COM1003 Java Programming
Autumn Semester 2022-3
Programming Assignment 2

Dr Siobhán North
Department of Computer Science, The University of Sheffield

Learning outcomes

This assignment will assess your ability to:

- Write a program from a specification;
- Write clear, good quality program code;
- Use loops in Java;
- Use the `sheffield` package for graphical output;
- Use arrays in Java.
- Use the Math class in Java

Your Program

You must write a program to read in information from a file and use it to display a picture. As in the earlier assignment, there are various increasingly difficult versions of this task each with an associated maximum number of marks that can be obtained but this time each task does not directly include the previous one although each version builds on skills you could develop in doing a simpler version.

Unless you are already a competent programmer I suggest you attempt the easiest one, get it to work and back up your program (and possibly upload it to Blackboard) before you attempt the next one by modifying the previous version. And if you are a competent programmer at least read the specifications for the easier versions because they contain information you are going to need.

For all versions the program you upload must be called `Assignment2.java`. Blackboard will present me with the last version you uploaded before the deadline to mark so the last version you upload should be the hardest you can make compile and run. Do not worry that they are all called `Assignment2.java`; I will to be able to recognise which version you are attempting from the output. Handing in only *one* version which must be called `Assignment2.java` is part of the specification.

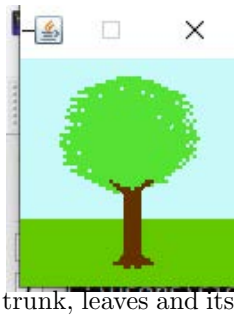
As before your program it must compile and run to get any marks and, in this assignment, it must do exactly one of the tasks to get any of the marks.

So a well written program that does something is always better than a program which would do more if it had worked. It is worth 20% of your mark for the first semester of the module and must be submitted by 25 November 2022. You will find information about the exact deadline, the marking scheme and how you must submit your work at the end of this document.

The specification is typical of real specifications in that, although it is not meant to be ambiguous, it is not going to tell you every detail of the program you must write. Be careful not to make any assumptions without checking them. As with the previous assignment there will be an FAQ page. Even if you think you understand everything and have assumed nothing it is a good idea to check this page before you submit.

The Simplest Program - a Small Tree

You have been provided with a data file called `tree.txt`. Your program must read the 15,360 upper case letters in the data file which you should store in the same directory as your program, whatever directory that is because that is what will happen when your program is marked. The 15,360 characters are encrypted but can be used to form a rudimentary picture of a tree. The encryption is very simple; any letter whose numeric value is even is part of the background and letters whose numeric value are odd represent the tree. If the letter's numeric value is divisible by 5 it is the trunk and any other odd number is a leaf. They are arranged so that the top line of the picture makes up the first 120 characters, the next 120 is the second line of the picture and so on with the last 120 as the bottom line of the picture.



There are no line breaks or other layout characters and nothing else except upper case letters. The data file your program will be tested against conforms to exactly the same rules although it my represent a different tree.

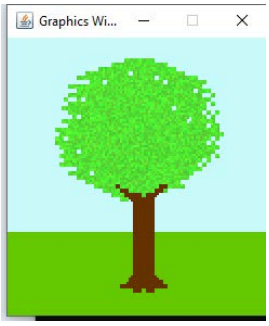
As you read in the characters, ideally one by one, you should store the information from the file in a suitable array then use the information in the array to display the tree in a graphics window with 128 rows of 120 pixels using the `EasyGraphics` class. The choice of colour is up to you as long as it is easy to distinguish between the tree's trunk, leaves and its background. The background consists of sky at the top and grass at the bottom with an horizon leaving the bottom 30% of the background green and the rest blue.

It is possible to read the file and display the tree without using an intermediate array but if you do that it will have a significant effect on your marks (the marking scheme is at the end of this document) and, if you are planning to go on to the more difficult tasks, you will need it.

Please note a suitable array is one that makes it easy to draw a tree not one that is a clone of the data file.

If you do this with a perfectly written program that demonstrates all the techniques identified in the marking scheme you can expect to get 30%.

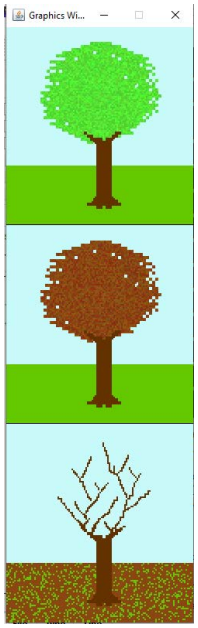
A Bigger Tree



This is the almost same as the previous version except scaled up so that so that both the screen and the picture are four times the area. Thus the graphics window should be 256 rows and 240 columns of pixels and the picture should be scaled up in proportion. It is not quite the same as the previous version because the leaves now have some random variation of colour. You need to this for full marks but can still get most of the marks for a bigger tree with plain green leaves.

If you do this with a perfectly written program that demonstrates all the techniques identified in the marking scheme you can expect to get 40% provided you don't sumbit the earlier version as part of the same program. If you submit both as one program you will get zero.

A Seasonal Picture



This time you need to display the tree three times in a graphics window exactly big enough for three versions of the previous picture one about the other with two black lines one pixel high in between the top and middle and middle and last pictures.

