Java 初級レベル

チェック項目

□(1)char 型、String 型

□(2)多重 for

□(3) 2 次元配列、配列宣言法もう一つ

□(4)複雑な条件判定、多重 if

□(5)switch 文

□(6)while 文と do_while 文

１．文字と文字列

今までに、変数のタイプとして、int と double が出てきました。さらに、文字列(String)と１文字型(char)があります。
文字型は、１文字しか書けない（入れられない）変数です。
文字列は文字の並びで、何文字でもかまいません。
文字列は String、　文字型は char と書きます。
char c1, c2;
String srt1, str2;
などと宣言します。
文字型変数への文字の代入は、
c1= 'Y';
c2= 'N';
という風に ‘　’ で文字を囲みます。
文字列への代入は、
Str1= "２重コーテーションでくくる";
のように、"　"で文字列をくくります。
キーボードから文字列型変数への入力は、
String str;
str= br.readLine();
System.out.println(str);
文字の入力は、
char moji;
moji= (br.readLine()).charAt(0);
System.out.println(moji);
です。

文字と文字列の入力プログラム例を示します。

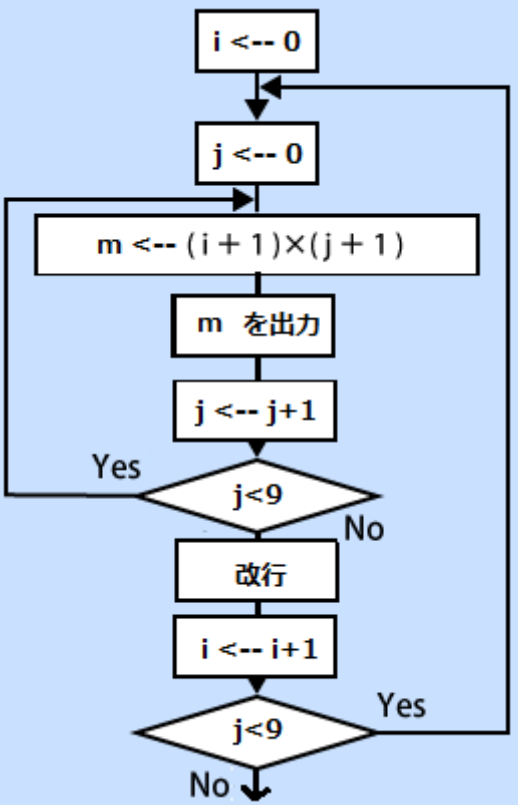
```
import java.io.*;↓
↓
public class  KeyIn4    {↓
    public static void main(String args[]) throws IOException{↓
    //---変数宣言↓
        char c;↓
        String str;↓
        BufferedReader br= new BufferedReader(new InputStreamReader(System.in));
                ↓
    //----プログラム本体↓
        ↓
        System.out.println("文字列を入力してください");↓
        str= br.readLine();↓
        System.out.println(str);↓
↓
        System.out.println("１文字入力してください");↓
        c= (br.readLine()).charAt(0);↓
        System.out.println(c);↓
        ↓
    } ↓
}[EOF]
```

２．多重 for ループ

多重ループは多次元配列とペアで効果を発揮します。それ以外で多重ループの効果を見せろ、と言われると次の２つしか思い浮かびません。

（例１）　九九表を出力する。

フローチャートは次の通りです。



プログラムは、以下の通りです。

```
public class Ikk2 {↓
    public static void main(String[] args) {↓
        int m;↓
        ↓
        for(int i=0; i<9; i++){↓
            for(int j=0; j<9; j++){↓
                m=(i+1)*(j+1);↓
                System.out.print(m + ",");↓
            }↓
            System.out.println();↓
        }↓
    }↓
}[EOF]
```

