Programming

Year One

Semester One

Andrew Dunkerley

Top Tips

You can’t use a variable before it has been declared

Look on documentation on VLE for help

Tab programs to make them easier to read

Shorter codes are easier to find bugs in.

Algorithm – A computable set of steps to achieve a desired result.

Always start as simple as possible

Put close brackets as soon as you open brackets then type between

You must understand the code (don’t just randomly add code)

NAN = Not a number

JavaScript is executed in the order it is written

Refine the solution to add complexity

Methodology – Always follow a route that is successful. (follow the track in a mud field )

Test the system frequently

Write a bit, and then test a bit

= assign value to

== equals to

!= not equal to

< less than

> greater than

<= less than or equal to

>= more than or equal to

Exercise 1.1 Welcome to the IPP JavaScript Environment

Initial Code

var phrase1;

phrase1 = "Hello, ";

var phrase2;

phrase2 = ", welcome to the Programming module.";

var name;

name = prompt( "Please enter your name:" );

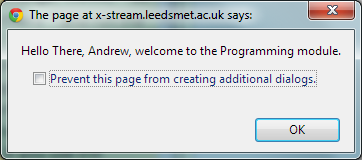
var message;

message = phrase1 + name + phrase2;

alert( message );

Exercise 1.1 Task 1.

In this first part of this task I had to changed “Hello” like it was on the exercise to “Hello there”

As you can see from this screenshot the output box now reads hello there instead of hello.

Exercise 1.1 Task 2

You can always recover from your changes by correcting them yourself − if you can remember what you did − or, more easily, by clicking the "Reload Initial Code" button. If you use the button, you will be advised that you will lose any changes − which is what you want!

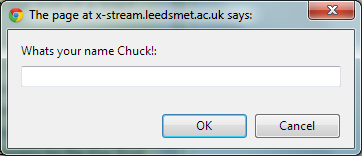
Exercise 1.1 Task 3

In this third task I had to change the text in the other quotes. For example it stated that the codes had to be changed

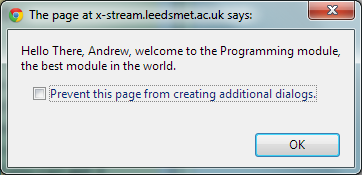
var phrase2;

phrase2 = ", welcome to the Programming module, the best module in the world.";

var name;

name = prompt( "Whats your name Chuck!:" );

As we can see from this screenshot the changes I have made have worked and the prompt has altered to what I have imput.

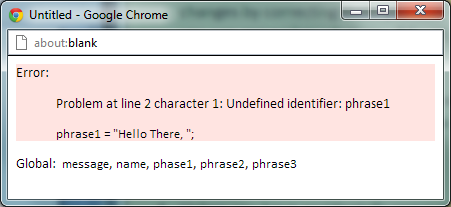


This screenshot on the right is the final product of the changes I have made. You can see that any changes I have made have worked and there are correctly displayed like they should be.

Exercise 1.1 Task 4

In this task I had to purposely spell a function work incorrectly to see the effects.

var phase1;

phrase1 = "Hello There, ";

I have incorrectly spelt phrase1 as phase1 to see the effects. As you can see a syntax error has accursed stating that there has been an “unidentified identifier”. This has occurred as anticipated.

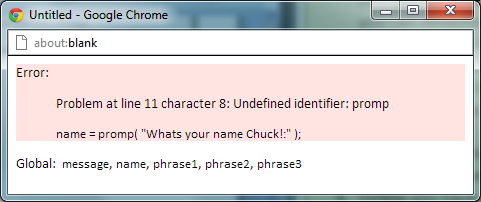
Exercise 1.1 Task 5

My next task was to misspell “alert” or “prompt” commands.

var name;

name = promp( "Whats your name Chuck!:" );

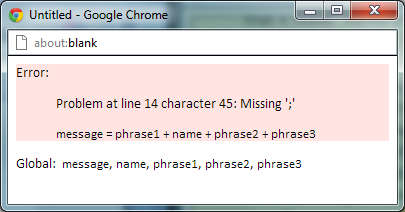
I have incorrectly spelt prompt t see the effects that it has on the program instead of spelling prompt I have spelt promp.

As we can see from this screenshot the there has been another unidentified identifier. This means that the syntax error has worked correctly like it should do.

Exercise 1.1 Task 6

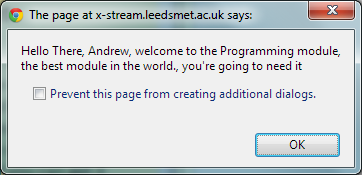
My task was to remove a semicolon to see the effects.

var message;

message = phrase1 + name + phrase2 + phrase3

As we can see from this code there is no semicolon at the end of this code to execute it.

As a result we are presented with this output box.

