コンピュータ基礎演習

第1回

理工学部 情報科学科 隅田 康明

sumida@ip.kyusan-u.ac.jp

今日の授業内容

- ガイダンス
  - 講義概要・到達目標、評価、受講上の注意
  - Processing最初の一歩
- コンピュータの基本操作
  - Office365からメールを送信
  - 署名作成
- レポート提出
  - メールでプログラム送信

自己紹介

- 名前：隅田　康明
- 役職：理工学部　情報科学科　講師

- 主にプログラミングの入門講義を担当

- オフィスアワー
  - ？？

自己紹介（２）

- 趣味
  - 軽い運動（ランニング・登山・他）
  - 研究、プログラミング
- 研究
  - 車いすナビゲーション、安全運転教育、交通流シミュレーション
- 不得意なこと：絵を描く（絵心ゼロです）
  - でも、Processingなら何とかしてくれる

授業の内容
Processingを用いたプログラミング

- Processingとは
  - プログラミングの統合開発環境の１つ
    - プログラミングの入門に適している
  - 視覚的な表現を(比較的)簡単に実現できる
    - 絵を描いたり、アニメーションを作れる
  - 描画に関するプログラムをシンプルに記述出来る
  - Webブラウザでも実行可能

プログラム・プログラミング

- プログラムとは
  - コンピュータに実行させる処理の記述

- プログラミングとは
  - プログラムを記述すること

- コンピュータで動くソフトウェアは、誰かが作ったプログラム
  - メモ帳、電卓、ワープロ、ゲーム、Webブラウザ…
  - スマートフォンのアプリも同じ

プログラミングはモノ作り

- 絵を描く、彫刻を彫る、模型を作る、料理を作る、曲を作る、映像を作る、ゲームを作る…
- プログラミングも同じ

- こんなものを作りたい！　　を実現する
  - 少しずつ完成イメージに近づけていく作業
  - 出来上がったときの達成感を体験しよう
  - 想像外のものが出来上がることも

到達目標

- コンピュータの基礎的な使用方法を習得する。
- デジタルアート制作の基本を習得する。
- プログラムの基本を理解し、雛形を元にしたプログラミングを行える。
- オフィスソフトウェアを用いて報告書を作成出来る。
- 一般的なコンピュータリテラシー、ネットワークリテラシーを身につける。

講義概要

- 本演習では Processingによるコンピュータグラフィックス制作を通して、コンピュータの基礎的な使用方法を学ぶ。演習では、Processingというプログラミング言語を用いて、プログラミングによるコンピュータグラフィックス制作を実践する。また、制作した作品についてのレポート作成を通して、ビジネス系ツールであるワードプロセッサー、プレゼンテーションツールの基本的な操作方法を学習し、ビジネスに必要とされる一般的な情報リテラシー、およびネットワークリテラシーを理解する。

  本授業科目は、芸術学部のDP2及びCP2に基づいて必要な技術・技能を習得することを目的として設置している。
  本授業科目は、芸術学部共通の基盤を構成するIT力育成科目として、1年次に配当されている。

※2年次以降でも履修可能

要約

- Processingを使ってCGを作成するための、基本的な知識と技術を身につける
  - Processing：プログラミング言語の1種

- Processingを使って作品を作り、作ったものをレポートで説明する
  - 作った作品を人に説明できるようになる
    - プログラム用語を使った説明を含む

講義予定：1～7

回	内容
1	ガイダンス 受講上の注意、コンピュータの基本操作、 キーボード入力によるプログラムの作成と実行
2	アニメーションと入力イベント処理 動く図形の描画、マウスやキーボード操作による図形描画
3	条件に応じた図形の移動や変化 条件分岐（１）：if文、if-else文の使い方
4	条件分岐を利用したプログラム 条件分岐（２）：当たり判定、簡単なゲームプログラミング
5	乱数の利用 プログラミングによるジェネラティブアート制作入門
6	繰り返しによる複数の図形描画 繰り返し（１）：for文の使い方
7	幾何学模様の描画 繰り返し（2）：繰り返しパターン模様の作成

講義予定：8～14

8	配列 配列の考え方、複数図形の描画と移動
9	座標変換 座標変換の考え方、座標変換による図形の移動
10	デジタルアート制作（１） オリジナルの作品制作：構想～制作
11	デジタルアート制作（２） オリジナルの作品制作：制作
12	デジタルアート制作（３） オリジナルの作品制作：レポート作成
13	応用 マルチメディア・3Dプログラミング
14	応用レポートの制作

※学生の適性や進捗状況により変更することがある

評価基準と単位数

- 単位数：2単位（選択）

- 評価基準（100点満点換算）
  - S：90点以上
  - A：80点以上90点未満
  - B：70点以上80点未満
  - C：60点以上70点未満
  - D：60点未満
  - E：出席回数不足など

評価方法（100点満点換算）

- 各回の小テスト：20点
  - 何度でも解きなおし可、解きなおしによる減点もなし（次回から）

- 演習点：80点満点
  - 各回のレポート：30点満点
  - 第12回のレポート：40点満点
    - 未提出なら不可
  - 第14回のレポート：10点満点
    - S評価を取りたいなら必須

オリジナルの作品制作・応用レポート

- **オリジナルの作品制作（第12回のレポート）**
  - 第9回までに作ってきたもののアレンジ
    - プログラムで静止画を描く
    - プログラムでアニメーションを動かす
    - プログラムでゲームを作る
      - ゲームもアート！ と言うと異論ある人もいそうですが
  - 第12回のレポートの40点＋2回分の演習点
    - **未提出の場合には必ず不可**
- 応用レポート
  - 13回の内容の応用または12回レポートの発展

出欠、小テスト・レポート締め切り

- 出席：教室のカードリーダー
- 小テスト：何度でも解きなおし可能
  - 中盤から、小テストで一定以上の点数を取らなければレポートを提出できなくする(かも)
- レポート：金曜日の18時まで
  - ただし、講義時間中の提出を推奨
  - 遅れた場合には減点
  - 遅れ提出が多い場合は評価をC以下に制限
  - 提出方法：Moodleから提出（第2回から）

カードリーダーでの出席

- 入室登録
  - 授業開始前10分～授業開始後0分
- 退出登録
  - 授業終了前15分～授業終了後10分
- 入退出で必ず2回ICカードリーダーに学生証をかざす

出席回数と成績評価

- 10回以上の出席が無ければ不可（E評価）

- やむを得ない事情で欠席する場合、
  それを証明する書類を提出し、
  小テスト・レポートを提出すれば出席扱いとする

なるべくPCを用意しておきましょう

- この講義は原則対面で実施
- ただし、感染対策のレベルによっては遠隔や半遠隔になることもあります。

- いつ遠隔授業になっても困らないように、
  なるべく授業で使えるPCは用意しておきましょう
  - ノートPC 1 台持ってると色々便利ですよ

履修上の注意

- 積み上げ式の講義なので、
  １回休んだだけで分からないことが雪だるま式に増えていくこともある

- 休んでも次回までに講義の動画を見て
  レポートを作成、提出しておくこと
  - 全て出席出来なくても、**レポートは全て出す！**

- これが出来ないなら、履修変更を勧めます
  - 大学の授業は全部そうですが

パソコン教室のPC利用の注意点

- 作業した内容は個人領域に保存
  - デスクトップ、ドキュメント等のフォルダの中身はPC再起動時に消去される
  - **レポート等は個人領域(Zドライブ)に保存する**

