ソケットを用いたネットワークプログラミング実習

1. はじめに

1.1. 実験の概要

授業科目「ネットワーク実験」の１課題として，ソケットを用いたクライアント/サーバプログラミングの実習を行い，ネットワークアプリケーションプログラミングの基礎を学習する．

1.2. 実験の内容

実験は 4 週間にわたって行う．前半の 2 週で，TCP/IP の基礎の復習とコネクションレス型ソケットを用いたクライアント/サーバプログラミングの実習を行う．また，後半の 2 週で，コネクション型ソケットを用いたクライアント/サーバプログラミングの実習を行う．

〔前半 2 週〕

1) TCP/IP プロトコルの基礎(復習)
2) コネクションレス型クライアント/サーバプログラミングの実習

〔後半 2 週〕

3) コネクション型クライアント/サーバプログラミングの実習

1.3. 到達目標

ソケットを用いたネットワークアプリケーションプログラミングの基礎技術を習得する．

1.4. テキスト

このテキストの他に，さらに詳しい資料，例題プログラム，演習問題等がWEBに用意されている．実験の授業では，これらWEB教材(http://home.soka.ac.jp/~matsumi/ne/)を利用する．

1.5. 参考書

1) 基礎からわかる TCP/IP　ネットワークコンピューティング入門，村山公保著，オーム社，¥2,200.
2) TCP/IP ソケットプログラミング　C 言語編, M.J. Donahoo, Kenneth L. Calvert 共著, 小高知宏監訳，オーム社．¥1,800.

2. ネットワークアプリケーションと TCP/IP

2.1. クライアント/サーバモデル

TCP/IP プロトコルに従うネットワークアプリケーションの典型的モデルは，クライアント/サーバモデルである．クライアント/サーバモデルでは，クライアントからの要求に対してサーバがサービスを提供する．クライアント/サーバモデルは，クライアントとサーバとの間に通信に先立って論理的な接続を確立するか否かにより，コネクション型とコネクションレス型という 2 つのタイプに分けられる．

2.2. TCP/IP プロトコル階層モデルとソケット

TCP/IP プロトコル階層モデルと OSI 参照モデルの関係を図 1 に示す．TCP/IP プロトコル階層モデルは，コンピュータへの実装に主眼をおいているため，各層での処理が重複しないように OSI 参照モデルに比べて階層数が少なくなっている．

<table>
<tr><td>アプリケーション層</td><td rowspan="3">アプリケーション層</td></tr>
<tr><td>プレゼンテーション層</td></tr>
<tr><td>セッション層</td></tr>
<tr><td>トランスポート層</td><td>トランスポート層</td></tr>
<tr><td>ネットワーク層</td><td>インターネット層</td></tr>
<tr><td>データリンク層</td><td rowspan="2">ネットワーク
アクセス層</td></tr>
<tr><td>物理層</td></tr>
<tr><td>OSI 参照モデル</td><td>TCP/IP プロトコル階層モデル</td></tr>
</table>

図 1　OSI 参照モデルと TCP/IP プロトコル階層モデル

TCP/IP プロトコル階層モデルにおいて，アプリケーション層は，アプリケーションごとの通信サービスのインタフェースを規定する．アプリケーション層のプロトコルには，SMTP(Simple Mail Transfer Protocol)，HTTP(Hyper Text Transfer Protocol)，FTP(File Transfer Protocol)，TELNET 等がある．トランスポート層のプロトコルには，TCP(Transmission Control Protocol)と UDP(User Datagram Protocol)がある．TCP はコネクション型のクライアント/サーバモデルにおいて，UDP はコネクションレス型のクライアント/サーバモデルにおいて用いられる．トランスポート層は，アプリケーション層のアプリケーションの識別，データの信頼性の保証，そして，コネクション型のクライアント/サーバモデルに対しては，クライアント/サーバ間のコネクションの確立と切断，及び通信の信頼性の保証といった役割を果たす層である．インターネット層は，ネットワークとホストの識別，経路制御等の役割を果たす層である．インターネット層のプロトコルには，IP(Internet Protocol)，ICMP(Internet Control Message Protocol)，ARP(Address Resolution Protocol)，RARP(Reverse Address Resolution Protocol)等がある．

ソケット API(Socket Application Programming Interface)は，アプリケーションに対して，異なるホスト間での通信をサポートするためのトランスポート層以下の機能を提供するプログラミングインタフェースで，クライアント/サーバモデルを実現するために用いられる. 図 2 にクライアント/サーバ, TCP/IP プロトコル階層とソケット API の関係を示す.

