[**CSE 1310**](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/index.html) **-** [**Assignments**](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/index.html) **- Programming Assignment 6**

The assignment will be graded out of 100 points.

Some tasks ask you to write code, and specify what name to use for the file where you save that code. You need to use exactly the name that is given (do not change the case, or make any other modification). Remember, the name of the main class must match the filename.

For some tasks you need to answer questions. Create a text document entitled answers.txt, or answers.docx, or answers.pdf, and put all your answers there. Acceptable file formats are plain text, Word document, OpenOffice document, and PDF. Put your name and UTA ID in the file on the first line.

Each task below will instruct you where to put your answers.

**Task 1 (10 pts.)**

public class task1

{

public static int foo(int a, int b)

{

if (b == 0)

{

return 1;

}

return a \* foo(a, b-1);

}

public static void main(String[] args)

{

int a = 3;

int b = 5;

int c = foo(b, a);

System.out.printf("c = %s\n", c);

}

}

If you execute this program, what will be printed? Put your answer in your answers file.

**Task 2 (10 pts.)**

public class task2

{

public static int foo(int a, int b)

{

if (b <= 1)

{

return 1;

}

return a \* bar(a, b-2);

}

public static int bar(int c, int d)

{

if (c <= 1)

{

return 1;

}

return d + foo(d, c);

}

public static void main(String[] args)

{

int a = 3;

int b = 5;

int c = foo(b, a);

System.out.printf("c = %s\n", c);

}

}

If you execute this program, what will be printed? Put your answer in your answers file.

**Task 3 (15 pts.)**

File [task3.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment6/task3.java) contains an incomplete program. The goal of the program is to take as input strings from the user, and then print out coded (encrypted) versions of those strings, by replacing each letter with another letter. Complete that program, by defining a print\_coded function, that satisfies the following specs:

* Function print\_coded takes three arguments, called word, sources, targets. They are all strings.
* Function print\_coded processes the letters of word one by one, in the order in which they appear in word. For every such letter X, the function processes X as follows:
  + If X is equal to the character at position P of sources, then the function prints the character at position P of targets.
  + Otherwise, the function prints X.

Note that arguments sources and targets are hardcoded in the main function, the user cannot change those values. The user can only specify the value of word.

**IMPORTANT: you are only allowed to modify the provided code by writing the print\_coded function. You are NOT allowed to modify in any way the main function.**

This is an example run of the complete program:

Enter some word, or q to quit: hello

ifmmp

Enter some word, or q to quit: HELLO

HELLO

Enter some word, or q to quit: Arlington

Asmjohupo

Enter some word, or q to quit: Texas

Tfybt

Enter some word, or q to quit: q

Exiting...

Your program's output should match EXACTLY the format shown above. There should be no deviations, no extra spaces or lines, no extra punctuation in your output. What you see above as uppercase letters should remain uppercase in your output, what you see as lowercase letters should remain as lowercase in your output, what you see as spaces and punctuation should remain exactly as spaces and punctuation in your output.

**Task 4 (15 pts.)**

File [task4.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment6/task4.java) contains an incomplete program. The goal of the program is to take as input sentences from the user, and then print out the number of words in each sentence. Complete that program, by defining a count\_words function, that satisfies the following specs:

* Function count\_words takes one argument, called text, which is a string, and which is allowed to contain only uppercase letters, lowercase letters, and the space character ' '. In other words, don't worry about handling cases cases where text contains characters that are punctuation, special symbols, or anything else that is not a letter or the space character.
* Function count\_words counts and returns the number of words in text.

The strategy for counting words is simple:

1. You initialize your counter to zero if your text starts with a space character, or to 1 if your text starts with a letter.
2. You go through the text, character by character. You increment your counter every time you find a letter whose previous character was space. For example, if your current character is 'm' and the previous character was ' ', it means that you have found a new word.

It may be that the text starts with multiple spaces, or that multiple spaces are placed between two words. Your code should handle that correctly.

