プログラミング１　第２回教材　　担 当　神保 雅人

2023 年 10 月 19 日（木）実施

キーボードからの入力

これまでのプログラムは固定的なデータを扱うものであったが，実用的なプログラムでは，実行するたびに異なるデータを扱えるものでなければならない。

Java 言語では，標準入力装置のキーボードからデータを入力する方法は複数あるが，ここではパッケージ java.util の中に用意されている Scanner クラスを利用することにする。その使い方の例を次に挙げる。

例）
```
Scanner sc = new Scanner(System.in);   // Scanner クラスのインスタンス sc を作成
int a = sc.nextInt();  // sc に働き掛けを行い，キーボードから整数を読み込む。
```

クラスの定義では枠組みを用意しているが，それを利用するに当たっては new 演算子を用いてインスタンスを作成する。System.in は標準入力装置から入力されるデータの並び（**ストリーム**）を表す。nextInt は整数を読み込む際に用いるメソッドであるが，他にも読み込むデータの型に応じたメソッドが用意されている。

例題１　（標準入力装置からの整数データの入力）

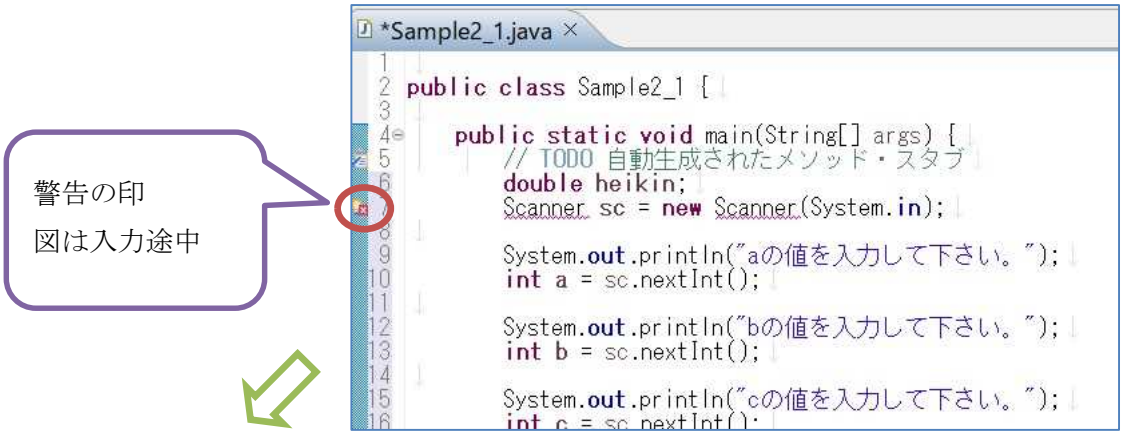
次のプログラムは，標準入力装置から整数データを入力して，標準出力装置に計算結果を表示させるものである。これを作成して，実行せよ。ここで，クラス名は Sample2_1，ソースファイル名は Sample2_1.java とする。

```
import java.util.Scanner;


public class Sample2_1 {

    public static void main(String[] args) {
        // TODO 自動生成されたメソッド・スタブ
        double heikin;
        Scanner sc = new Scanner(System.in);

        System.out.println("a の値を入力して下さい。");
        int a = sc.nextInt();

        System.out.println("b の値を入力して下さい。");
        int b = sc.nextInt();

        System.out.println("c の値を入力して下さい。");
        int c = sc.nextInt();

        heikin=(double)(a+b+c)/3;

        System.out.println("a, b, c の平均値は " + heikin);
    }

}
```

【解説】nextInt メソッドはキーボードから入力があるまで待機していて，コンソールには何も表示しないので，プログラムの利用者に入力を促すガイドコメント（"a の値を入力して下さい。" 等）を表示する必要がある。

（double）の様に括弧内にデータ型を書いた物は**型キャスト**と呼ばれ，その場だけデータを括弧内のデータ型に変換する。ここでは実数割る整数の計算が行われ，結果は実数として求まる。

