1. What (if anything) is wrong with each of the following statements?
   1. if (a > b) then c = 0;

**then is not a part of java reserved words**

* 1. if a > b { c = 0; }

**Parenthesis are missing on if sentence**

* 1. if (a > b) c = 0;

**ok**

* 1. if (a > b) c = 0 else b = 0;

**‘;’ is missing after c=0**

1. Suppose that i and j are both of type int. What is the value of j after each of the following statements is executed?
   1. for (i = 0, j = 0; i < 10; i++) j += i;

**45**

* 1. for (i = 0, j = 1; i < 10; i++) j += j;

**1024**

* 1. for (j = 0; j < 10; j++) j += j;

**15**

* 1. for (i = 0, j = 0; i < 10; i++) j += j++;

**0**

1. Write a program [FivePerLine.java](http://introcs.cs.princeton.edu/java/13flow/FivePerLine.java.html) that, using one for loop and one if statement, prints the integers from 1000 to 2000 with five integers per line. Hint: use the % operator.
2. **public** **class** FivePerLine {
3. **public** **static** **void** main(String[] args) {
4. // print integers from 1000 to 2000, 5 per line
5. **int** start = 1000, end = 2000;
6. **for** (**int** i = start; i <= end; i++) {
7. System.***out***.print(i + " ");
8. **if** (i % 5 == 4) System.***out***.println();
9. }
10. System.***out***.println();
11. }
12. }
13. What is the value of m and n after executing the [following code](http://introcs.cs.princeton.edu/java/13flow/DigitReverser.java.html)?

|  |
| --- |
| int n = 123456789;  int m = 0;  while (n != 0) {  m = (10 \* m) + (n % 10);  n = n / 10;  } |

**m=987654321**

**n=0**

1. **Calendar.** Write a program Calendar that takes two command line arguments m and y and prints out the monthly calendar for the mth month of year y. For example, your output for Calendar 2 2009 should be

|  |
| --- |
| February 2009  S M Tu W Th F S  1 2 3 4 5 6 7  8 9 10 11 12 13 14  15 16 17 18 19 20 21  22 23 24 25 26 27 28 |

*Hint:* see programs [LeapYear.java](http://introcs.cs.princeton.edu/java/12types/LeapYear.java.html) and [DayOfWeek.java](http://introcs.cs.princeton.edu/java/12types/DayOfWeek.java.html).

**public** **class** PrintMonth {

**public** **static** **void** main(String[] args) {

Scanner in = **new** Scanner(System.***in***);

System.***out***.print("Month and Year: ");

String monthText = in.next();

String yearText = in.next();

in.close();

**try** {

**int** month = Integer.*parseInt*(monthText);

**int** year = Integer.*parseInt*(yearText);

// check if it's a valid month and year

**if** (month < 1 || month > 12 || year < 0){

**throw** **new** Exception("Invalid index for month: " + month);

}

*printCalendarMonth*(month, year);

} **catch** (NumberFormatException e) {

System.***err***.println("Numberat Error: " + e.getMessage());

} **catch** (Exception e) {

System.***err***.println(e.getMessage());

}

}

/\*

\* Prints calendar month

\*/

**private** **static** **void** printCalendarMonth(**int** month, **int** year) {

Calendar cal = **new** GregorianCalendar();

cal.clear();

cal.set(year, month - 1, 1);

// Calendar Header

System.***out***.println("\n "+ cal.getDisplayName(Calendar.***MONTH***, Calendar.***LONG***, Locale.***US***) + " " + cal.get(Calendar.***YEAR***));

System.***out***.println("S M Tu W T F S");

**int** weekdayIndex = 0;

// Get weekday of the first day of month.

**int** firstWeekdayOfMonth = cal.get(Calendar.***DAY\_OF\_WEEK***);

// Get all days in month.

**int** numberOfMonthDays = cal.getActualMaximum(Calendar.***DAY\_OF\_MONTH***);

// leave/skip Weekdays

**for** (**int** day = 1; day < firstWeekdayOfMonth; day++) {

System.***out***.print(" ");

weekdayIndex++;

}

// Days of month in tabular format.

**int** day = 0;

String printDay = "";

**for** (day = 1; day <= numberOfMonthDays; day++) {

// Print Day

printDay = (day<10)? " "+day : ""+day;

System.***out***.printf(printDay);

// Next Weekday

weekdayIndex++;

// if it is the last weekday

**if** (weekdayIndex == 7) {

// reset it

weekdayIndex = 0;

// and go to next line

System.***out***.println();

} **else** {

// print space

System.***out***.print(" ");

}

}

// print a final new-line.