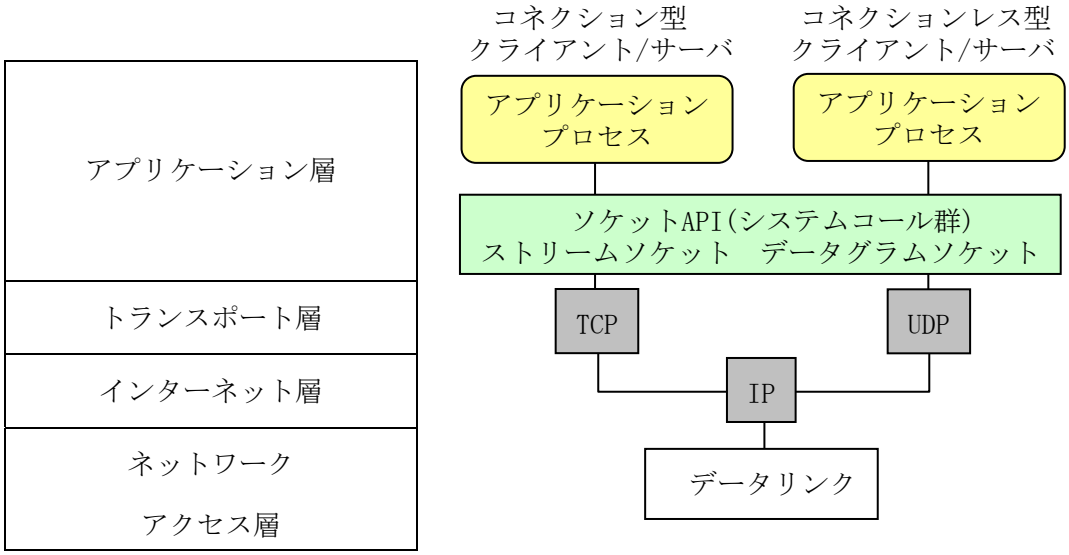

図 2　TCP/IP プロトコル階層，クライアント/サーバとソケット API

コネクション型のクライアント/サーバモデルの実現には，ストリームソケットと呼ばれるタイプのソケットが，コネクションレス型のクライアント/サーバモデルの実現には，データグラムソケットと呼ばれるタイプのソケットが用いられる.

2.3.　トランスポート層

2.3.1.　ポート番号

トランスポート層の役割の 1 つに，アプリケーション層のどのアプリケーションとデータのやり取りをすればよいかを指示することがある．これは，サーバホストにおいては要求するサービスを提供するサーバアプリケーションを識別することであり，クライアントホストにおいては，サービスを要求したクライアントアプリケーションを識別することである．これらアプリケーションの識別には，ポート番号と呼ばれる 16 ビット(0～65535 の範囲)の整数が用いられる.

ポート番号は，その用途により大きく次の 3 つの区分に区分けされる.

1) The Well Known Ports(0~1023) : 特に良く利用されるアプリケーションを識別するために予約されている番号でウェルノウンポート番号と呼ばれる．どのサーバホストも既定のサービスはウェルノウンポート番号に従って提供する.
2) The Registered Ports(1024~49151) : 特定のアプリケーションに割り当てられているこ

ともあるが，他のアプリケーションに割り当ててもよいことになっている番号をいう．最近のサービスの種類の増加により，この範囲の番号でもサービスとして正式に登録されているのもが増えている．

3) The Dynamic and/or Private Ports(49152~65535)：クライアントのポート番号として動的に割り当てたり，独自のアプリケーションのポート番号として自由に使ってよい番号である．ただし，慣例的には，Windows 系 OS では 1024 番以降，UNIX 系 OS では 8000 番以降の，その時点でクライアントホストが使用していないランダムな番号がクライアントのポート番号として動的に割り当てられる．

