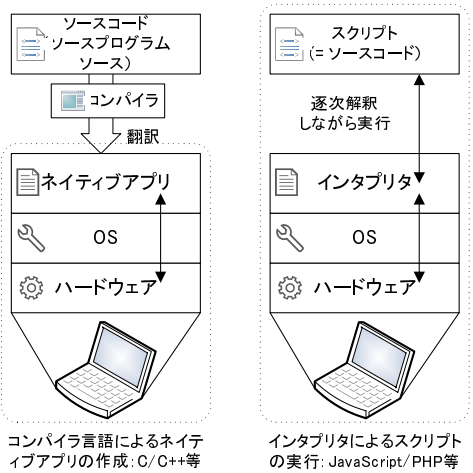

図 1.8　コンパイラ言語によるネイティブアプリの開発 (左) とインタプリタによるスクリプト言語の実行 (右)

コンパイラは，ソースコードを CPU が直接解釈できる機械語 (machine code) に変換するものと考えてよい。Java のようにソフトウェアの仮想マシンが解釈するコードに変換

するものもあるが，ソースコードを一括してバイナリ実行ファイルに変換させるという点では変わらない。コンパイラを使うことが一般的なコンピュータ言語をコンパイラ言語と呼ぶ。本書では C 言語を用いるが近年では発展形である C++ が多いようである。

インタプリタは，ソースコードを実行するたびに一行ずつ解釈して動作させる。そのためコンパイラによるバイナリ実行ファイルより動作が遅くなりがちである。文字列操作，データベース操作のように，プログラムの実行速度があまり問題にならない処理に用いられる。もっぱらインタプリタによって実行されるソースコードをスクリプト（Script) と呼び，スクリプトを記述する言語をスクリプト言語と呼ぶ。本書では Web アプリケーションをこのスクリプト言語で記述する。言語としては PHP を使用する。

コンパイラ言語，インタプリタ言語，それぞれ短所長所があるが，文法的には C/C++ と似かよっている。言語が提供する機能も似てきており，もはや言語間の相違はあまり問題ではなく，むしろその言語が提供されているインタプリタ・OS・Web などの環境，そして言語を使用するユーザコミュニティの提供するソースコードの蓄積でどのようなソフトウェアが実現されているか，という要素の方が大きいと言える。

課題 A

Windows 7 のコマンドプロンプト (スタートボタン→「すべてのプログラム」→「アクセサリ」→「コマンドプロンプト」）から，「ドキュメント」フォルダに移行し (エクスプローラから「ドキュメントフォルダ」を Shift キーを押しながら右クリック→「コマンドウィンドウをここで開く」でも可)，“unix00”フォルダを作成せよ。その結果を，エクスプローラ画面とコマンドプロンプト画面から切り出して Word ファイルに貼り付け，A4 用紙 1 枚に印刷し，学籍番号・氏名を明記の上，提出せよ。

課題 B

1. PC 向けの代表的な OS としては Linux 以外に Microsoft の Windows が存在する。Windows の歴史を調べ，Linux とどのように異なっているのかを簡潔に説明せよ。
2. UNIX と Linux の違いについて，簡潔に説明せよ。
3. Android の特徴と，ネイティブアプリケーションの開発方法について調べ，簡潔にまとめよ。

第 1 章の学習チェックリスト

- □ コンピュータの基本的な仕組みと，OS の役割を第三者に説明することができる。
- □ Linux とはどういうものかを第三者に説明することができる。
- □「ネイティブアプリケーション」と「Web アプリケーション」を第三者に説明することができる。
- □ アプリケーションの開発方法を，「コンパイラ」と「インタプリタ」という用語を使って説明することができる。