＊ Eclipse で main メソッドの内容を入力し終わって保存しても，次の図の様な警告が残っていたら，[ソース] → [インポートの編成]を選択して適用する。その結果として，ソースプログラムの先頭に『import java.util.Scanner;』の１行が追加される。これは Scanner クラスを利用する為に必要となる。（設定によっては自動的に付加される場合もある）



プログラムの制御構造

1960 年代後半にダイクストラが提唱した**構造化プログラミング**という考え方では，手続き型のプログラムを記述する際には，**順次**，**選択**，**反復**という標準的な**制御構造**のみを用い，先ずプログラムの概略構造を設計し，その大まかな単位を段階的に詳細化して処理を記述していく。

順次構造

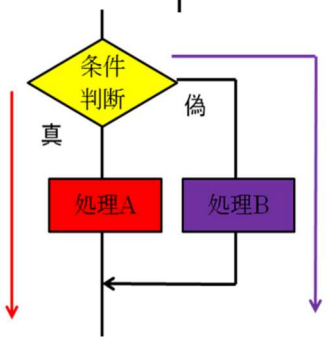
順次構造とは，プログラム中の文を処理していく順に記述したものである。これまで扱ったプログラムは，全て順次構造によって記述されたものであり，最も基本的な制御構造と言える。

プログラムの処理の流れを図示する手法の一つに**流れ図**がある。この流れ図で順次構造を表すと右図の様になる。(色矢印は処理の流れを補足)

選択構造

選択構造とは，条件や式の値によってプログラムの処理の流れを分ける構造である。選択構造の基本は 2 分岐と呼ばれる構造で，この構造を流れ図で表すと右図の様になる。

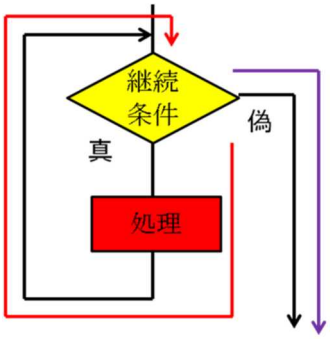
また，式の値によって，幾つもの異なる処理が必要なときには，**多分岐**という選択構造も利用可能である。



反復構造

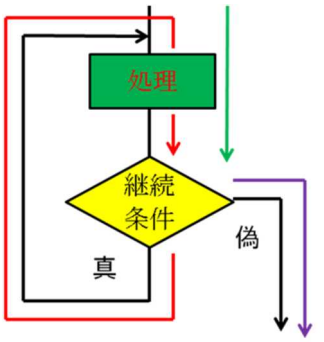
反復構造とは，継続条件が満たされている間，定められた範囲内の文に記述された処理を繰り返して実行する構造である。(Java 言語以外のプログラム言語では，終了条件が満たされない間，文を繰り返して実行する構造を持つものもある)

なお，反復構造には，右の流れ図で表される 2 種類の場合がある。



1) 0 回以上の繰り返し（右上図）

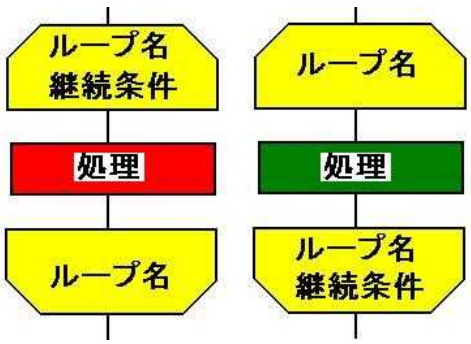
先ず継続条件が判定され，真であれば定められた範囲内の文に記述された処理を実行する。始めから継続条件が満たされない場合には，文は全く実行されないため，0 回以上の繰り返しと呼ばれる。



2) 1 回以上の繰り返し（右下図）

先ず定められた範囲内の文に記述された処理が実行され，その後に継続条件が判定される。始めから継続条件が満たされない場合でも，最初の 1 回は定められた範囲内の文に記述された処理が実行されるため，1 回以上の繰り返しと呼ばれる。