System.***out***.println();

}

**Web Exercises**

1. What is wrong with the following code fragment?

|  |
| --- |
| double x = -32.2;  boolean isPositive = (x > 0);  if (isPositive = true) System.out.println(x + " is positive");  else System.out.println(x + " is not positive"); |

***Answer*: It uses the assignment operator = instead of the equality operator ==. A better solution is to write if (isPositive).**

1. What does the following program do?

|  |
| --- |
| public static void main(String[] args) {  int N = Integer.parseInt(args[0]);  int x = 1;  while (N >= 1) {  System.out.println(x);  x = 2 \* x;  N = N / 2;  }  } |

***Answer*: prints out all of the powers-of-two less than or equal to N.**

1. Write a program [Triangle.java](http://introcs.cs.princeton.edu/java/13flow/Triangle.java.html) that takes a command-line argument N and prints an N-by-N triangular pattern like the one below.

|  |
| --- |
| \* \* \* \* \* \*  . \* \* \* \* \*  . . \* \* \* \*  . . . \* \* \*  . . . . \* \*  . . . . . \*  **public** **class** Triangle {  **public** **static** **void** main(String[] args) {  Scanner in = **new** Scanner(System.***in***);  System.***out***.print("Number of dots of Triangle: ");  String dots = in.next();  in.close();  **int** triangleDots = Integer.*parseInt*(dots);    **for**(**int** i=0; i<triangleDots; i++){  **for**(**int** j=0; j<i; j++){  System.***out***.print(" . "); // Print dots  }  **for**(**int** k=0; k<triangleDots-i; k++){  System.***out***.print(" \* "); // Print Triangle  }  System.***out***.println();  }    }  } |

1. Write a program [Ex.java](http://introcs.cs.princeton.edu/java/13flow/Ex.java.html) that takes a command-line argument N and prints a (2N + 1)-by-(2N + 1) ex like the one below. Use two for loops and one if-else statement.

|  |
| --- |
| \* . . . . . \*  . \* . . . \* .  . . \* . \* . .  . . . \* . . .  . . \* . \* . .  . \* . . . \* .  \* . . . . . \* |

**public** **class** Ex {

**public** **static** **void** main(String[] args) {

**int** N = Integer.*parseInt*(args[0]);

**for** (**int** i = -N; i <= N; i++) {

**for** (**int** j = -N; j <= N; j++) {

**if** ((i == -j) || (i == j)){

System.***out***.print("\* ");

} **else** {

System.***out***.print(". ");

}

}

System.***out***.println();

}

}

}

1. Write a program [Diamond.java](http://introcs.cs.princeton.edu/java/13flow/Diamond.java.html) that takes a command-line argument N and prints a (2N + 1)-by-(2N + 1) diamond like the one below.

|  |
| --- |
| % java Diamond 4  . . . . \* . . . .  . . . \* \* \* . . .  . . \* \* \* \* \* . .  . \* \* \* \* \* \* \* .  \* \* \* \* \* \* \* \* \*  . \* \* \* \* \* \* \* .  . . \* \* \* \* \* . .  . . . \* \* \* . . .  . . . . \* . . . . |

**public** **class** Diamond {

**public** **static** **void** main(String[] args) {

**int** N = Integer.*parseInt*(args[0]);

**for** (**int** i = -N; i <= N; i++) {

**for** (**int** j = -N; j <= N; j++) {

**if** (Math.*abs*(i) + Math.*abs*(j) <= N){

System.***out***.print("\* ");

} **else** {

System.***out***.print(". ");

}

}

System.***out***.println();

}

}

}

1. **Seasons.** Write a program Season.java that takes two command line integers M and D and prints the season corresponding to month M (1 = January, 12 = December) and day D in the northern hemisphere. Use the following table

|  |  |  |
| --- | --- | --- |
| **SEASON** | **FROM** | **TO** |
| Spring | March 21 | June 20 |
| Summer | June 21 | September 22 |
| Fall | September 23 | December 21 |
| Winter | December 21 | March 20 |

**public** **class** Season {

**public** **static** **void** main(String[] args) {

Scanner in = **new** Scanner(System.***in***);

System.***out***.print("Month and Day: ");

String monthText = in.next();

String dayText = in.next();

in.close();

**try** {

**int** month = Integer.*parseInt*(monthText);

**int** day = Integer.*parseInt*(dayText);

// check if it's a valid month and year

**if** (month < 1 || month > 12 || day < 1 || day > 31){

**throw** **new** Exception("Invalid index for month or day");

}

*printSeason*(month, day);

} **catch** (NumberFormatException e) {

System.***err***.println("Numberat Error: " + e.getMessage());

} **catch** (Exception e) {

System.***err***.println(e.getMessage());

}

}

/\*

\* Prints Season

\*/

**private** **static** **void** printSeason(**int** month, **int** day) {

**if**((month==3 && day>=21) || (month>3 && month<6) || (month==6 && day<=20)){

System.***out***.print("Spring...");

} **else** **if** ((month==6 && day>=21) || (month>6 && month<9) || (month==9 && day<=21)){

System.***out***.print("Summer...");

} **else** **if**((month==9 && day>=22) || (month>9 && month<12) || (month==12 && day<=21)){

System.***out***.print("Fall...");

} **else** **if**((month==12 && day>=22) || (month>0 && month<3) || (month==3 && day<=20)){

System.***out***.print("Winter...");

} **else** {

System.***out***.print("Does not exist a valid season for the input data");

}

}

}