This output box states that line 14 is missing a semi colon. This means that the syntax error has worked correctly. And the program has worked correctly

Exercise 1.1 – A final Experiment.

In this task I had to add a third phrase in it to expand the code.As we can see here I have add a third phrase “you’re going to need it”

The code for this was

var phrase3;

phrase3 = ", you're going to need it ";

Exercise 1.2 Graphical Welcome

This program makes use of a different form of output compared with the previous one. Instead of a single line dialogue box, this uses a graphical area on which text and geometrical shapes can be drawn.

If you look at the check box beneath the code pane, you'll see that it is ticked indicating that this is a graphical program.

Load up the initial code and look carefully at it. You should see a number of lines that are exactly the same as in the last program. However, there is one extra line at the start, a different line of code used to output the message and one extra line at the end of the program.

Run the program and notice that a new window opens in which the text is displayed.

Initial Code

var canvas;

canvas = openGraphics();

var phrase1;

phrase1 = "Hello, ";

var phrase2;

phrase2 = ", welcome to the Programming module.";

var name;

name = prompt( "Please enter your name:" );

var message;

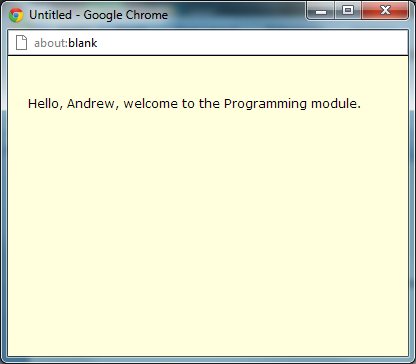
message = phrase1 + name + phrase2;

canvas.drawString( message, 20, 40 );

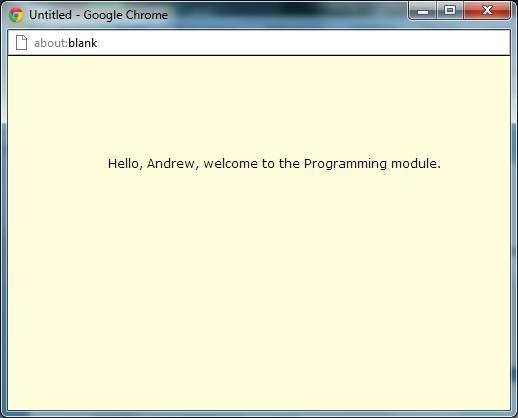
canvas.paint();

Exercise 1.2 Task 1

My first task was to change the values of the two numbers in the drawstring line.

 This is the initial Image which hasn’t been altered in any way.

canvas.drawString( message, 20, 40 );



We can see from this screenshot that the location of the text has move dramatically. As I have changed the values in the canvas.drawString code.

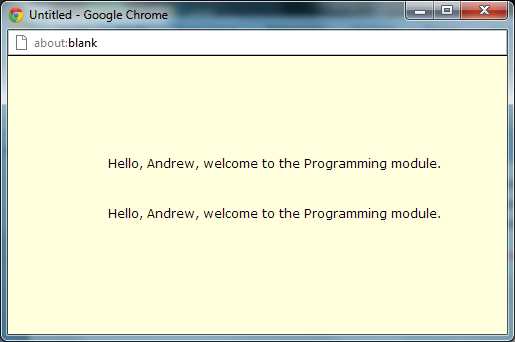
These values represent the X and the Y coordinates of the text and this in turn effects where the text is placed.

canvas.drawString( message, 100, 100 );

Exercise 1.2 Task 2

My second task was to duplicate the output line so I get more than one line of output on the screen.

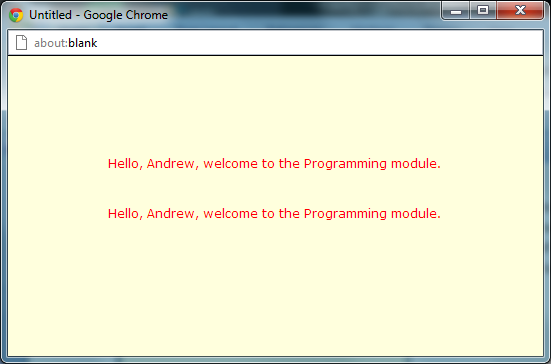
The way in which I did this was simply by copy the initial line of code which was the canvas.drawString( message, 100, 100 ); and pasting a new line of code directly underneath it. The end result of this process was.

As we can see from this screenshot the output box now has two lines of code within it. So as a result I can safely say that the code I input has worked correctly.

Exercise 1.2 Task 3  
In this third task on this exercise I had to add a new line of code before the drawstring lines.

The exact piece of code which used before the drawstring line was

canvas.setColor( "red" ); This piece of code would set the colour of the text to red or any other colour which I put in between the “ marks.