- 授業中に、各員のPCを教員が操作することがある
  - マウスやキーボードをロックする、画面を暗くする、ファイルを転送する、など

第1回演習内容解説
パソコン教室での作業を想定

Processingで静止画描画

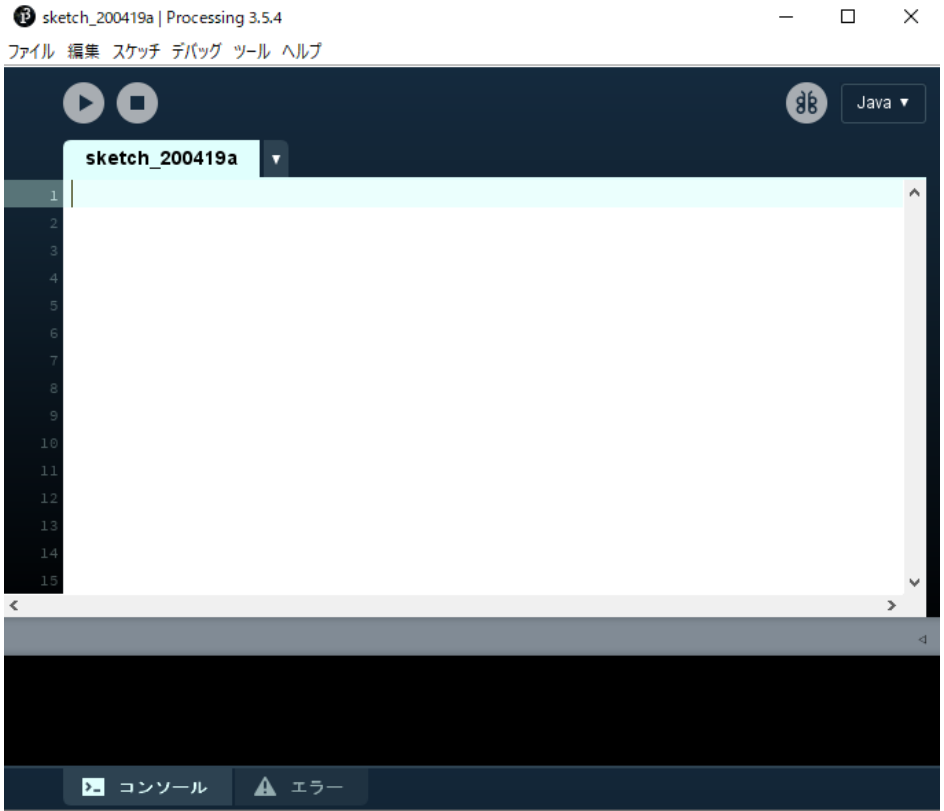
以降の作業での注意

- 分からない、上手くいかない場合は、
  すぐに手を挙げること

- 今分からない、は恥ずかしいことではありません
- 後で分かってなかった、ことが分かるとすごく困ります

Processingを起動

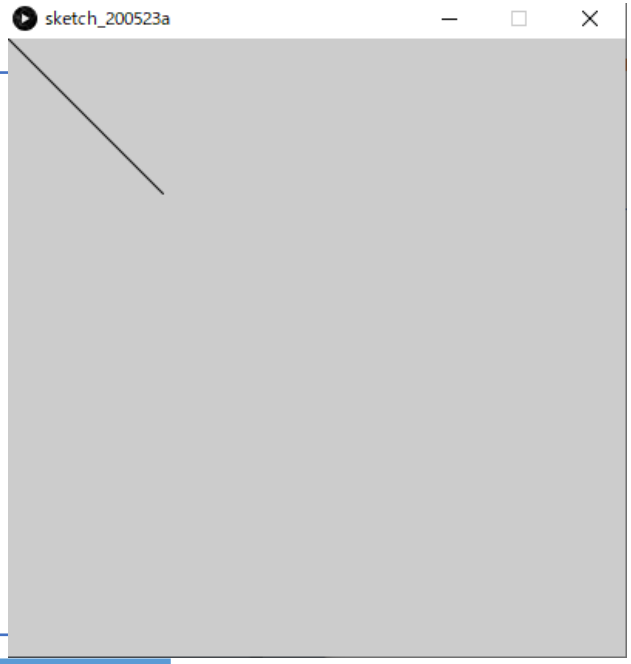
- Processingを起動
  - スタート→「プログラム開発」→「Processing」



Processing最初の一歩

- 下記のプログラムを入力して実行

```
void setup(){
  size(400,400);
   background(255 , 255 , 255 );
   line(0, 0, 100, 100);
}
void draw(){
}
```



線が一本引かれるのを確認したら次へ

線が引かれなければ打ち間違っている、よく見直そう

Processingの基本形

- まずはこれを書いてからプログラミングスタート
  - 絶対に書く決まり事と思っておけば良い

```
void setup() {


}

void draw() {


}
```

setupメソッド
{} の中に最初に1度だけ
実行する内容を記述する
動かさないならここだけ使えばOK

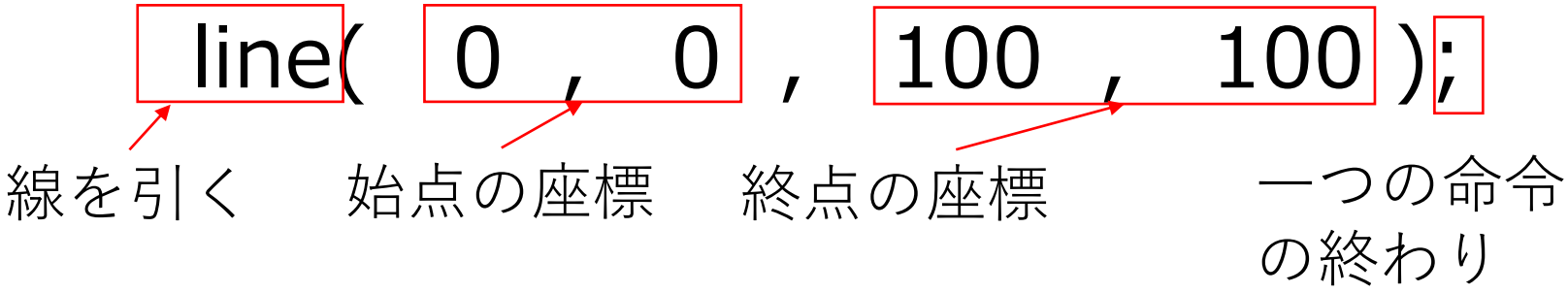
drawメソッド
{} の中に何度も繰り返し
実行する内容を記述する
アニメーションを作るときに使う

これを書かないと、図形を動かせない（静止画を書くだけなら、なくてもいい）

プログラムの説明

- size(横, 縦);
  - Processingを実行した時の、実行画面の大きさを設定する命令
  - 数値を変えて実行し、実行画面サイズが変わることを確認しよう
- background(赤,緑,青);
  - 背景色を指定して塗りつぶし
  - 色の3原色ではなく、光の3原色で指定
  - 3つの数値を 0~255の間で変えると背景の色が変わることを確認しよう

今書いたプログラムの解説



- 意味：(0, 0) から (100,100)をつなぐ線を引く

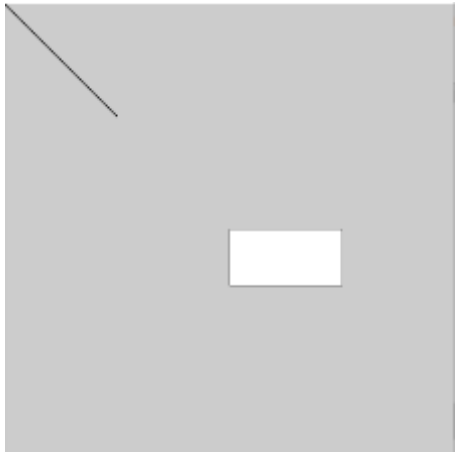
- 一つの命令を書いたら、[;](セミコロン)を記述
  - 文章を。で区切るようなイメージ
- 今回使う命令は、全て”メソッド”という種類の命令

命令には、それぞれ意味があり、厳格にルールが決められている

他の図形も描いてみる

- line(0, 0, 100, 100);　の下に長方形を描いてみる

  line(0, 0, 100, 100);
  rect(200,200,100,50);