（例２）＊マークを、次のように出力する。

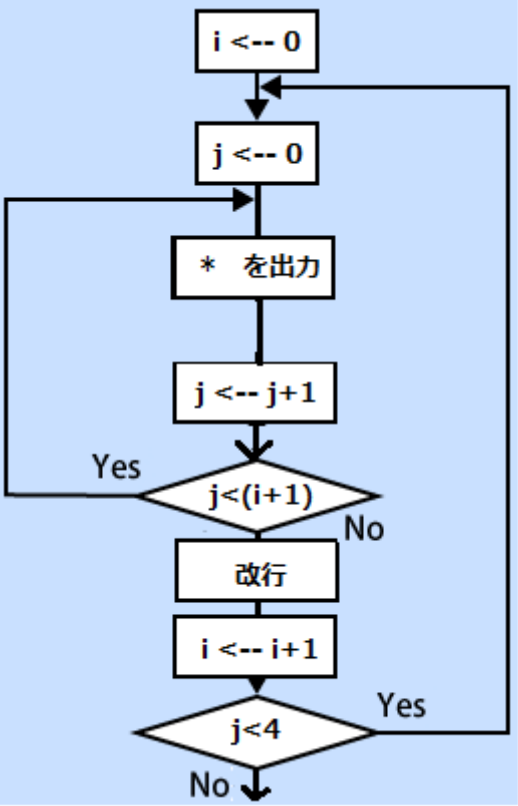
＊

＊＊

＊＊＊

＊＊＊＊

フローチャートです。



プログラムです。

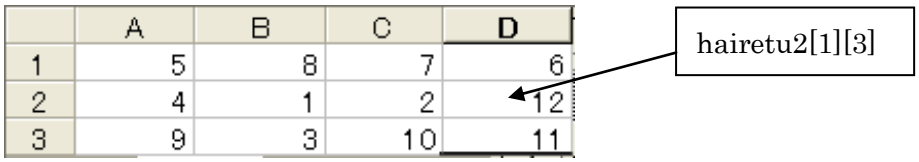
```
public class Ikk1 {↓
    public static void main(String[] args) {↓
↓
        for(int i=0; i<4; i++){↓
            for(int j=0; j<(i+1); j++){↓
                System.out.print("*");↓
            }↓
            System.out.println();↓
        }↓
    }↓
}[EOF]
```

３．多次元配列

３．１　２次元配列

下図はＥＸＣＥＬの表の一部です、２次元配列は、この表のイメージです。これは３×４の２次元配列です。EXCEL ではセル（マス）を（３．Ａ）なんて指定します（３．Ａの数値は９）。しかし Java では下図の縦 1,2,3,を０，１，２と指定し、横 A,B,C,D も０，１，２，３と指定します。ですから、もし、下の２次元配列名を hairetu2 とすると、hairetu2[2][0] の値が９になります。要素位置指定は“縦”“横”の順ですので注意してください。この２次元配列を宣言し下のように初期値をセットするには次のようにします。

int　hairetu2[ ][ ]={{5,8,7,6},{4,1,2,12},{9,3,10,11}};



３．２　配列の宣言

配列の宣言の仕方は沢山あります。入門レベルでは、簡単のために最も簡単な方法を使いました。初級レベルではもう一つ、普段よく用いられる宣言法を示します。

(1)入門レベルでの配列宣言

１次元配列の宣言は、例えば、

int　a[] = { 1,2,3,4,5 };

です。

２次元配列の宣言は、例えば、

int　a[] = {{ 1,2,3,4,5 },{3,3,4,5,1}};

のように書きます。

(2)新しい配列宣言法

配列の宣言は２行必要になります。例えば、要素数５で配列名 a の１次元配列を宣言するには次のようになります。

int[]　a;

a= new int[5];