As we can see from this screenshot the code I have input has worked and I have been presented with two line of code which have been coloured in red.

The code which I was left with was

canvas.setColor( "red" );

canvas.drawString( message, 100, 100 );

canvas.drawString( message, 100, 150 );

Exercise 1.2 task 4

This task involved me haveing to add another piece of code. This piece of code would expand the output text and change it to show something different.

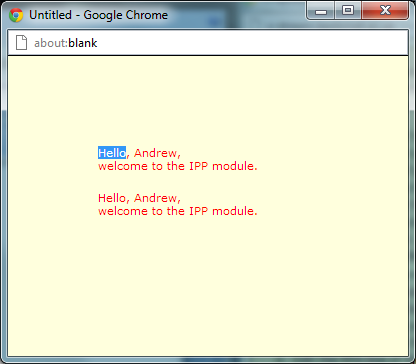
The new piece of code which I implemented into my program was

phrase2 = ",<br />welcome to the IPP module.";

This was to replace current piece of code.

var phrase2;

phrase2 = ", welcome to the Programming module.";

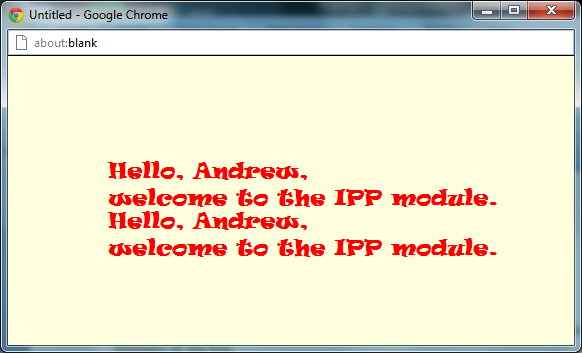


As we can see from this screenshot the text has changed, which means that the code I have input is correct and the code is functioning properly.

Exercise 1.2 Task 5

The final part of this task was add a final piece of code the code being

canvas.setFont( "times", "20px", Font.BOLD ); this code would change the font and the thickness of the font.

As we can see from this screenshot the code which I have input has worked correctly and the difference can be seen on this screenshot compared to previous screenshot from this exercise.

canvas.setFont( "ravie", "20px", Font.BOLD ); This was the exact code I used.

Exercise 1.3 – Expressions

This exercise provides some practice for entering some expressions and checking that the results are correct. You should work out what you expect to get for each of the tasks.

Load up the initial code into the editor pane. You will see that some identifiers have been declared and initialised (though in this case all to 0). There is an expression whose result is labelled and output using an alert().

Edit the program so that you can output results as follows (remember to work out what you expect so you know if you have done it correctly.)

Exercise 1.3 Task 1

In this task I has to find the resulting value when you

a + b – c

where a is 5, b is 15 and c is 7

I changed the values of A, B and C according to what they were in the question and changed the additions and subtractions to what the exercise asked for.

var a;

a = 5;

var b;

b = 15;

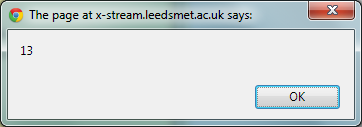
var c;

c = 7;

var result;

result = a + b - c;

alert( result );

This is the code that was left with and when I ran it I was presented with this output box.

As we can see the computer has provided us with 13 as a resulting value and this is correct so I can say they program has worked.  
Exercise 1.3 Task 2

In this task I has to find the resulting value when you

ab + bc + ac

Where a is 2, b is 3 and c is 4

Remember, ab means a times

I changed the values of A, B and C again according to what they were in the question and changed the additions and subtractions to what the exercise asked for.

var a;

a = 2;

var b;

b = 3;

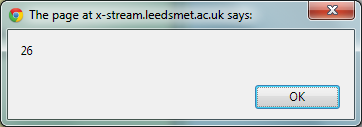
var c;

c = 4;

var result;

result = a \* b + b\*c + a\*c ;

alert( result );

This was the code that I was left with and when I ran it I was presented with this output box.

As we can see from this task when I ran the program the value that I was presented with was 26. This is the correct number and the number which I wanted to see. So therefore I can say that the code that I input was correct

Exercise 1.3 Task 3

In this task I has to find the resulting value when you

(a + b)(a - b)

where a is 5 and b is 4

I changed the values of A, B and C again according to what they were in the question and changed the additions and subtractions to what the exercise asked for.

var a;

a = 5;

var b;

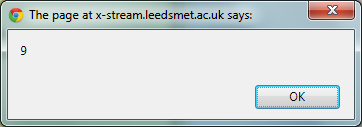
b = 4;

var result;

result = (a+b) \* (a-b);

alert( result );

This was the code that I was left with and when I ran it I was presented with this output box

As we can see from this screenshot the output box which I have been presented with has a value of 9.

This is the correct value and therefore shows that the code which Input into the program was correct and has work efficiently.

