ループ処理の続き

前回までに「ループ処理」を習った。ループ処理はある条件を定め、その条件が成立している間は繰り返し処理を行う方法である。日常的な出来事に照らし合わせれば、例えば、

「お金がある限り買い物を続ける」
「品数が 10 品になるまで買い物を続ける」

といった事柄が繰り返し処理に当たる。Java でこれらの繰り返し処理を実現する文として、**while 文**、**for 文**、**do-while 文**といった文が用意されている。

本日はループ処理に関連した処理として、**無限ループ**、**break 文**、**continue 文**、**ラベル付き break 文**、**ラベル付き continue 文**について学習していく。

無限ループと break 文

while 文、for 文、do-while 文を用いて無限ループを作成することができる。while 文、do while 文ではループを抜けるための条件式を常に“真”にすることで無限ループにすることができる。この条件式を常に真にするには boolean 型の値である“true”を用いて以下の使用例のように記述すればよい。また、for 文ではループを抜けるための条件式を常に“真”にするか、あるいは省略することで無限ループとすることができる。ただし、このままではプログラムが終了しなくなってしまうので、実際使用する場合には、無限ループを抜けるためにループの内部で条件分岐させ、break 文を用いて利用する。この break 文は switch 文、または while などの繰り返し文のブロック内で使用し、実行されるとそれ以降のブロック内の処理を行わないでブロックの外の次の処理に移らせる。

使用例

```
while(true)
{
        System.out.print("数値を入力してください:");
        int num = stdIn.nextInt();
        if(num == 0) break;
        System.out.println("入力した値は" + num + "です");
}
```

使用例

```
for(int i=1; ; i++)
{
        System.out.print("数値を入力してください:");
        int num = stdIn.nextInt();
        if(num == 0) break;
        System.out.println(i + "番目に入力した値は" + num + "です");
}
```

continue 文

プログラムの作成上、ループ処理の途中で処理する文をジャンプして次の繰り返しを開始したい場合がある。その場合には **continue 文**を用いる。この continue 文を for 文で用いる場合には再初期化の式（以下の書式での式３にあたる）の実行に移る。この continue 文はループに対して使われるだけで switch 文では使えない。continue 文は if 文などを用いて次のように書く。

この場合では for 文の途中の if 文で式 4 を判定する。式 4 が“真”ならば、文 2 を実行せずに式 3 での再初期化を行い、式 2 の条件式を判定する。

書式

```
for(式１;  式２;  式３){
        文１;
        if(式４) continue;
        文２;
}
```

使用例

```
        for(int i=1; i<=5; i++)
        {
                if(i==3) continue;
                System.out.print("数値を入力してください:");
                int num = stdIn.nextInt();
                System.out.println(i + "番目に入力した値は" + num + "です");
        }
```

ラベル付き break 文とラベル付き continue 文

C 言語には指定した位置に処理の流れをジャンプさせる **goto 文**と分岐（ジャンプ）先を示すための**ラベル文**というものがある。

書式（C 言語）

```
    for(…){
        if(式 1)
            goto ラベル文;
    }
    ラベル文: 文 1;
```

文 1 は省略することが可能で、その場合には「**ラベル文**:;」のように書くこともできる。

この goto 文はプログラムの作成上、不可欠のものではなく、たいがいのプログラムは goto 文を用いなくても書くことができ、この処理は多用することで処理の流れを把握することを難しくしてしまう。そのため、goto 文を多用したプログラムはスパゲッティプログラム（複雑に入り交じっている意味）と呼ばれて敬遠されてきた。

例外として、goto 文が有用となる場合もある。それは以下のような多重ループを抜ける時などである。

この場合では for 文の途中の if 文で式 1 を判定する。式 1 が“真”であるならば、ラベル文までジャンプし、文 1 が実行される。

使用例（C 言語）

```
        for( i = 0;  i < 5;  i++ ){
            for( j = 0;  j < 3;  j++ ){
                if( i = = 2 ) goto end;
                printf("%d ",  i + j );
            }
        }
        end : printf( “i が 2 の時にジャンプしました” );
```

Java ではこういった理由から goto 文を除外し（予約語には残っているが）、代わりの構文としてラベル付き break 文とラベル付き continue 文というものを用意している。

以下にラベル付き break 文の使用例を示す。

使用例

```
import java.util.Scanner;

public class Prog05_10
{
        public static void main(String[ ] args)
        {
                Scanner stdIn = new Scanner(System.in);

        Outer:
                for(int i=0; i<5; i++)
```

```
                {
                Inner:
                        for(int j=0; j<3; j++)
                        {
                                System.out.print("整数を入力して下さい:");
                                int t = stdIn.nextInt();

                                if(t == 99999)
                                {
                                        System.out.println("外側のループを抜けます");
                                        break Outer;
                                }
                                else if(t == 88888)
                                {
                                        System.out.println("内側のループを抜けます");
                                        break Inner;
                                }
                                else
                                {
                                        System.out.println("(i,j) = (" + i + ", " + j +
                                                ") で入力された値は" + t + "です");
                                }
                        }
                }
        }
}
```

実行例

```
整数を入力して下さい:1
(i,j) = (0, 0) で入力された値は 1 です
整数を入力して下さい:2
(i,j) = (0, 1) で入力された値は 2 です
整数を入力して下さい:3
(i,j) = (0, 2) で入力された値は 3 です
整数を入力して下さい:4
(i,j) = (1, 0) で入力された値は 4 です
整数を入力して下さい:5
(i,j) = (1, 1) で入力された値は 5 です
整数を入力して下さい:88888
内側のループを抜けます
整数を入力して下さい:6
(i,j) = (2, 0) で入力された値は 6 です
整数を入力して下さい:7
(i,j) = (2, 1) で入力された値は 7 です
整数を入力して下さい:99999
外側のループを抜けます
```