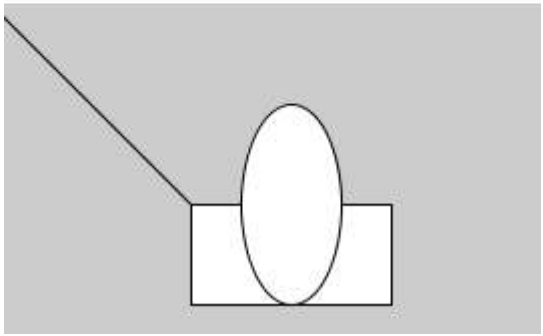
- rect(200,200,100,50);
  - rect(左上x,左上y,幅,高さ); は長方形を描く命令
  - (200,200)を左上に
    横100、縦50の大きさの長方形を描く
    - 縦の方が小さいので、横長の四角形になる

数値を変えて、図形の形や位置がどう変わるのか試してみよう

更に図形を追加してみる

```
sketch_200520a
1 void setup() {
2   size(400, 400);
3   line(0, 0, 100, 100);
4   rect(100, 100, 100, 50);
5   ellipse(150, 100, 50, 100);
6 }
7 void draw() {
```

sketch_200520a



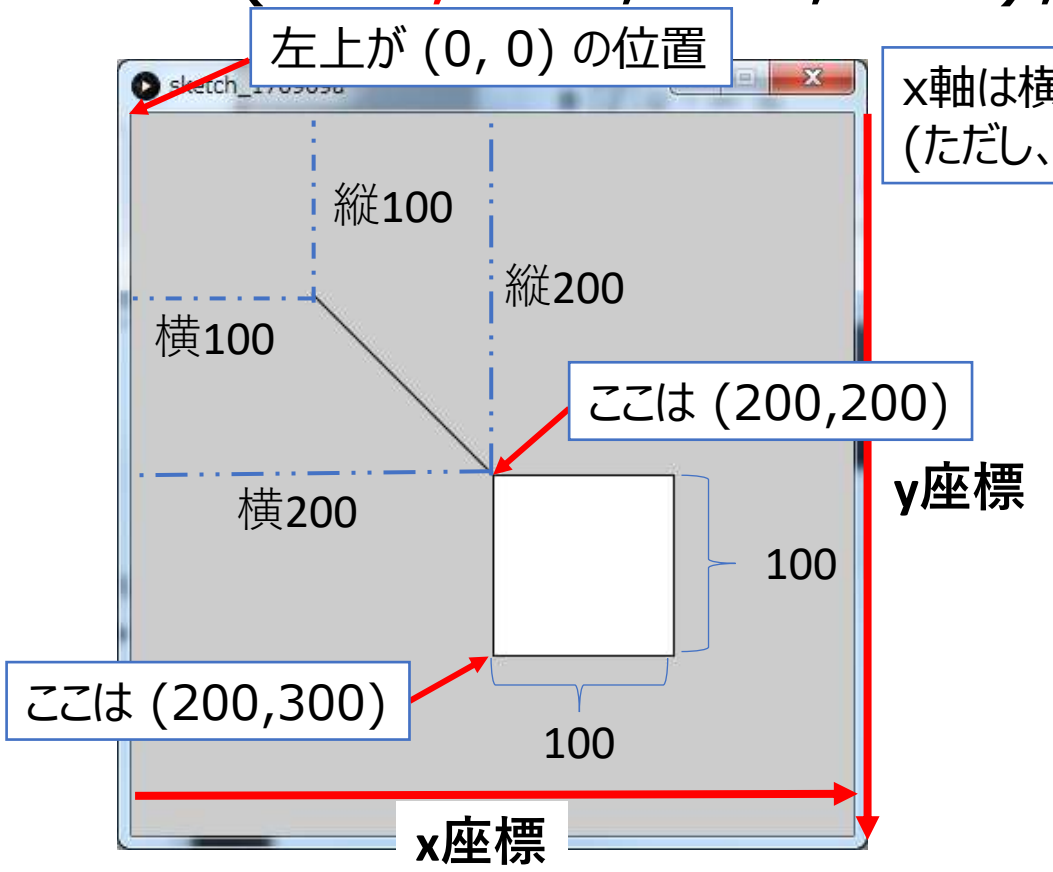
- 円を追加
  line(0, 0, 200, 200);
  rect(200,200,100,50);
  ellipse(150,100,50,100);

- **ellipse(150,100,50,100);**
  - ellipse(中心x,中心y,幅,高さ); は円を描く命令
  - (150,100)を左上に
    横50、縦100の大きさの長方形を描く
    - 横の方が小さいので、縦長の楕円になる

位置の指定：座標

```
line(100,100,200,200);
rect(200,200,100,100);
```

こんな命令をしたとする



x軸は横、y軸は縦 (数学のグラフと同じ)
(ただし、y の向きは逆(数値が大きいほど下)

一番左が x座標 0
一番上が y座標 0

Processingでは、
位置は0から数え始める

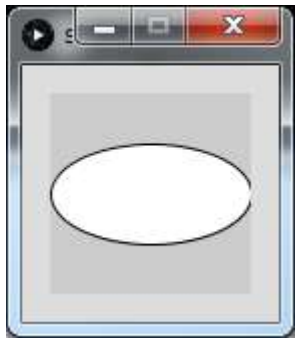
右に何歩、下に何歩と
移動してから描くイメージ

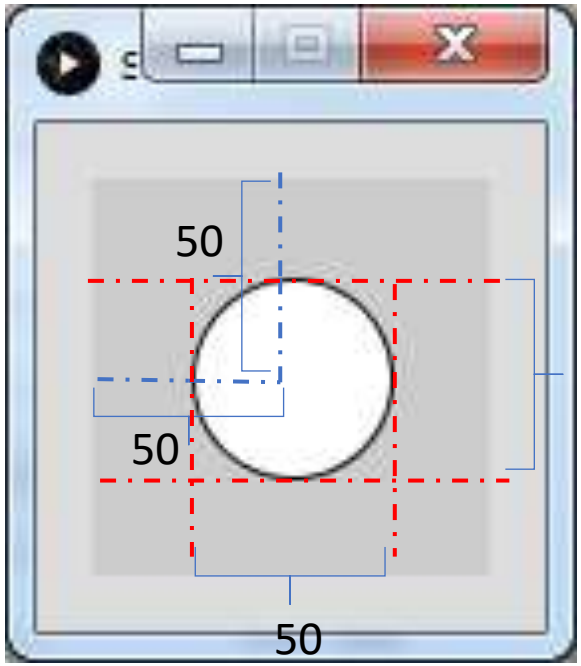
このプログラムは書かなくてもいい（既にあるellipseの命令を書き換えてみよう)

（例）楕円を描く命令

ellipse(中心のx,中心のy,幅,高さ);
ellipse(50,50,50,50);
(50,50)を中心に、幅が50、高さが50の円を描く

ellipse(50,50,100,50);
幅が100、高さが50の円＝楕円





位置をずらしたいなら

```
ellipse(50,50,50,50);

ellipse(70,50,50,50); //x座標を増やすと→に
ellipse(30,50,50,50); //x座標を減らすと←に

ellipse(50,70,50,50); //y座標を増やすと↓に
ellipse(50,30,50,50); //y座標を減らすと↑に
```

大きさを変えたいなら

```
ellipse(50,50,50,50);

ellipse(70,50,70,50); //幅を増やすと横が大きく
ellipse(70,50,30,50); //幅を減らすと横が小さく

ellipse(70,50,50,70); //高さを増やすと縦が大きく
ellipse(70,50,50,30); //高さを減らすと縦が小さく
```

命令にはそれぞれ意味がある

- line(100,100,100,100);
  rect(100,100,100,100);
  ellipse(100,100,100,100);
- カッコ内の数値は，(カンマ)で区切って4つずつ
  - 命令によって、いくつ値を書いていいかも決まっている
  - しかし、描かれる図形は全く違う
- line(始点x, 始点y, 終点x,終点y);
  rect(左上x,左上y,幅,高さ);
  ellipse(中心x,中心y,幅,高さ);

このプログラムは書かなくてもいい（既にあるellipseの命令を書き換えてみよう）

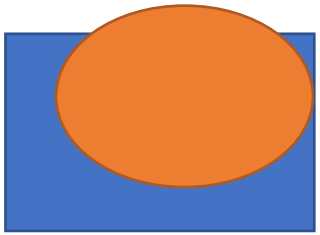
プログラムの逐次実行

- プログラムは１行目から順々に実行(例外もある)