2.3.2. TCP と UDP

トランスポート層のプロトコルには TCP(Transmission Control Protocol)と UDP(User Diagram Protocol)がある．TCP プロトコルに従う通信では，クライアントとサーバとの間でコネクションを確立した後でデータの通信を行う．コネクションとは，クライアントとサーバ間でデータを流すために用いる仮想的な通信路をいう．TCP では，TCP セグメントのシーケンス番号と確認応答によりデータが確実に相手に届けられることを随時確認するため通信の信頼性が保証される一方，コネクションの確立・切断の制御，送達確認等にパケットが使われるために通信効率は低くなるという特徴を有する．UDP プロトコルに従う通信では，コネクションを確立することなくクライアントとサーバとの間でデータの通信を行う．UDP では，データが相手に到着することを確認しないため通信の信頼性は保証されないが，データ以外の制御・確認用パケットを送受信しないので通信効率は高くなるという特徴を有する．

2.4. インターネット層

2.4.1. IP アドレス(IPv4 アドレス)

インターネット上の個々のホストを識別するために用いられる 32 ビットのビットパターンを IP アドレスという．IP アドレスは，図 3 に示すように，通常 4 つの 10 進数をドット(.)で区切った形式で表記される．各 10 進数は 8 ビットに対応しており，0~255 の範囲の整数値をとりうる．ただし，「0.1.2.3」のように先頭が 0 で始まるアドレスは利用できない．

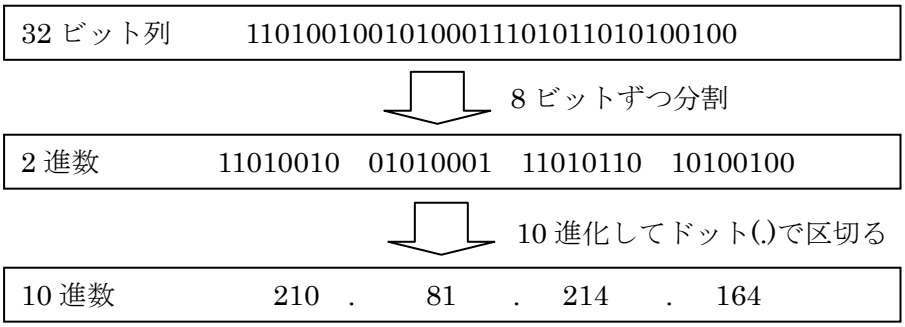

図 3　IP アドレスの表記方法

IP アドレスはホストに割り当てられるが，より正確にはホストとネットワークの接点であるネットワークインタフェース(NIC: Network Interface Card)ごとに割り当てられる．

2.4.2. IP

IP アドレスは，図 4 に示すようにネットワーク部とホスト部から構成される．ネットワーク部は，ホストが所属するネットワークを識別するネットワークアドレスを表す．ホスト部は，そのネットワークアドレスで識別されるネットワーク内でホストを識別するためのアドレスを表す．

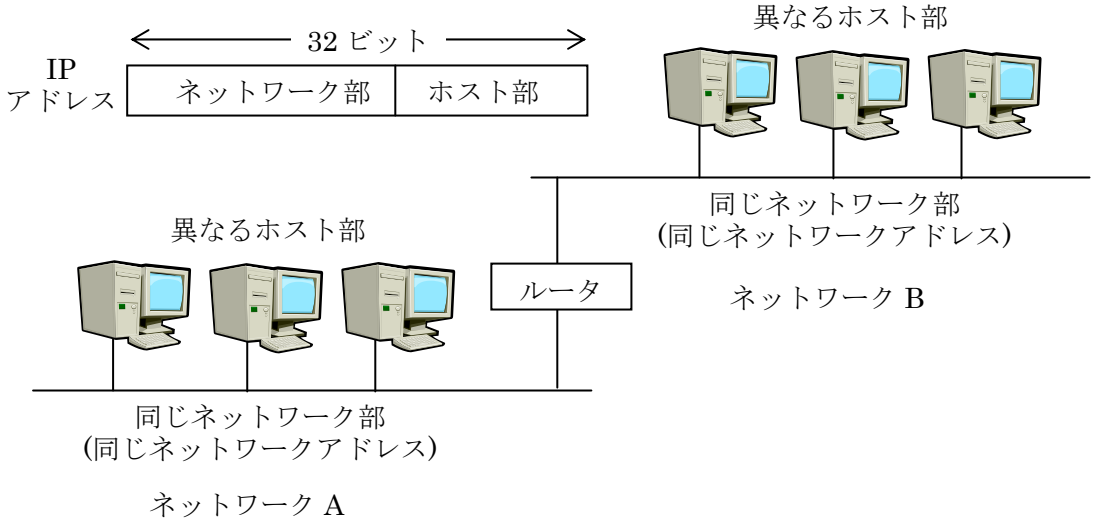

図 4　IP アドレスのネットワーク部とホスト部

インターネット層の最も重要なプロトコルが IP(Internet Protocol)である．IP は，宛先を示す情報(IP アドレス)をデータにつけたり，別のネットワークのホストにデータを届けたりする場合の経路制御を行う．