The top picture is the same as the one in the earlier exercise, in the middle one the tree's leaves have turned brown, again with random variations of colour, and in the bottom picture the leaves have gone leaving just branches and the grass randomly patterned in green and brown with 20% of the ground green and the rest brown. The bare branches are in the original data file. Letters whose numeric value is divisible by 7 are branches.

If you do this with a perfectly written program that demonstrates all the techniques identified in the marking scheme you can expect to get 55%.

If you submit this version you should *not* submit the 30% or 40% versions.

A Forest

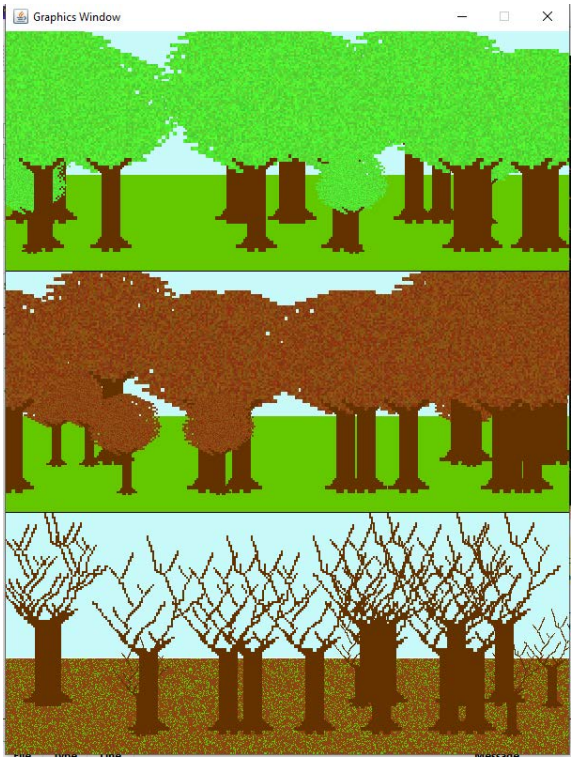
This time the graphics screen should be five times the width of the original picture and twice its height and should contain between 15 and 25 trees with the number generated at random each time the program runs. The background should be the similar to the earlier versions except with the horizon 40% of the way up the window.You will find it a lot easier if you generate the background first and add the trees afterwards.

The trees should randomly placed horizontally but arranged in two horizontal lines vertically. Half of them, and it should be the half you draw first, should start with their base at about 30% of the distance between the bottom of the screen and the horizon and the second with their base at the bottom of the screen.

The trees should mostly be the scaled up size of the second task but with 20% of them the smaller original version. You will need to draw these small trees a bit higher up the screen than the bigger ones or you can get some odd effects when they overlap. All the trees should have slightly varying leaf colour.

If you do this with a perfectly written program that demonstrates all the techniques identified in the marking scheme you can expect to get 80%. If you submit this version you should *not* submit any of the other versions

A Seasonal Forest



This is the forest version of the Seasonal Picture task. This time you need three versions of the task above one on top of the other in a single graphics window with pixel thick horizontal black lines between them. The top forest should be the summer version, the middle one Autumn and the bottom one Winter.

The random elements of the pictures, the number of trees and their positions horizontally across the window should be generated anew for the Summer, Autumn and Winter versions so the picture should not show the same trees in the same places in each season.

If you do this with a perfectly written program that demonstrates all the techniques identified in the marking scheme you can expect to get 100%. If you submit this version you should *not* submit any other version in the same program.

Submission and deadline

You must submit a file called `Assignment2.java` via the submission point on Blackboard by 3pm on Friday 25th November. Do not submit anything else and do not submit it in any other format. Blackboard is very bad at displaying Java programs so you may think that Blackboard has messed up the layout of your program but I will download the program text and the layout will be preserved.

Late work will be penalised using the standard University scale (a penalty of 5% per working day late; work will be awarded a mark of zero if it is more than 5 working days late).

This is an individual assignment. You must work on it alone and hand in your own work. If you work collaboratively and then pretend you did the work alone we will find out (we have a very good plagiarism checker and all submitted work will go through it) and, as you have already been told, we take the use of unfair means in the assignment process very seriously. Don't even think about handing in work you didn't do yourself.

The Marking Scheme

The mark for this assignment is worth 20% of the first semester mark and so 10% of the overall mark for COM1003.

The marking scheme for the various versions is as follows:

Version	Small Tree	Big Tree	Autumn Scene	Forest	Autumn Forest
EasyReader	2	2	2	2	2
Arrays	3	5	6	8	10
Loops	4	4	5	6	8
EasyGraphics	4	5	10	12	15
Calculation	2	4	5	12	15
Specification	3	4	5	8	10
Style	12	16	22	32	40
Total	30	40	55	80	100

The style mark will be calculated as a percentage as follows and scaled to

Good use of identifiers	20
Good use of types	25
Good use of comments	20
Good use of constants	15
Readability	20
Total	100

the maximum for whatever version you submit. Readability encompasses layout, indention and a clear, consistent structure.

The criteria for program style quality are similar to the first assignment so remember to check your feedback for that assignment before it is too late to submit a new version of this assignment. It is a good idea to check the FAQs then too.