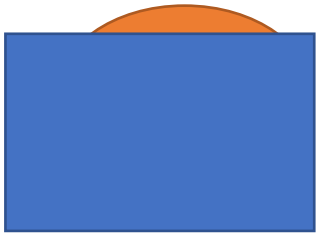
```
size(400,400);
line(100,100,200,200 );
ellipse(200,200,100,100);
rect(200,200,100,100);
```

画面の大きさを変える
線を引く
丸を描く
四角を描く

１.四角を描く
２.丸を描く



１.丸を描く
２.四角を描く

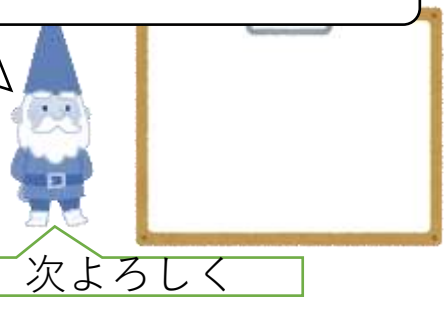


下に書いた命令の図形が、
上に重なっていく

上から順に、一つずつ実行していく

```
void setup(){
  size(400,400);
  line(0,0,50,50);
  rect(50,50,50,100);
}
```

(400,400) のキャンバス作ったよ



次よろしく

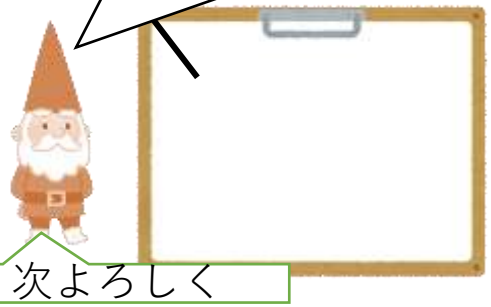
sizeさん

幅 高さ

(400,400)

幅と高さの2つの数値を受け取ると、
その幅高さの描画領域を作る

(0,0) から (50,50)に線を引いたよ



次よろしく

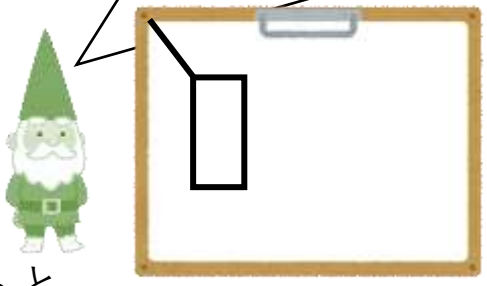
lineさん

(始点x, 始点y, 終点x,終点y)

(0,0, 50, 50)

始点座標と終点座標を受け取ると、
その2点を結ぶ線を描いてくれる

(50,50)に幅50, 高さ100の四角を描いたよ



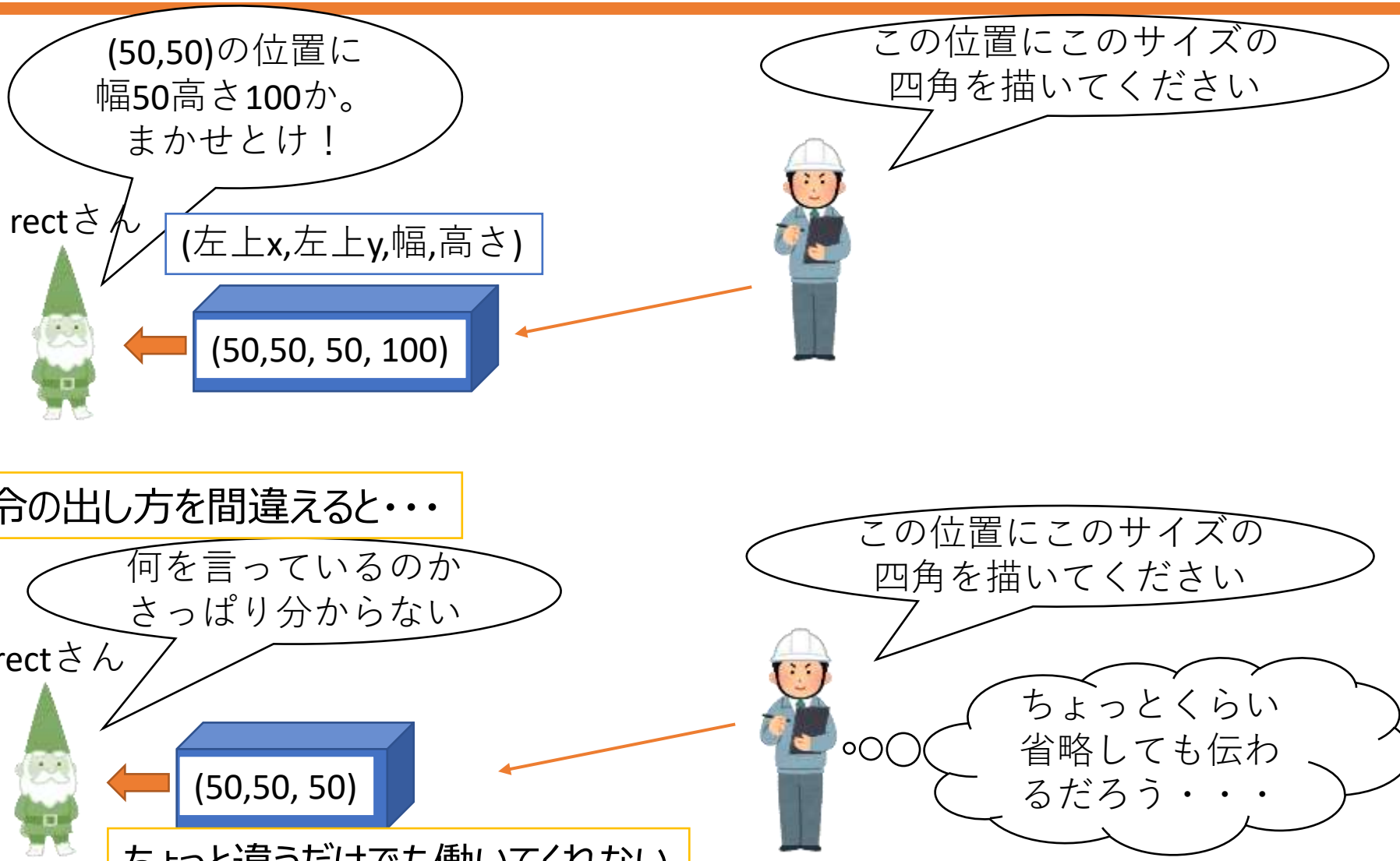
rectさん

(左上x,左上y,幅,高さ)

(50,50, 50, 100)

左上の頂点の座標と幅と高さを受け取ると、
左上座標を基点に、その幅と高さの長方形を描いてくれる

メソッドを書く(呼び出す)ときの決まり：ルールを守って命令を出そう



図形に色を付ける

- 何も指示しないと、線の色は黒色、塗りつぶしは白色になる
- 色を変えたければ、何色にするか**命令する**必要がある

- 線の色を変える命令：**stroke(赤, 緑, 青,透明度);**
  - それぞれ、0～255の数値で入力する
    - 透明度は数値が小さいほど薄い
- 塗りつぶしの色を変える命令：**fill(赤, 緑, 青,透明度);**
  - それぞれ、0～255の数値で入力する
    - 透明度は数値が小さいほど薄い
- 線を書かない：noStroke();
- 塗りつぶさない：noFill();

その他の命令（メソッド）
今回は、line,rect,ellipseだけでOK

- Processingには、
  ここまでに紹介した以外にも沢山の命令がある
- 次のページに、よく使う【図形を描く命令】を載せておくので、
  - 命令を実際に書いて、
  - 数値を変えるとどう変わるのか確かめながら、
  - 命令の意味を理解していこう