3.　ソケットを用いたクライアント/サーバプログラミング

3.1.　クライアント/サーバモデルと TCP/IP

クライアント/サーバモデルは，ネットワークアプリケーションにおけるプログラム(プロセス)間の相互作用の標準的モデルである．サーバは，クライアントからの要求を受け入れ，それに対するサービスを実行し，結果を返すプロセスである．クライアントは，ある処理を，サーバに要求を送りそれに対する結果を受け取ることにより実行するプロセスである．クライアントとサーバ間の通信は，クライアントの IP アドレスとポート番号，サーバの IP アドレスとポート番号により識別される．このとき，IP アドレスはホストを識別し，ポート番号はプロセスを識別する．図 5 にクライアントとサーバの関係を示す．

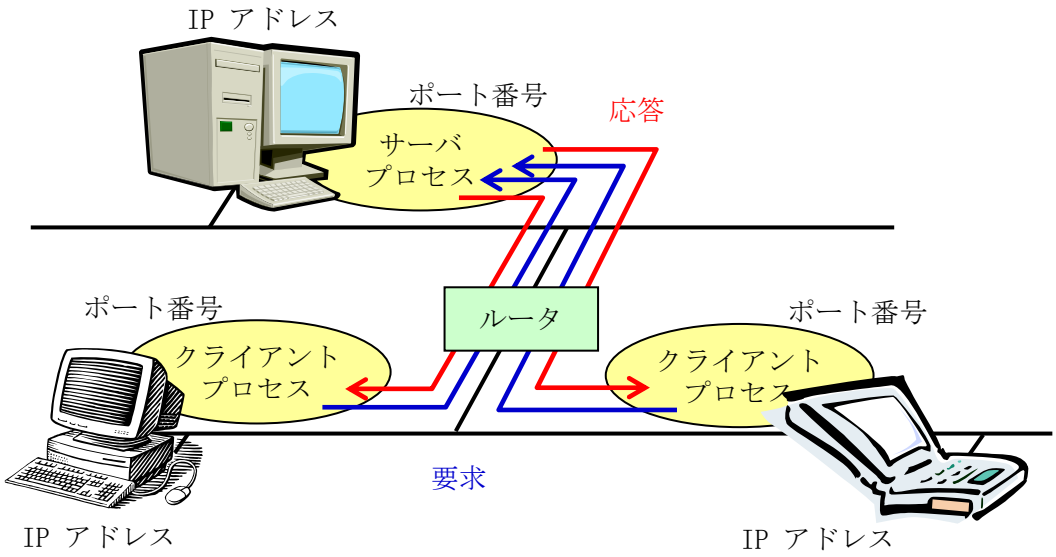

図 5　クライアント/サーバモデル

3.2.　サーバとクライアントの動作

サーバとクライアントは次のように動作する．まず，サーバは，通信チャネル(ソケット)をオープンし，自らのアドレス(IP アドレスとポート番号の組をアドレスと呼ぶ)を OS に登録する．即ち，登録したアドレスに届いたメッセージをオープンした通信チャネルに渡すように OS に依頼する．次に，そのアドレスにおいて，クライアントからの要求が到着するのを待ち，受け入れた要求を処理して結果をクライアントに返すことを繰り返す．クライアントは，通信チャネル(ソケット)をオープンし，あるホストのサーバ(アドレス)に要求を送る準備をする(例えば，自らのアドレスを登録したり，接続を確立したりする)．そして，サーバに要求を送り処理結果を受け取り，必要な処理が終わったならば，通信チャネルをクローズして終了する．