＊ 反復構造に対する流れ図に表れる閉線図形をループと呼ぶことから，反復構造のプログラム構造をループと称することがある。



複雑な処理は，順次構造，選択構造，反復構造の組み合わせで実現される。プログラムの構造は，最も大掴みにした概略構造で見ると順次構造となる。選択構造や反復構造を中間の概略構造と看做した場合，その詳細構造として，それぞれの構造の流れ図で『処理』と書かれた箇所に，選択構造や反復構

造を埋め込んだ構造も可能である。そこで，反復構造の開始位置と終了位置とを『ループ端』によって明示し，構造を見易くした右のような流れ図も利用される。

2 分岐のプログラム

if 文

Java 言語で 2 分岐のプログラムを実現するための文として，if 文が用意されている。if 文の構文は次の様になる。

if（ 条件式 ）真文 [else 偽文]

ここで，[]内は省略可能であり，省略時は条件式が偽のときは何もしない場合を表す。また，真文または偽文を{}で囲んだブロックとして，複数の文で構成することが出来る。

条件式

条件式では，条件が満たされる（真となる）場合には，値が true となり，条件が満たされない（偽となる）場合には，値が false となる。true 及び false は論理型(型名は boolean)のリテラルである。（第 1 回教材の p. 14 参照）

条件式で用いられる**関係演算子**及び**等価演算子**を次に挙げる。

関係演算子	書式	意味
<	x < y	x が y より小さければ true，それ以外は false
>	x > y	x が y より大きければ true，それ以外は false
<=	x <= y	x が y より小さいか，両者が等しければ true，それ以外は false
>=	x >= y	x が y より大きいか，両者が等しければ true，それ以外は false

等価演算子	書式	意味
==	x == y	x と y とが等しければ true，それ以外は false
!=	x != y	x と y とが等しくなければ true，それ以外は false

また，複数の条件式を組み合わせるために用いられる**論理演算子**を次に挙げる。

論理演算子	書式	意味
&&	式 1 && 式 2	式 1 及び式 2 が共に true であれば true，それ以外は false
||	式 1 || 式 2	式 1 または式 2 のどちらか一方が true であれば true（両者が true の場合も含む），それ以外は false
!	!式	式が true であれば false，式が false であれば true

例題 2　（if 文の基本）

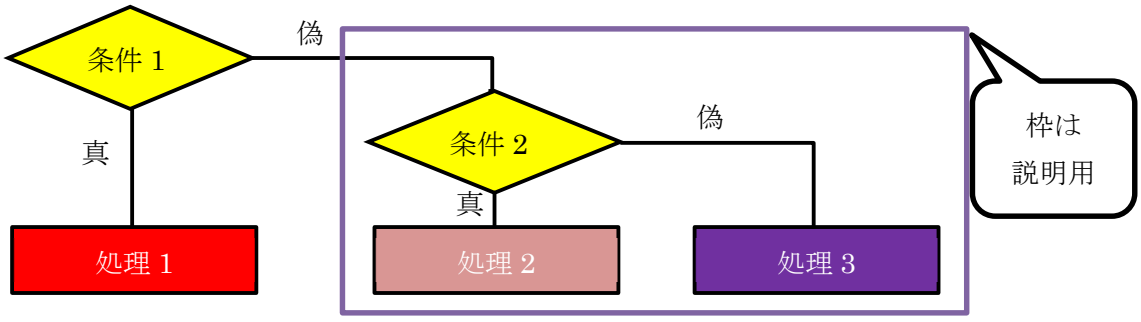
次のプログラムは，標準入力装置から入力された年齢に応じて，2 つの異なる結果を表示させるものである。これを入力して，実行せよ。なお，実行は 2 回行い，それぞれの実行時に 20 未

満，20 以上の整数を入力する。ここで，クラス名は Sample2_2，ソースファイル名は Sample2_2.java とする。

```
import java.util.Scanner;


public class Sample2_2 {

    public static void main(String[] args) {
        // TODO 自動生成されたメソッド・スタブ
        Scanner sc = new Scanner(System.in);

        System.out.println("あなたの年齢を入力して下さい。");
        int age = sc.nextInt();

        if (age < 20)
            System.out.println("あなたにはこの品物をお売りすることが出来ません。");
        else
            System.out.println("それでは代金をお支払下さい。");
    }

}
```