- 打ち間違い、カッコの中の数値の数、に特に気をつける

手を動かして、書いて、実行してみて、理解する。
最初は上手く行かなくてもいい、まず試してみよう！

命令の意味	命令文	例
HSBモードにする	colorMode(HSB, 360, 100, 100, 100);	
線の色を設定する	stroke(赤,緑,青);	stroke(255,0,0);
透明度も設定する	stroke(赤,緑,青,透明度);	stroke(255,0,0,128);
線の太さを設定する	strokeWeight(線の太さ);	strokeWeight(5);
塗り潰しの色を設定	fill(赤,緑,青);	fill(0,0,255);
透明度も設定する	fill(赤,緑,青,透明度);	fill(0,0,255,128);
線を引かない	noStroke();	
塗り潰さない	noFill();	
線を引く	line(始点x, 始点y, 終点x, 終点y);	line(100, 100, 200, 200);
長方形を描く	rect(左上のx,左上のy, 幅, 高さ);	rect(100,100,50,80);
三角形を描く	triangle(x1,y1,x2,y2,x3,y3);	triangle(10,50,30,40,70,90);
四角形を描く	quad(x1, y1, x2, y2, x3, y3, x4, y4);	quad(30, 30, 80, 20, 70, 60, 30, 80);
円を描く	ellipse(中心のx,中心のy, 幅, 高さ);	ellipse(100, 100, 50, 80);
円弧を描く	arc(中心x,中心y,幅,高さ,開始角,終了角)	arc(100,100,50,50,0,PI/2);
文字を書く	text("書きたい文字列",左上のx,左上のy)	text("Hellow",100,50)
文字の大きさを設定	textSize(フォントサイズ);	textSize(24);
点を打つ	point(x座標,y座標);	point(10,10);

図形を描く手順

1. strokeやfillで線、塗りつぶしの**色を指定**
2. **図形を描画**
3. この繰り返し

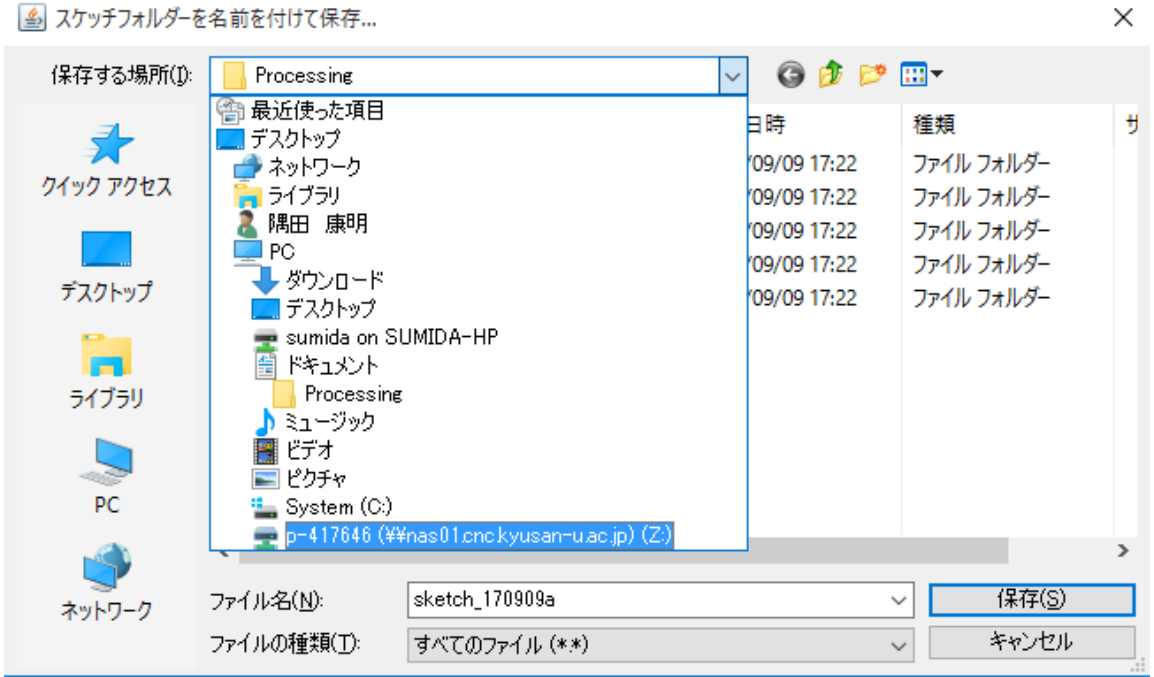
最初はこの３行を１セットと考えて図形を増やしていく

```
stroke(255,0,0,50);  //線の色を指定
fill(0,0,255,255); //塗りつぶしの色を指定
rect(10,10,100,50); //図形を描く
```

！重要！　スケッチを保存する

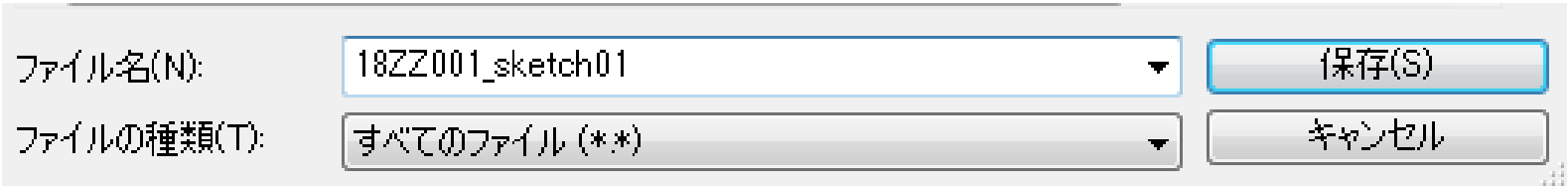
- 作ったプログラム(スケッチ)を保存する

1. [ファイル]メニューから[名前を付けて保存]
2. 保存先にZドライブを選択



- [Z:¥PC基礎演習¥sketch]を選択
  - フォルダ名は任意
  - Processingフォルダ内には保存しない
    （後日アップデートするかも知れないので）

- スケッチに名前を付けて保存
  ファイル名：学籍番号_sketch講義回



  21AA001_sketch01
  21AA001_sketch01-2 (2個以上になる場合)

他の色の指定方法

- Processingでの色の指定方法
  - RGBモードで指定：光の三原色 (赤,緑,青)で指定
    - デフォルトではこのモード。コンピュータの世界では、色はRGBモードモードが標準(ディスプレイがRGB)
      - 0～255で、各色の強さを指定
  - HSBモードで指定：色相(H)、彩度(S)、輝度(B)
    - ドロー系ソフトで絵を描く人は馴染みがある？
    - HSVやHSL(厳密には違う)などの呼び方も
    - 直感的に色を作りやすい
      - H：0～360で赤～黄～緑～シアン～青～紫～赤で１周する
      - S：0～100で、大きいほど鮮やかな色に
      - B：0～100で、大きいほど明るい色に
  - カラーコードで指定
    - HTMLではお馴染みの方法(考え方はRGBと同じ)
      - 色を変化させられないので、この授業での出番はない。紹介のみ。