以下にラベル付き continue 文の使用例を示す。

使用例

```
import java.util.Scanner;

public class Prog05_11
{
        public static void main(String[ ] args)
```

```
        {
                Scanner stdIn = new Scanner(System.in);

        Outer:
                for(int i=0; i<5; i++)
                {
                Inner:
                        for(int j=0; j<3; j++)
                        {
                                System.out.print("整数を入力して下さい:");
                                int t = stdIn.nextInt();

                                if(t == 99999)
                                {
                                        System.out.println("外側のループを進めます");
                                        continue Outer;
                                }
                                else if(t == 88888)
                                {
                                        System.out.println("内側のループを進めます");
                                        continue Inner;
                                }
                                else
                                {
                                        System.out.println("(i,j) = (" + i + ", " + j +
                                                ") で入力された値は" + t + "です");
                                }
                        }
                }
        }
}
```

実行例

```
整数を入力して下さい:1
(i,j) = (0, 0) で入力された値は 1 です
整数を入力して下さい:2
(i,j) = (0, 1) で入力された値は 2 です
整数を入力して下さい:3
(i,j) = (0, 2) で入力された値は 3 です
整数を入力して下さい:4
(i,j) = (1, 0) で入力された値は 4 です
整数を入力して下さい:5
(i,j) = (1, 1) で入力された値は 5 です
整数を入力して下さい:88888
内側のループを進めます
整数を入力して下さい:6
(i,j) = (2, 0) で入力された値は 6 です
整数を入力して下さい:7
(i,j) = (2, 1) で入力された値は 7 です
整数を入力して下さい:99999
外側のループを進めます
整数を入力して下さい:8
(i,j) = (3, 0) で入力された値は 8 です
整数を入力して下さい:
                :
```

：

演習

テキストで示した使用例５つ（C 言語での使用例を除く）全て動作できるように作成し、実行結果を確認しなさい。

課題３の続き

次のプログラムを作成しなさい。

Kadai3_5

10 人の身長データを順に cm 単位(実数)で入力したとき、一番大きな身長は何 cm かを表示するプログラムを作成しなさい。

出力例

1 人目の身長（cm）を入力してください:*(入力待ち)*
2 人目の身長（cm）を入力してください:
(略)
10 人目の身長（cm）を入力してください:*(入力待ち)*
この中で一番身長が高い人は○○cm です

Kadai3_6

買い物の金額を計算し、整理するプログラムを次の仕様で作成しなさい。

・「税抜き単価」と「個数」を入力するたびに次の出力が得られる。
　１．合計金額と税込み合計金額
　２．一番単価が安かったものの金額（税抜き）
　３．一番単価が高かったものの金額（税抜き）
・すべての商品に 10%の消費税がかかる。
・単価に 0 が入力されたら処理を終了する。

出力例

買い物したものの単価は：*(入力待ち)*
何個買いましたか？：*(入力待ち)*
これまでの合計金額は、○○（税込み：△△）円です。
一番安いものの値段は、税抜き××円です。
一番高いものの値段は、税抜き□□円です。
：*(処理の繰り返し)*
買い物したものの単価は：*(0 を入力して終了)*

kadai3_7 2～1000 の範囲の全ての素数を求める。素数を求めるアルゴリズムについては検索すると簡単に見つかる。プログラミングの練習のため、以下のように作成する。

・整数型変数 i、j を用意する。
・i を 2～1000 まで変化させ、素数かどうかを調べる。
・j を 2～i-1 まで変化させ、i を j で割り、その余りが 0 ならば、i は素数ではない。
・ループは 2 重になる。
・外側は、i のループで 2～1000 まで範囲である。
・内側は、j のループで 2～i-1 までの範囲である。ただし、i を j で割った余りが 0 になればループから抜ける(break 文)。
・i が素数かどうかにはマークを付けることで対応する。例えば、Boolean 型変数の prime を用意し、これが true ならば素数、false ならば素数でないとする。
・外側のループのブロックの始まり部分では、prime=true とする。
・内側のループのブロックでは i を j で割ったとき余りが 0 ならば、prime=false として、break 文をもちいて、ループから脱出する。

・外側のループのブロックの終り部分では、prime=true ならば、i をディスプレイに書き出す命令を書く。
・このような使い方をする変数 prime のことをフラグと言う。ラベル付き continue 文を用いるとフラグを用いないプログラムにすることができる。フラグを用いないプログラムを作成し、こちらのプログラムを提出すること。

2～1000 の範囲の全ての素数（確認用）
2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199 211 223 227 229 233 239 241 251 257 263 269 271 277 281 283 293 307 311 313 317 331 337 347 349 353 359 367 373 379 383 389 397 401 409 419 421 431 433 439 443 449 457 461 463 467 479 487 491 499 503 509 521 523 541 547 557 563 569 571 577 587 593 599 601 607 613 617 619 631 641 643 647 653 659 661 673 677 683 691 701 709 719 727 733 739 743 751 757 761 769 773 787 797 809 811 821 823 827 829 839 853 857 859 863 877 881 883 887 907 911 919 929 937 941 947 953 967 971 977 983 991 997