です。

２次元配列の宣言は、例えば、

int[][]　a;
a= new int[2][5];
のように書きます。
この段階では要素にデータは入っていません。

３．３　２重 for での２次元配列操作

以下では簡単のために、h2 という名の２×３の配列を用います。

配列 h2

0	0	0
0	0	0

（例１）配列要素に数値を代入する
２×３の、２次元配列 h2 の要素に数値（０）を代入し、上の通り２段に表示します。

(a) 繰り返しのないプログラム：

```
class Ikk3_1 {
    public static void main(String args[]){
        int[][] h2;
        h2= new int[2][3];

        h2[0][0]= 0;
        h2[0][1]= 0;
        h2[0][2]= 0;
        h2[1][0]= 0;
        h2[1][1]= 0;
        h2[1][2]= 0;
        System.out.print(h2[0][0] + "," );
        System.out.print(h2[0][1] + "," );
        System.out.print(h2[0][2] + "," );
        System.out.println();
        System.out.print(h2[1][0] + "," );
        System.out.print(h2[1][1] + "," );
        System.out.print(h2[1][2] + "," );
        System.out.println();
    }
}
```

(b) 列を繰り返さないプログラム：

```
class Ikk3_2 {
    public static void main(String args[]){
        int[][] h2;
        h2= new int[2][3];

        for(int i=0; i<2; i++){
            h2[i][0]= 0;
            h2[i][1]= 0;
            h2[i][2]= 0;
        }
        for(int i=0; i<2; i++){
            System.out.print(h2[i][0] + "," );
            System.out.print(h2[i][1] + "," );
            System.out.print(h2[i][2] + "," );
            System.out.println();
        }
    }
}
```

(c) 行、列とも繰り返しのプログラム：

```
class Ikk3_3 {
    public static void main(String args[]){
        int[][] h2;
        h2= new int[2][3];

        for(int i=0; i<2; i++){
            for(int j=0; j<3; j++){
                h2[i][j]= 0;
            }
        }
        for(int i=0; i<2; i++){
            for(int j=0; j<3; j++){
                System.out.print(h2[i][j] + "," );
            }
            System.out.println();
        }
    }
}
```

（例２）h2 の要素の総和を求める

```
class Ikk4 {
    public static void main(String args[]){
        int h2[][] = {{ 1,2,3 },{ 4,5,6 }};
        int s;

        s=0;
        for(int i=0; i<2; i++){
            for(int j=0; j<3; j++){
                s= s + h2[i][j];
            }
        }
        System.out.print(s);
    }
}
```

（例３）h2 の最大値を求める

```
class Ikk5 {
    public static void main(String args[]){
        int h2[][] = {{ 1,2,3 },{ 4,5,6 }};
        int s;

        s=h2[0][0];
        for(int i=0; i<2; i++){
            for(int j=0; j<3; j++){
                if( h2[i][j]> s){
                    s= h2[i][j];
                }
            }
        }
        System.out.print(s);
    }
}
```

（例４）何をやっているでしょう？

```
class Ikk6 {
    public static void main(String args[]){
        int h2[][] = {{ 1,2,3 },{ 4,5,6 }};
        int buf;

        for(int i=0; i<3; i++){
            buf=h2[0][i];
            h2[0][i]= h2[1][i];
            h2[1][i]= buf;
        }
        for(int i=0; i<2; i++){
            for(int j=0;j<3; j++){
                System.out.print(h2[i][j]+",");
            }
            System.out.println();
        }
    }
}
```

（例５）何をやっているでしょう？

```
for(i=0; i<2; i++){
  for(j=0; j<2; j++){
    h2[i][j]= h2[i][j+1];
  }
}
```