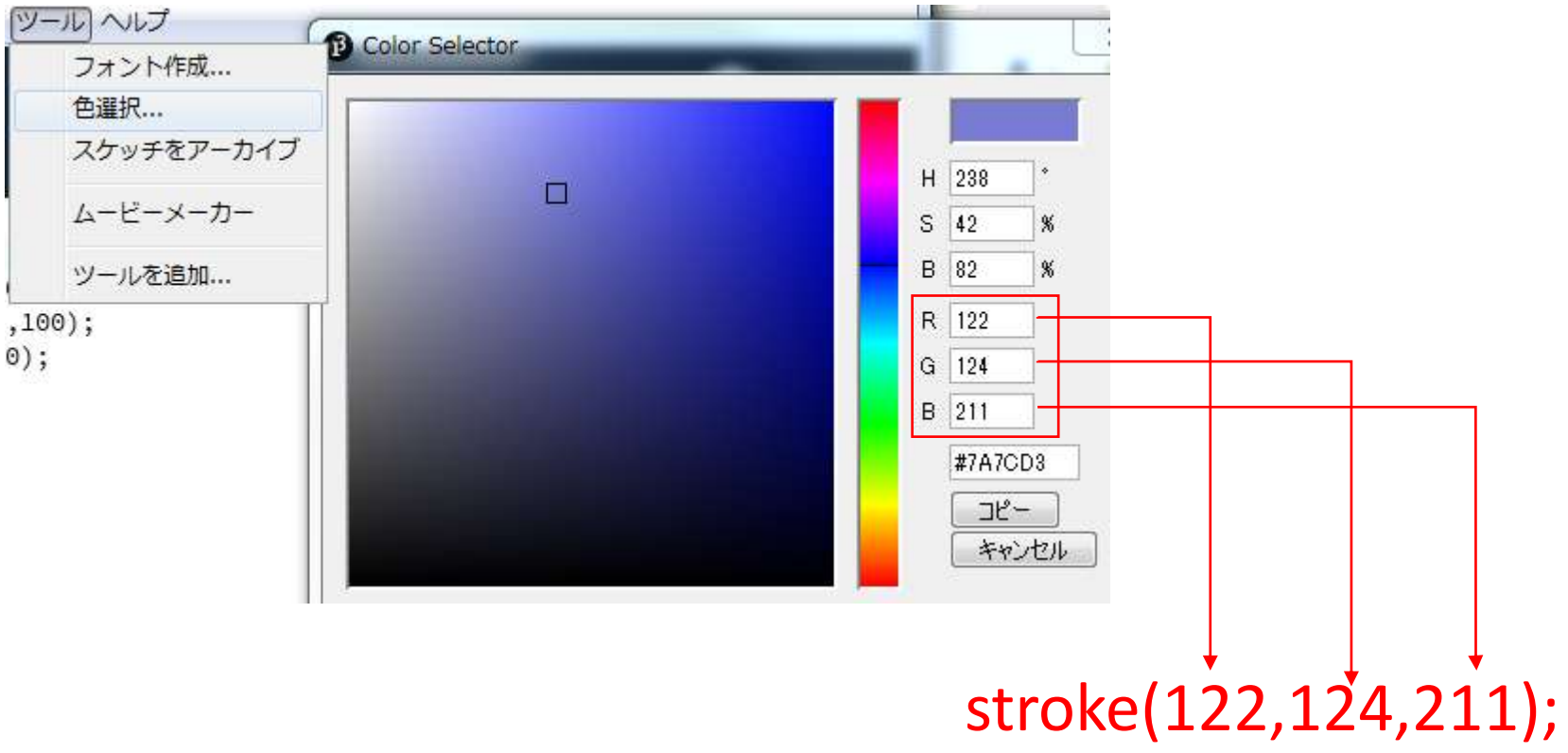
HSBモードでの色指定（今回は基本的にはRGBで書けばいい）

- **colorMode(HSB,360,100,100,100);** の命令を入れると、色の指定がHSBモードになる
  - colorModeは最後に実行したモードになる

```
colorMode(HSB,360,100,100,100);
fill(0,100,100);  //HSBでは赤色
ellipse(0, 0, 100, 100); //赤色の円
colorMode(RGB,255,255,255,100);
fill(0,100,100);  //RGBでは暗い緑色
ellipse(0, 0, 100, 100); //暗い緑の円
```

思い通りの色を作れない(PCの場合)
カラーセレクターで色を選ぶ

- RGBの数値だけだと、何色になるのか分かりにくい
- [ツール]メニューの「色選択…]



思い通りの色を作れない(その他の場合)
カラーセレクターで色を選ぶ

- オンライン上のツールを利用させてもらう
  - RGBとHSV・HSBの相互変換ツールと変換計算式 - PEKO STEP
    - https://www.peko-step.com/tool/hsvrgb.html

- 色の見本から選ぶ
  - **Webで使えるカラーネームと、そのカラーコード・RGB値一覧**
    - https://www.webcreatorbox.com/webinfo/color-name

プログラム入力時の注意

- 入力するプログラムは、全て半角英数
- **大文字小文字**、カッコの形、記号の形に気を付ける
  - 大文字、小文字は別の文字として区別される
  - 一文字違うだけでも動作しないこともあるので、打ち間違えないように気を付けること

- 間違えやすい記号
  - [,]と[.]　今回は、カンマ [,]のみ。 [.]は使わない
  - [;]と[:]　今回は、セミコロン[;] のみ

日本語入力(ひらがなモード) は使わない！ ！

第1回の基本形

```
//コンピュータ基礎演習 第1回テンプレート
void setup(){
  size(400,400); //実行画面のサイズ(横,縦)
  //ここからプログラムを追加していく



  //ここから下には何も書かない
}

void draw(){
  //第1回はここには何も書かない
}
//Enterキーで画像を保存するプログラム
void keyPressed(){
  if (keyCode == ENTER){
    saveFrame("image-####.png");
  }
}
```

プログラムで絵を描いてみよう

- line ， rect, ellipse の命令を組み合わせて、なにか意味のある形を作る
- 色を設定して、図形を描画

```
stroke(255,0,0,50);  //線の色を指定
fill(0,0,255,255); //塗りつぶしの色を指定
rect(10,10,100,50); //図形を描く
```

- 5つ以上の図形を描いて何かの形にしよう

- このプログラムを次回の演習で背景として利用する

高度なアレンジ

やる気があって物足りない学生向けの内容
（ここまでしなくても満点は取れる）

線を沢山書きたい、丸を沢山並べたい、といった場合の参考に
（100行のプログラムが3行になることもある）

線を並べる

- 繰り返し文を使うと、線や図形を簡単に沢山並べられる
  - 詳しい説明は後の回
    - 今は、よく分からないけど、数値を変えたら描けたで十分

縦に線を10本描く

```
for(int i = 1; i <= 10; i++){
  line(i*40, 0, i*40, 400);
}
```

横に線を20本描く

```
for(int i = 1; i <= 20; i++){
  line(0,i*20, 400, i*20);
}
```

□の数値を変えて色々試してみよう

繰り返し(for文)の構文

```
for(ループ変数の初期化; 条件判定; 更新) {
  繰り返したい処理;
}
```

- ループ変数の初期化;
  - 繰り返し条件に使う変数の宣言と初期化を行う
- 条件判定;
  - 繰り返しを続ける条件を書く
- 更新
  - 変数の値を変化させる処理
    - 基本的にはループ変数を変化させる

for文の例

```
for (int i = 0; i <= 5; i++) {
  rect(i*20, 10, 20, 20);
}
```

- ループ変数の初期化;
  `int i = 0;` 整数型の変数 i を初期値 1 で宣言
- 条件判定;
  `i <= 5;` i が 5 以下の間繰り返す
- 更新
  `i++` i に 1 を加算する (1ずつ増やす)

`i+=2` `i = i - 5` の書き方でもOK

ネットリテラシー

- メールの作法

メールの作法

- **宛先を間違えない**
  - 送る前に再確認するように気をつける
- **件名は具体的**に、簡潔に
  - 相手はまず件名を見る
  - 一目で用件が分かるように！
    - 件名が悪いと無視されてしまうかも
      - 「件名なし」は絶対ダメ！
- **本文も簡潔**に、相手が読みやすいように気を付ける
- **署名を付ける**
  - 誰から来たメールなのか、相手にしっかり伝える

宛先(To)、CC、BCCの使い分け

- 複数人に送るとき(1人相手なら[宛先]で)
  - **宛先　："あなたに宛てたメールです"**
  - **CC　　："あなたにも念のために送ります"**
    - CC：Carbon Copy（カーボンコピー）
    - 会話で例えると：一応、○○さんも聞いといてね
  - **BCC　：他の受信者に分からないように送信**
    - BCC：Blind Carbon Copy（ブラインド カーボンコピー）
    - 会話で例えると：１人１人にこっそり耳打ち
      - 同じ内容を全員知ってるけど、誰に送られたかは送信者以外分からない
    - 顧客に対して一斉送信、などの場合に使われる

署名

- メールの最後に付ける、名刺のようなもの
  - 署名があると、
    誰からのメールなのかはっきり分かる
    - 受け取った人が安心出来る
  - 就職活動では、適切な署名を付けること
- 署名に書くこと
  - 名前、所属、
    連絡先(電話番号、住所、メールアドレスなど)
    - **相手に知られたくない情報は書かないこと！**
    - 今回は、連絡先は不要

