**Weekend Assessment (Batch 7)**

1. (Science: day of the week) Zeller’s congruence is an algorithm developed by Christian Zeller to calculate the day of the week. The formula is

**Where :**

* h is the day of the week (0: Saturday, 1: Sunday, 2: Monday, 3: Tuesday, 4: Wednesday, 5: Thursday, 6: Friday).
* q is the day of the month.
* m is the month (3: March, 4: April, . . ., 12: December). January and February are counted as months 13 and 14 of the previous year.
* j is the century (i.e., ).
* k is the year of the century (i.e., year % 100).

**Note:** The division in the formula performs an integer division. Write a program that prompts the user to enter a year, month, and day of the month, and displays the name of the day of the week. Here are some sample runs: Enter year: (e.g., 2012):

Enter month: 1-12

Enter the day of the month: 1-31:

Day of the week is Sunday 25 1 2015

Enter year: (e.g., 2012):

Enter month: 1-12:

Enter the day of the month: 1-31:

Day of the week is Saturday 12 5 2012 (Hint: January and February are counted as 13 and 14 in the formula, so you need to convert the user input 1 to 13 and 2 to 14 for the month and change the year to the previous year.)

1. (Game: lottery) **Lottery**, to generate a lottery of a three digit number. The program prompts the user to enter a three-digit number and determines whether the user wins according to the following rules: 1. If the user input matches the lottery number in the exact order, the award is $10,000. 2. If all the digits in the user input match all the digits in the lottery number, the award is $3,000. 3. If one digit in the user input matches a digit in the lottery number, the award is $1,000.
2. (Random character) Write a program that displays a random uppercase letter using the Math.random() method.
3. Given two sorted arrays, merge and return and a sorted array.
4. (Factorials) Factorials are used frequently in probability problems. The factorial of a positive integer n (written n! and pronounced “n factorial”) is equal to the product of the positive integers from 1 to n. Write an app that evaluates the factorials of the integers from 1 to 5. Display the results in tabular format. What difficulty might prevent you from calculating the factorial of 20?
5. (Find the smallest element) Write a method that finds the smallest element in an array of double values using the following header:

**public static double min(double[] array).**

1. (Reverse an array) The reverse method in Section 6.7 reverses an array by copying it to a new array. Rewrite the method that reverses the array passed in the argument and returns this array. Write a test program that prompts the user to enter ten numbers, invokes the method to reverse the numbers, and displays the numbers.
2. (Eliminate duplicates) Write a method that returns a new array by eliminating the duplicate values in the array using the following method header:

**public static int[] eliminateDuplicates(int[] list)**

Write a test program that reads in ten integers, invokes the method, and displays the result. Here is the sample run of the program:

**Enter ten numbers: 1 2 3 6 4 5 1 2 3 2 1 6 3 4 5 2**

**The distinct numbers are: 1 2 3 6 4 5.**

1. (Factorials) The factorial of a nonnegative integer n is written as n! (pronounced “n factorial”) and is defined as follows: n! = n · (n – 1) · (n – 2) · … · 1 (for values of n greater than or equal to 1 ) and n! = 1 (for n = 0) For example, 5! = 5 · 4 · 3 · 2 · 1, which is 120. Write an app that reads a nonnegative integer and computes and displays its factorial.
2. (Infinite Series: Mathematical Constant e) Write an app that estimates the value of the mathematical constant e by using the formula

The predefined constant Math.E (class Math is in the System namespace) provides a good approximation of e. Use the WriteLine method to output both your estimated value of e and Math.E for comparison.

1. (Infinite Series: ex) Write an app that computes the value of ex by using the formula

Compare the result of your calculation to the return value of the method call Math.Pow(Math.E, x) [Note: The predefined method Math.Pow takes two arguments and raises the first argument to the power of the second. We discuss Math.Pow in Section 6.4.

1. (Prime Numbers) An integer is said to be prime if it’s greater than 1 and divisible by only 1 and itself. For example, 2, 3, 5 and 7 are prime, but 4, 6, 8 and 9 are not. a) Write a method that determines whether a number is prime. b) Use this method in an app that displays all the prime numbers less than 10,000. c) Initially, you might think that n/2 is the upper limit for which you must test to see whether a number is prime, but you need only.
2. (Reversing Digits) Write a method that takes an integer value and returns the number with its digits reversed. For example, given the number 7631, the method should return 1367. Incorporate the method into an app that reads a value from the user and displays the result.
3. (Computer-Assisted Instruction) The use of computers in education is referred to as computer-assisted instruction (CAI). Write a program that will help an elementary school student learn multiplication. Use a Random object to produce two positive one-digit integers. The program should then prompt the user with a question, such as How much is 6 times 7? The student then inputs the answer. Next, the program checks the student’s answer. If it’s correct, display the message "Very good!" and ask another multiplication question. If the answer is wrong, display the message "No. Please try again." and let the student try the same question repeatedly until the student gets it right. A separate method should be used to generate each new question. This function should be called once when the app begins execution and each time the user answers the question correctly.
4. (Computer-Assisted Instruction: Reducing Student Fatigue) One problem in CAI environments is student fatigue. This can be reduced by varying the computer’s responses to hold the student’s attention. Modify the program of Exercise 7.39 so that various comments are displayed for each answer. Possible responses to a correct answer: Very good! Excellent! Nice work! Keep up the good work! Possible responses to an incorrect answer: No. Please try again. Wrong. Try once more. Don't give up! No. Keep trying. Use random-number generation to choose a number from 1 to 4 that will be used to select one of the four appropriate responses to each correct or incorrect answer. Use a switch statement to issue the responses.
5. (Find the highest score) Write a program that prompts the user to enter the number of students and each student’s name and score, and finally displays the name of the student with the highest score.
6. (Find the two highest scores) Write a program that prompts the user to enter the number of students and each student’s name and score, and finally displays the student with the highest score and the student with the second-highest score.
7. (Simulation: heads or tails) Write a program that simulates flipping a coin one million times and displays the number of heads and tails.
8. (Multiples of 2) Write an app that keeps displaying in the console window the powers of the integer 2—namely, 2, 4, 8, 16, 32, 64 and so on. Loop 40 times.
9. (Display characters) Write a method that prints characters using the following header: public static void printChars(char ch1, char ch2, int numberPerLine) This method prints the characters between ch1 and ch2 with the specified numbers per line. Write a test program that prints ten characters per line from 1 to Z. Characters are separated by exactly one space.