（例６）何をやっているでしょう？

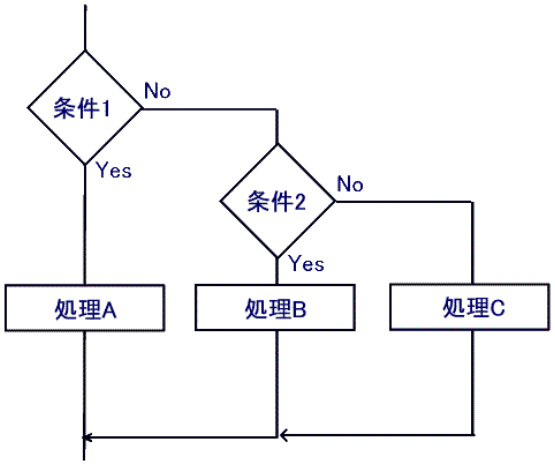
```
for(i=0; i<2; i++){
  for(j=1; j<3; j++){
    h2[i][3-j]= h2[i][2-j];
  }
}
```

４．複雑な条件分岐

４．１　if～else～if～

　if～else…の…部には、また if 文を書けます。

フローチャートは次の通りです。



プログラムでは、

```
if　（式）　{
　　文（複数可）
}　else if　（式）　{
　　文（複数可）
}　else if　（式）　{
　　文（複数可）
}　else　{
　　文（複数可）
}
```

になります。

（例）

```
class Ikk13_2{
    public static void main(String[] args){
        int score=75;
        
        if( score>= 80){
            System.out.print("A");
        } else  if( score>= 70) {
            System.out.print("B");
        } else  if( score>= 60) {
            System.out.print("C");
        } else {
            System.out.print("D");
        }
    }
}
```

４．２　switch 文

前節の if~else~if….は switch 文で簡単に書けます。

switch 文の書き方は、

```
switch　（式）　{
    case　値：　文（複数可）；　break；
    case　値：　文（複数可）；　break；
    case　値：　文（複数可）；　break；
    default：　文（複数可）；　break；
}
```

です。

（注：default は「その他」ですが、default は、無くてもよい）

これを if 文で書くと

```
if(式 == 値){
        文（複数可）
}else if((式 == 値){
        文（複数可）
}else{
        文（複数可）
}
```

switch 文の例も、思いつくものは次の２つしかありません。

（例１）数値で場合分け

```
import java.io.*; ↓
↓
public class  Ikk10 {↓
↓
    public static void main(String args[]) throws IOException{  ↓
        int i;↓
        BufferedReader br= new BufferedReader(new InputStreamReader(System.in));
        ↓
        i= Integer.parseInt(br.readLine());↓
        ↓
        switch(i){//iの値によって分岐します↓
            case 1:↓
                System.out.println("iは1です。");        //iが1の場合↓
                break;↓
            case 2:↓
                System.out.println("iは2です。");        //iが2の場合↓
                break;↓
            case 3:↓
                System.out.println("iは3です。");        //iが3の場合↓
                break;↓
            default:↓
                System.out.println("iはそれ以外です。");//それ以外↓
        }↓
    } ↓
} [EOF]
```

（例２）文字で場合分け

```
import java.io.*; ↓
↓
public class  Ikk11 {↓
↓
    public static void main(String args[]) throws IOException{  ↓
        char c;↓
        BufferedReader br= new BufferedReader(new InputStreamReader(System.in));
        ↓
        c= (br.readLine()).charAt(0);↓
        switch(c)   {↓
            case 'a': // c が'a'のとき、↓
                System.out.println("a が入力されました。");↓
                break;↓
            case 'b': // c が'b'のとき、↓
                System.out.println("b が入力されました。");↓
                break;↓
            default: // c が'a'でも'b'でもないとき、↓
                System.out.println("a かb を入力してください。");↓
                break;↓
        }↓
    } ↓
} [EOF]
```

４．３　複雑な条件文

if 文の条件部に複雑な論理式を書くことができます。&&は論理積、||は論理和です。

（例１）

```
class Ikk12{
    public static void main(String[] args){
        int a= 5;
        int b= 6;
        int c= 3;
        if( (a>b) && (a>c)){
            System.out.print("aが最大");
        }
        if( (b>a) && (b>c)){
            System.out.print("bが最大");
        }
        if( (c>a) && (c>b)){
            System.out.print("cが最大");
        }
    }
}
```