授業についての質問メールの書き方

[授業名(曜日時限)]についての質問	件名
～先生	誰宛か
[授業名(曜日時限)]を受講しています、 20AA999の九産太郎です。	何者か
（質問内容） -- 20AA999 九産太郎 九州産業大学　芸術学部　○○学科 1年	用件 署名

相手に丁寧に対応して欲しいなら、自分も丁寧に！

メールのマナーをしっかり身につける

- 就職活動では必須！
  - 失礼なメールを送るだけで落ちることも
- 働き始めてからも必須！
  - ビジネスの場ではメールでのやり取りが多い
- 学生のうちに、メールの扱いによく慣れておこう

今回のレポート提出

※今回はメールで提出、次回からはMoodleで提出
　今回メールで提出するのは、メールで質問する練習をするため

今回のレポート提出

- **教員にメールを送信**
  1. Office 365 OutLookにサインイン
  2. メールの**署名**を設定
  3. 本文にProcessingのプログラムを貼り付ける
     - メールで質問するための練習

- **次回、今回のプログラムを利用するので削除しないように！**
  - メールで送っておけば、バックアップにもなる

Office365にサインイン

- Microsoft Office365のWebサイトを開く
  - https://www.office.com/

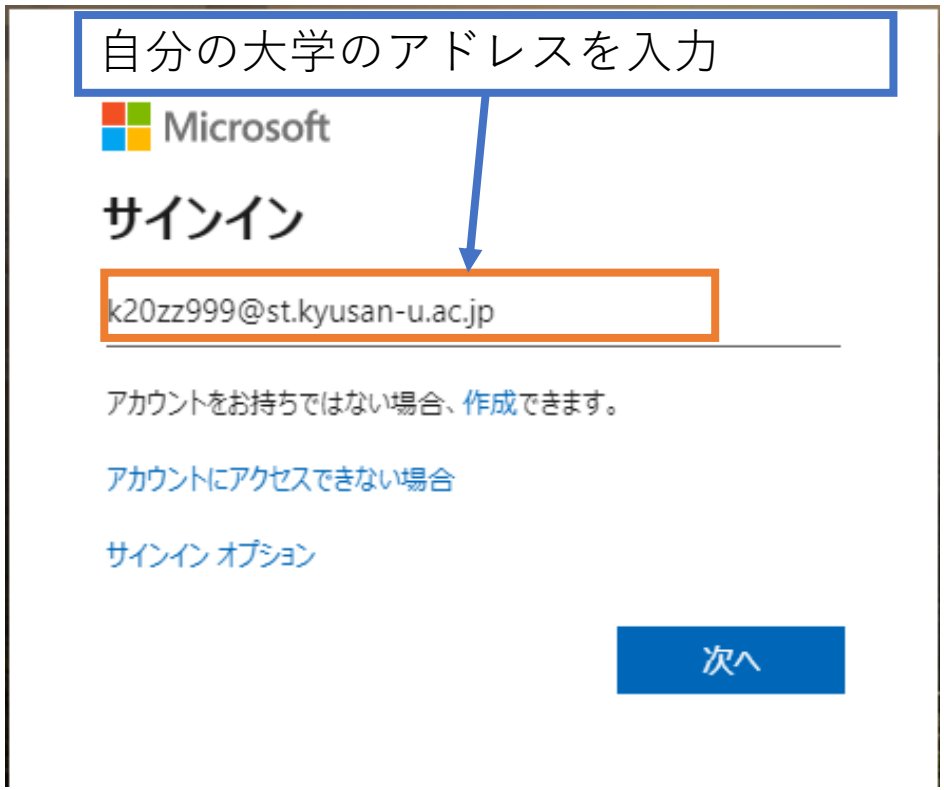
- Office365ログインをクリック

Office365にサインイン

- サインインをクリック

- メールアドレスを入力して「次へ」

メールアドレスを入力

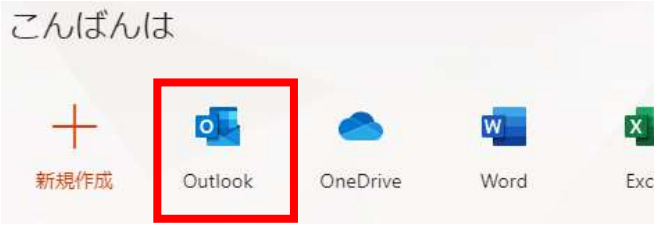


@以下があっていれば、大学ロゴのページに移動

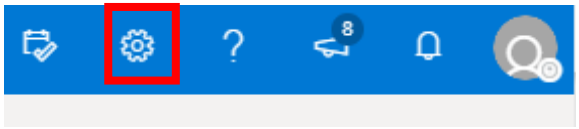
PCの場合

メール：署名の設定を開く

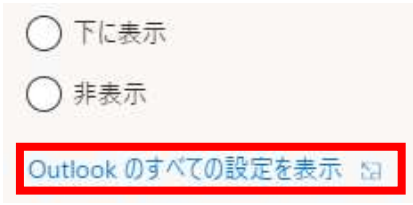
1.Office 365でOutlookを開く



2.右上の設定ボタンをクリック

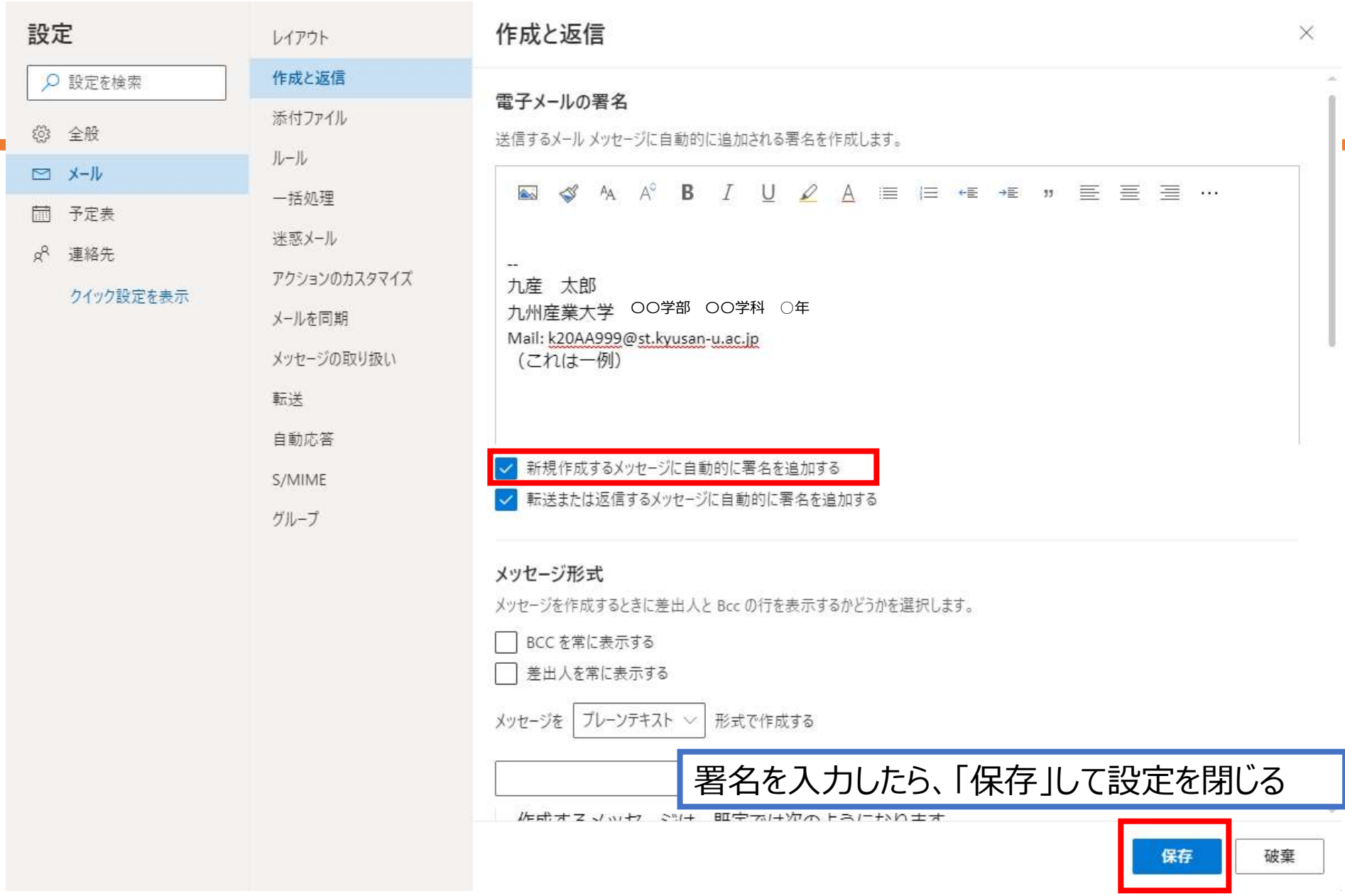


3.Outlookの全ての設定を表示



4.メール→作成と返信

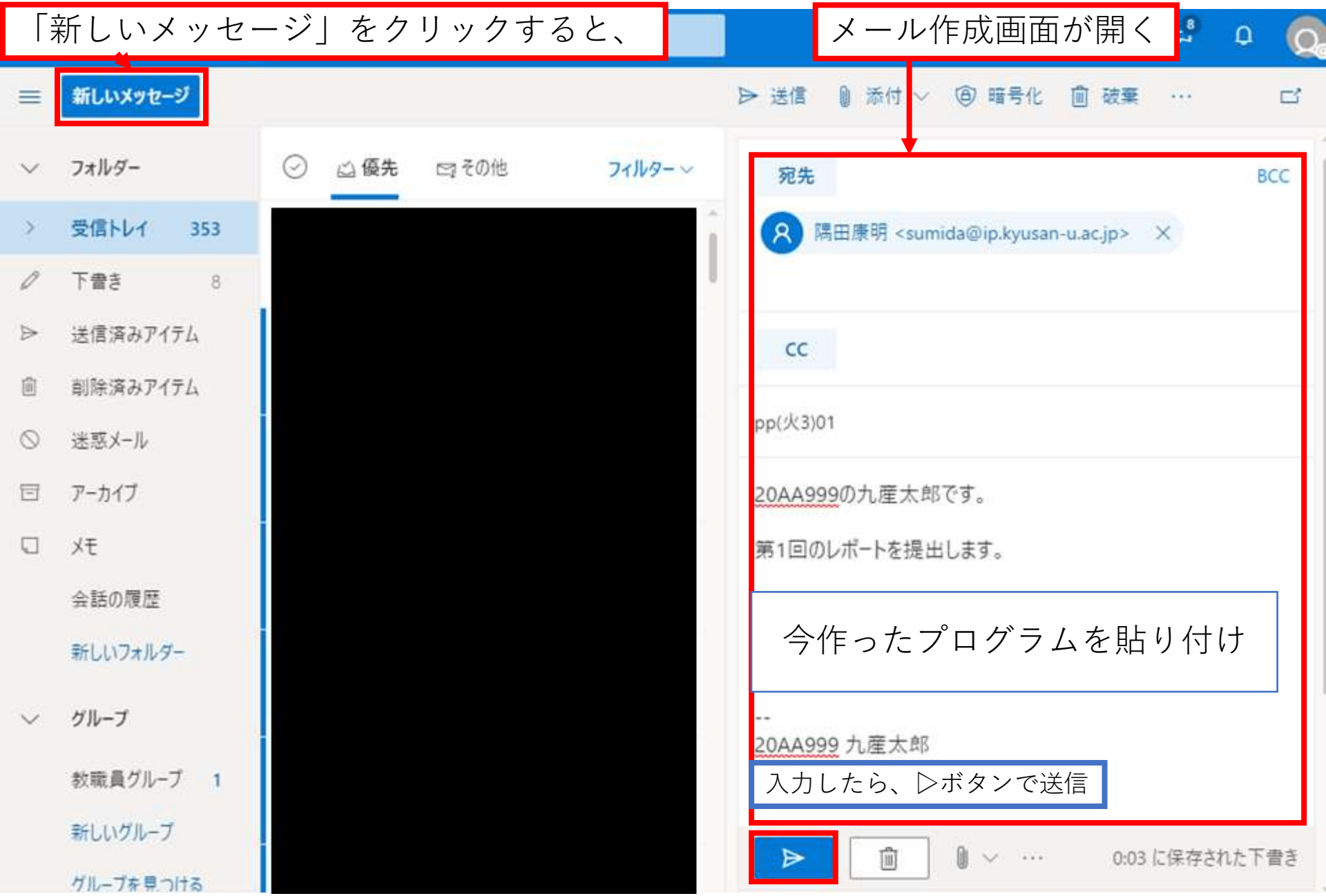
PCの場合


設定
設定を検索
全般
メール
予定表
連絡先
クイック設定を表示
レイアウト
作成と返信
添付ファイル
ルール
一括処理
迷惑メール
アクションのカスタマイズ
メールを同期
メッセージの取り扱い
転送