**IMPORTANT: you are only allowed to modify the provided code by writing the count\_words function. You are NOT allowed to modify in any way the main function.**

This is an example run of the complete program:

Enter some text, or q to quit: this is the fourth week of the semester

Counted 8 words.

Enter some text, or q to quit: hello world

Counted 2 words.

Enter some text, or q to quit:

Counted 0 words.

Enter some text, or q to quit: h hhg

Counted 2 words.

Enter some text, or q to quit: q

Exiting...

Your program's output should match EXACTLY the format shown above. There should be no deviations, no extra spaces or lines, no extra punctuation in your output. What you see above as uppercase letters should remain uppercase in your output, what you see as lowercase letters should remain as lowercase in your output, what you see as spaces and punctuation should remain exactly as spaces and punctuation in your output.

**Task 5 (15 pts.)**

File [task5.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment6/task5.java) contains an incomplete program, whose goal is to take as input text from the user, and then print out that text so that each letter of the text is repeated multiple times. Complete that program, by defining a repeat\_letter function, that satisfies the following specs:

* repeat\_letter takes two arguments, called text, times.
* The function goes through the letters of text in the order in which they appear in text, and prints each such letter as many times as specified by the argument times.

This is an example run of the complete program:

Enter some text, or q to quit: hello

Enter number of times (must be > 0): 3

hhheeellllllooo

Enter some text, or q to quit: good morning

Enter number of times (must be > 0): 2

ggoooodd mmoorrnniinngg

Enter some text, or q to quit: a b

Enter number of times (must be > 0): 7

aaaaaaa bbbbbbb

Enter some text, or q to quit: q

Exiting...

Your program's output should match EXACTLY the format shown above. There should be no deviations, no extra spaces or lines, no extra punctuation in your output. What you see above as uppercase letters should remain uppercase in your output, what you see as lowercase letters should remain as lowercase in your output, what you see as spaces and punctuation should remain exactly as spaces and punctuation in your output.

**Task 6 (15 pts.)**

File [task6.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment6/task6.java) contains an incomplete program, that takes as input from the user a year and a month, and prints the number of days in that year and in that specific month of the year. Complete that program, by defining two functions. The first function is called year\_days, and should satisfy the following specs:

* year\_days takes one argument, called year.
* If year is a leap year, then year\_days(year) returns 366.
* Otherwise, year\_days(year) returns 365.

The second function is called month\_days, and should satisfy the following specs:

* month\_days takes two arguments, called year, month. The month argument is an integer between 1 and 12.
* The function returns the number of days that the month lasts for. Unless the month is 2 (for February), the result does not depend on the year. If the month is 2, then obviously the result should be 28 or 29, depending on whether the year is a leap year.

**IMPORTANT: you are only allowed to modify the provided code by writing the year\_days and months\_days functions. You are NOT allowed to modify in any way the main function.**

Hint: our [LeapYear.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/codebase/05a_conditionals/LeapYear.java) program has code that you may find useful, and you should feel free to reuse.

This is an example run of the complete program:

Enter a year (must be > 0): 1900

Year 1900 has 365 days.

Enter a month(must be between 1 and 12): 2

Month 2, 1900 has 28 days.

Enter a year (must be > 0): 2000

Year 2000 has 366 days.

Enter a month(must be between 1 and 12): 2

Month 2, 2000 has 29 days.

Enter a year (must be > 0): 2001

Year 2001 has 365 days.

Enter a month(must be between 1 and 12): 2

Month 2, 2001 has 28 days.

Enter a year (must be > 0): 2005

Year 2005 has 365 days.

Enter a month(must be between 1 and 12): 12

Month 12, 2005 has 31 days.

Enter a year (must be > 0): q

Exiting...

Your program's output should match EXACTLY the format shown above. There should be no deviations, no extra spaces or lines, no extra punctuation in your output. What you see above as uppercase letters should remain uppercase in your output, what you see as lowercase letters should remain as lowercase in your output, what you see as spaces and punctuation should remain exactly as spaces and punctuation in your output.