（例２）

```
class Ikk13{
    public static void main(String[] args){
        int score=75;
        int a= 80;
        int b= 70;
        int c= 60;

        if( score>= a){
            System.out.print("A");
        }
        if( (a>score) && (score>= b)){
            System.out.print("B");
        }
        if( (b>score) && (score>=c)){
            System.out.print("C");
        }
        if(score<c){
            System.out.print("D");
        }
    }
}
```

５．　while 文、do_while 文

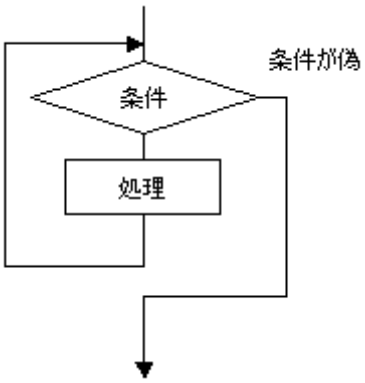
繰り返し命令で、次の形をしていて、条件が満たされている間は、「処理」を繰り返します。

５．１　while 文

次のように書きます。

while（条件）｛
　　繰り返される処理；
｝

フローチャートは以下です。



（例１）何をしていますか？

```
class Ikk14{
    public static void main(String[] args){
        int i=0;

        while (i < 3){
            System.out.println(i);
            i++;   //iを+1します
        }
    }
}
```

（例２）何をしていますか？

```
import java.io.*;
↓
public class  Ikk16 {↓
↓
    public static void main(String args[]) throws IOException{
        int data;↓
        int sum=0;↓
        BufferedReader br= new BufferedReader(new InputStreamReader(System.in));
        ↓
        while( (data=Integer.parseInt(br.readLine())) > 0){↓
            sum= sum + data;↓
        }↓
        System.out.println(sum);↓
    }↓
} [EOF]
```

５．２　do_while 文

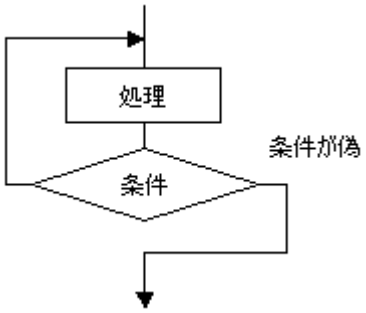
条件が満たされている間、「繰り返される処理」を繰り返します。

表記法と、フローチャートは以下です。

do {

　　繰り返される処理

}while（ 条件）；



（例１）何をしていますか？

```
class Ikk15{↓
    public static void main(String[] args){↓
        int i = 6; ↓
        ↓
        do {↓
            System.out.println("iは" + i + "です。");↓
            i++; ↓
        } while (i <= 5);↓
    }↓
}↓
[EOF]
```

（例２）　当たるまで続く数当て

```
import java.io.*;
public class  Ikk17 {

    public static void main(String args[]) throws IOException{
        int ans=8;
        int data=0;
        BufferedReader br= new BufferedReader(new InputStreamReader(System.in));

        do{
            System.out.println("１０以下の数字を入れてください");
            data=Integer.parseInt(br.readLine());
            if( ans==data ){
                System.out.println("当たり");
            }
        }while(ans != data);
    }
} [EOF]
```

（例３）　少し乱暴な平方根の求め方

```
import java.io.*;
public class  Root {

    public static void main(String args[]) throws IOException{

        double data;
        double heihoukon= 0.0;
        double delta= 0.01;
        BufferedReader br= new BufferedReader(new InputStreamReader(System.in));

        System.out.println("数を入力して下さい");
        data= Double.parseDouble(br.readLine());
        do{
            heihoukon= heihoukon + delta;
        }while(data > (heihoukon* heihoukon));
        System.out.println(heihoukon);
    }

}  [EOF]
```