自動応答
S/MIME
グループ
作成と返信
電子メールの署名
送信するメール メッセージに自動的に追加される署名を作成します。
--
九産　太郎
九州産業大学　〇〇学部　〇〇学科　〇年
Mail: k20AA999@st.kyusan-u.ac.jp
（これは一例）
新規作成するメッセージに自動的に署名を追加する
転送または返信するメッセージに自動的に署名を追加する
メッセージ形式
メッセージを作成するときに差出人と Bcc の行を表示するかどうかを選択します。
BCC を常に表示する
差出人を常に表示する
メッセージを
プレーンテキスト
形式で作成する
署名を入力したら、「保存」して設定を閉じる
保存
破棄

PCの場合

「新しいメッセージ」をクリックすると、

メール作成画面が開く

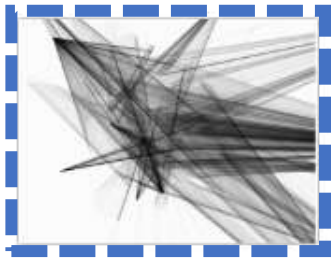


レポートの画面例

宛先　隅田　康明 ×　sumida@mail.kyusan-u.ac.jp

CC

コンピュータ基礎演習(月〇)レポート01



画像を添付する場合
（今回はなくていい、
添付する場合は次のスライドを参考）

20AS999の九産太郎です。

第1回のレポートを提出します。

ここにプログラム全文を貼り付け

--

署名
名前、学部学科学年
E-mail: k21AA999@st.kyusan-u.ac.jp

送信　破棄

参考メールに画像ファイルを添付

- エクスプローラーで画像ファイルの保存場所を開く
- 画像ファイルをドラッグ＆ドロップ(摘んで離す)

第1回レポート（今回はメール）

- 締切：授業日週の金曜日　18：00
  - 1秒でも遅れたら遅れ提出で減点
  - 基本的には授業時間中の提出を目指すこと
  - 未提出の場合、欠席とすることがある
  - **全員、現時点のプログラムで一度提出すること**

- 次回以降は、Moodleでの提出になる
  - レポートの点数が付くのは次回からだが、
    次回のレポートに今回の内容が含まれる

メール送信の例

宛先：sumida@ip.kyusan-u.ac.jp

件名：PC基礎(月1) 01

21AA999の九産太郎です。
第1回のレポートメールを送ります。

～～を描きました。
[プログラム]
（コピーしたプログラムを貼り付け）

--
21AA999 九産太郎
九州産業大学　芸術学部　○○学科 1年

授業時間外の質問方法

- 基本的にはメールでの質問を推奨
  - sumida@ip.kyusan-u.ac.jp
- オフィスアワーの時間に研究室に来る
  - なるべく、メールでアポイントメントを取ってから
- Teamsを使えるなら、Teamsで質問でも可