**Task 7 (20 pts.)**

File [task7.java](http://vlm1.uta.edu/%7Eathitsos/courses/cse1310_fall2015/assignments/assignment6/task7.java) contains an incomplete program, that takes as input from the user a date and outputs two things: the number of days that have passed from 12/31/999 up to the specified date, and the day of the week for that specified date. Complete that program, by defining two functions. The first function is called days\_passed, and should satisfy the following specs:

* Function days\_passed takes three arguments, called year, month, day.
* The function returns the number of days that have passed from 12/31/999 to the specified date.

Hint: to compute that, use your solutions from Task 6. Do a for loop from 1000 up to (and not including) the specified year, adding up the days of each year in the loop. Then, do another for loop, from 1 up to (and not including) the specified month, adding up the days of each month in the loop. Finally, add the specified day. The second function is called day\_of\_week, and should satisfy the following specs:

* Function day\_of\_week takes three arguments, called year, month, day.
* The function returns the day of the week (e.g, Sunday, Tuesday, and so on) that corresponds to the specified date.

Hint: the answer depends only on the remainder of dividing by 7 the result of days\_passed(year, month, day). For example, if the remainder is 0 the function should return "Tuesday", if the remainder is 1 the function should return "Wednesday", and so on.

**IMPORTANT: You are NOT allowed to modify in any way the main function.**. You are allowed to modify the provided code by writing the days\_passed and day\_of\_week functions, and any other auxiliary functions that you may find useful. You should also put in your code your solution for month\_days from Task 6, and you are free to also put in your code and use your solution for year\_days from Task 6.

This is an example run of the complete program:

Enter a year (must be >= 1000): 1000

Enter a month(must be between 1 and 12): 1

Enter a day: 1

1 days have passed from 12/31/999 to 1/1/1000.

1/1/1000 is a Wednesday.

Enter a year (must be >= 1000): 2015

Enter a month(must be between 1 and 12): 8

Enter a day: 4

370937 days have passed from 12/31/999 to 8/4/2015.

8/4/2015 is a Tuesday.

Enter a year (must be >= 1000): 1776

Enter a month(must be between 1 and 12): 7

Enter a day: 4

283614 days have passed from 12/31/999 to 7/4/1776.

7/4/1776 is a Thursday.

Enter a year (must be >= 1000): q

Exiting...

Your program's output should match EXACTLY the format shown above. There should be no deviations, no extra spaces or lines, no extra punctuation in your output. What you see above as uppercase letters should remain uppercase in your output, what you see as lowercase letters should remain as lowercase in your output, what you see as spaces and punctuation should remain exactly as spaces and punctuation in your output.

**Suggestions**

Pay close attention to all specifications on this page, including file names and submission format. Even in cases where the program works correctly, points will be taken off for non-compliance with the instructions given on this page (such as wrong file names, wrong compression format for the submitted code, and so on). The reason is that non-compliance with the instructions makes the grading process significantly (and unnecessarily) more time consuming. Contact the instructor or TA if you have any questions.

**How to submit**

The assignment should be submitted via [Blackboard](http://elearn.uta.edu). Submit a ZIPPED directory called assignment6.zip (no other forms of compression accepted, contact the instructor or TA if you do not know how to produce .zip files).

To create a zipped directory called assignment6.zip, follow these steps:

1. Create a folder called assignment6.
2. Copy to that folder all your solutions (your answers file, and all your Java files).
3. Zip that folder. On windows, you can zip a folder by right-clicking on the folder, and then selecting Send to->Compressed (zipped) folder.

Your zip file should contain only 6 files: your answers document and all the Java code files (task3.java, task4.java, task5.java, task6.java, task7.java).

**Submission checklist**

* Did you create the answers file with your name, UTA ID, and answers to non-programming tasks?
* Did you zip everything into a file called assignment6.zip?