入れ子の if 文

if 文では，真文または偽文として if 文を用いることが可能である。これを"**入れ子の if 文**"と呼ぶ。If 文は 2 分岐であるが，入れ子の if 文を用いることによって，3 以上の分岐(**多分岐**)を表現することが可能となる。次の図は，偽文として if 文を用いた入れ子の if 文の流れ図である。



例題 3　（**入れ子の if 文**）

次のプログラムは，標準入力装置から入力された点数に応じて，異なる成績を表示させるものである。これを入力して，実行せよ。なお，実行は 7 回行い，それぞれの実行時に負の整数，100 超の整数，90 以上 100 以下の整数，80 以上 89 以下の整数，70 以上 79 以下の整数，60 以上 69 以下の整数，0 以上 59 以下の整数を入力する。ここで，クラス名は Sample2_3，ソースファイル名は Sample2_2.java とする。

```
import java.util.Scanner;


public class Sample2_3 {
```

```
    public static void main(String[] args) {
        // TODO 自動生成されたメソッド・スタブ
        Scanner sc = new Scanner(System.in);

        System.out.println("点数(0～100)を入力して下さい。");
        int point = sc.nextInt();

        if (point < 0 || point > 100)
            System.out.println("点数が範囲外です。");
        else if (point >= 90)
            System.out.println("成績は秀です。");
        else if (point >= 80)
            System.out.println("成績は優です。");
        else if (point >= 70)
            System.out.println("成績は良です。");
        else if (point >= 60)
            System.out.println("成績は可です。");
        else
            System.out.println("成績は不可です。");
    }

}
```

演習

次のプログラムは，標準入力装置から入力された文字列の先頭の文字に応じて，異なる結果を表示させるものである。その場合分けは次の３通りに設定されている。

(1) Y または y が入力された場合 ⇒ 『このサービスが利用出来ます。』と表示される。
(2) N または n が入力された場合 ⇒ 『このサービスは学部内限定です。』と表示される。
(3) 上の(1)，(2)に挙げられたもの以外の文字が入力された場合 ⇒ 『Y，N 以外の文字が入力されました。』と表示される。

このプログラムリストの空欄を埋めて完成させたプログラムを入力し，実行せよ。なお，実行は３回行い，それぞれの実行時に上の(1)～(3)を確かめる。ここで，クラス名は Ex2，ソースプログラム名は Ex2.java とする。

```
import java.util.Scanner;


public class Ex2 {

    public static void main(String[] args) {
        // TODO 自動生成されたメソッド・スタブ
        Scanner sc = new Scanner(System.in);
        char ch;

        System.out.println("あなたはサービス創造学部の学生ですか？");
        System.out.println("(Y または N を入力して下さい。)");
        String str = sc.next();
```

```
        ch = str.charAt(0);

        if(ch == 'Y' || ch == 'y')
            System.out.println("このサービスが利用出来ます。");
        else 1)     (ch == 'N' || ch == 'n')
            System.out.println("このサービスは学部内限定です。");
        2)
            System.out.println("Y, N 以外の文字が入力されました。");
    }

}
```

【解説】 charAt(引数)は，文字列の中から，引数で指定された番号(0 は先頭)の文字を読み込むメソッドである。

提出物：

1) 例題 1，例題 2，例題 3 及び演習のプログラムのコンソールへの出力結果をコピーして貼り付けたテキストファイル res2.txt をメールに添付する。
2) 演習のソースプログラムのファイル Ex2.java をメールに添付する。
3) 第 2 回の理解度確認用の質問ファイル Prog1_Questions_2nd.txt に解答を記入して，メールに添付する。

＊ メールの件名は『プログラミング 1　第 2 回課題』（鍵括弧は要らない）とし，メールの本文の書き方は情報入門に準拠する。

余裕のある人向けの課題

例題 3 の入れ子の if 文に対する流れ図を Word で作成せよ。