3.3.　サーバのタイプ

サーバは並行サーバと反復サーバの 2 つのタイプに分けられる．並行サーバは，クライアントからの要求に対して，fork システムコールにより自らのコピーである子プロセスを生成して，その子プロセスに要求の処理を任せる．子プロセスは，要求を処理し，クライアントとの通信チャネルをクローズして終了する．一方，親プロセスであるサーバは，別のクライアントからの要求を待つ．並行サーバは，要求の処理に要する時間が不明で，要求によって処理時間が変わりうる処理を並行的に行う必要がある場合に用いられる．反復サーバは，クライアントからの要求に対して，自らがその処理を行って結果をクライアントに返す．反復サーバは，あらかじめわかっている短い時間で処理可能な要求を逐次的に処理する場合に用いられる．

3.4. ソケット

ソケットAPI(Socket Application Programming Interface)は，アプリケーションに対して，異なるホスト間の通信をサポートするためのトランスポート層以下の機能を提供するプログラミングインタフェースで，クライアント/サーバモデルを実現するために用いられる．APIを構成するシステムコールには，socket()，bind()，listen()，accept()，connect()，send()，recv()，sendto()，recvfrom()，close()等がある．これらシステムコールの詳細は，WEB教材(http://home.soka.ac.jp/~matsumi/ne/)で解説されている．

ソケットの種類には，ストリームソケットとデータグラムソケットの 2 種類がある．ストリームソケットは，TCP プロトコルに従うコネクション型の接続形態，即ちクライアントプロセスとサーバプロセスの間に通信に先立って論理的な接続を確立する信頼性の高い通信を提供するのに用いられる．コネクション型の接続形態はバーチャルサーキットとも呼ばれる．データグラムソケットは，UDP プロトコルに従うコネクションレス型の接続形態，即ちデータグラムと呼ばれるメッセージを単位に行われる送信効率の高い通信を提供するのに用いられる．コネクションレス型の接続形態はデータグラムとも呼ばれる．

並行サーバに対してはコネクション型の接続形態，反復サーバに対してはコネクションレス型の接続形態が用いられることが多い．

3.5. コネクションレス型クライアント/サーバモデルの実装の枠組

ソケットを用いたコネクションレス型のクライアント/サーバモデルの実装の枠組を，基本的なシステムコール呼び出しの流れにより図 6 に示す．これらシステムコールを用いた実際のプログラミング，及びネットワークプログラミング独自の注意すべき点については，具体的な例題を用いてWEB教材(http://home.soka.ac.jp/~matsumi/ne/)で解説されている．

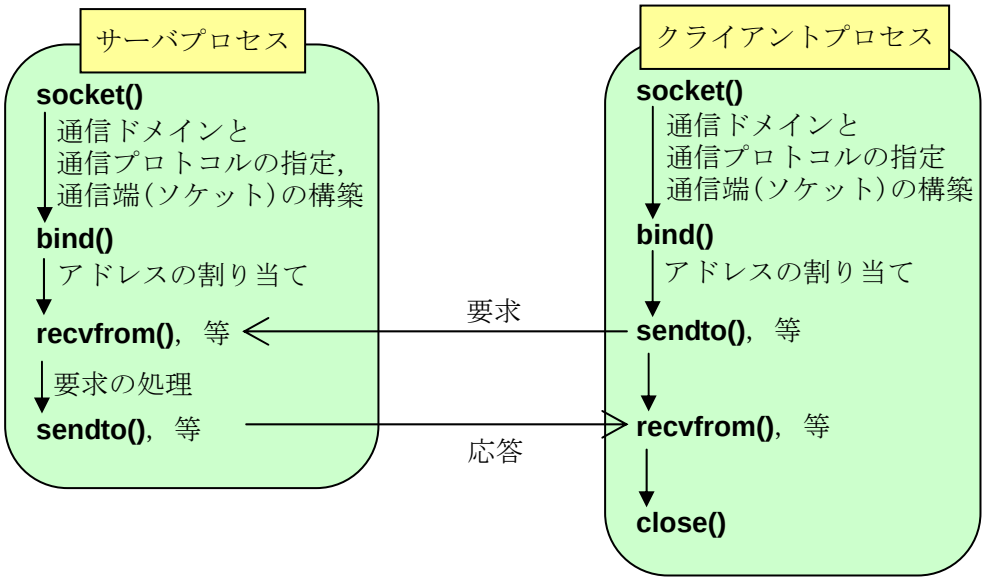

図 6　コネクションレス型クライアント/サーバの実装の枠組

3.6. コネクション型クライアント/サーバモデルの実装の枠組

ソケットを用いたコネクション型のクライアント/サーバモデルの実装の枠組を，基本的なシステムコール呼び出しの流れにより図 7 に示す．これらシステムコールを用いた実際のプログラミング，及びネットワークプログラミング独自の注意すべき点については，具体的な例題を用いてWEB教材(http://home.soka.ac.jp/~matsumi/ne/)で解説されている．

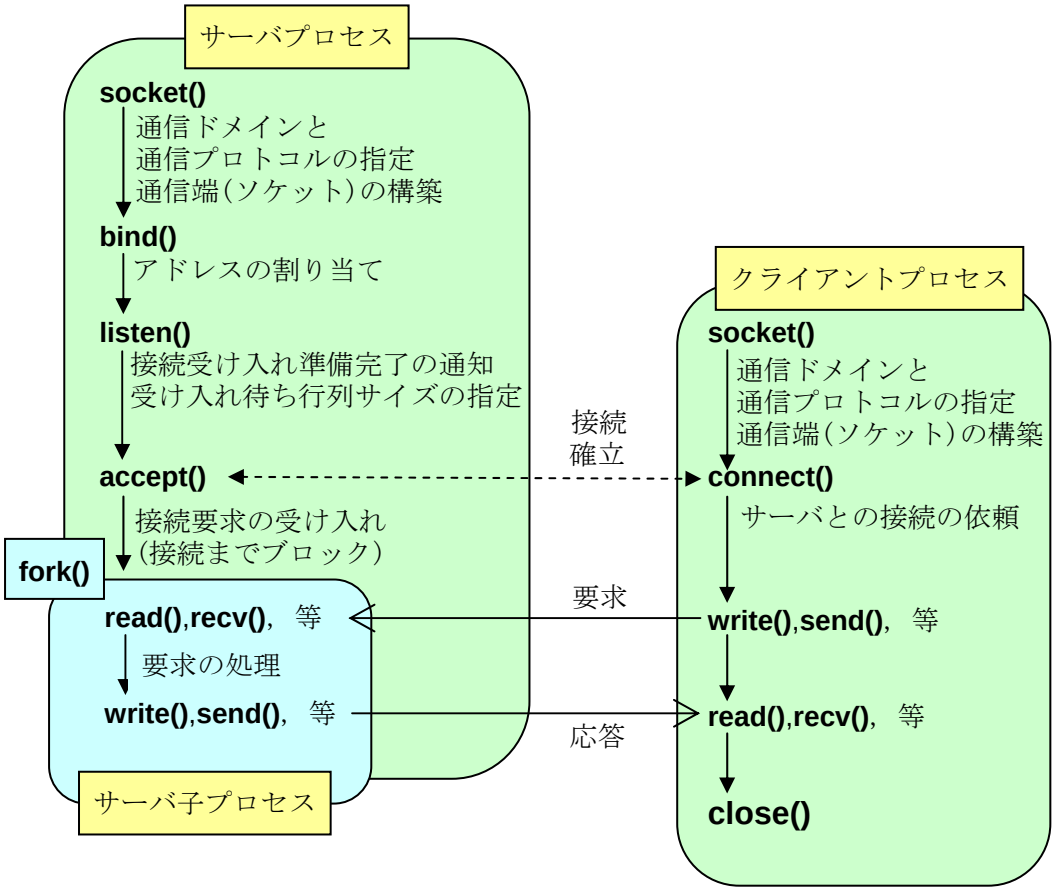

図 7　コネクション型クライアント/サーバの実装の枠組

4. 課題レポート

課題レポートは，前半 2 週の実験に対する課題レポートと後半２週の実験に対する課題レポートの２つからなる．

1) **課題レポート 1**：各自提出

- 内容：WEB テキスト(詳細版)の問題 1
- 実施日：前半 2 週の実験で行う
- レポート形式：テキスト(詳細版)を参照のこと
- 提出期限：第 3 週の実験日の朝 9 時
- 提出先：ポータル

2) **課題レポート 2**：各自提出

- 内容：WEB テキスト(詳細版)の問題 2
- 実施日：後半 2 週の実験で行う
- レポート形式：テキスト(詳細版)を参照のこと
- 提出期限：次の実験テーマの最初の実験日の朝 9 時
- 提出先：ポータル