Exercise 1.3 – My thoughts

I felt that this program was a very easy one. The complications were simple to understand and the coding was also very simple. The only problem which I encountered when doing this exercise was confusing the times signs I thought the signs were the standard X sign but in fact time was actually the \* sign.

Exercise 1.4 Shape Area

Initial Code

var canvas;

canvas = openGraphics();

var size;

size = 50;

canvas.drawRect( 10, 10, size, size );

// drawRect, short for drawRectangle, requires four values:

// drawRect( x, y, width, height );

// x and y are the coordinates of the top left corner

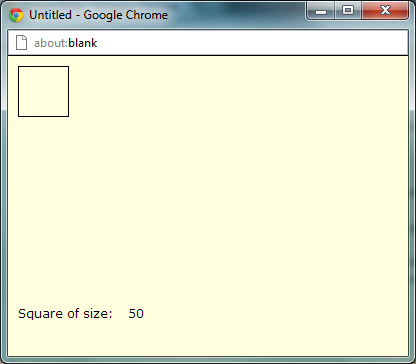
// width and height are what they say!

// Setting width and height to the same value draws a square

canvas.drawString( "Square of size:", 10, 250 );

canvas.drawString( size, 120, 250 );

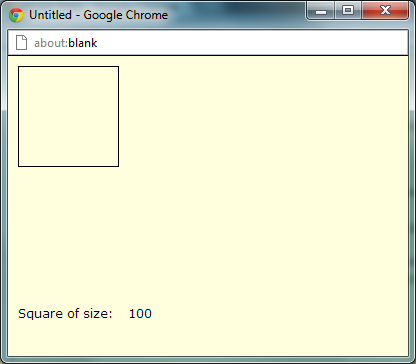
canvas.paint();

When we run the initial code we are presented with a standard square shape.

As we can see from this screenshot the initial code has provided us with a standard square shape and the size of the square heading and it value.

Exercise 1.4 Task 1

In this 1st task I had to change the size of the square.

As we can see from this screenshot compare to the previous one, the square as majorly increased in size.

var size;

size = 100;

This was the code I changed; I changed the size value from 50 to 100

Exercise 1.4 Task 2

My next task was to add a new variable called area and assign to it the area of the square.

var area;

area= size\*size;

This code to the right has been altered. I have added new piece of code, Instead of having one variable named size. I have another variable called area.

canvas.drawRect( 10, 10, size, size );

canvas.drawString( "Square of size:", 10, 250 );

canvas.drawString( size, 120, 250 );

Exercise 1.4 Task 3

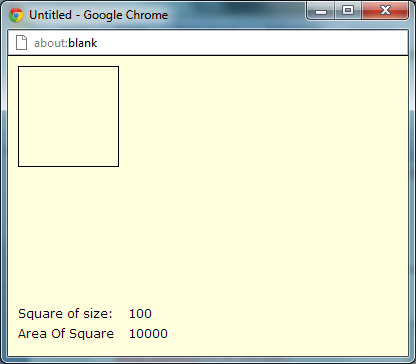
The third task involved adding two new pieces of code so that the value and the variable of the new code can be presented underneath the already existing variable.

The code which I input so that the new variable could be display was a canvas.drawString code. This attached the text to a string on the canvas.

The exact pieces of code I used was

canvas.drawString("Area Of Square",10,270);

canvas.drawString( area, 120, 270 );

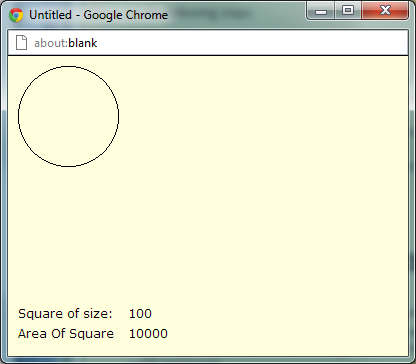
As we can see from this screenshot the “area of the square” variable and its value of have been displayed correct. This means that the drawstring code I input has worked correctly.

Exercise 1.4 Change to a circle

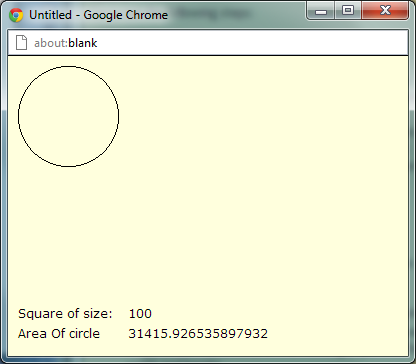
This task involved me having to change the square which I started with to a circle. The way in which this can be done is through the drawEllipse command is similar to the drawRect command instead it draw a circle.

The piece of code which I change now looks like this.

canvas.drawEllipse( 10, 10, size, size );

This piece of code is now going to display a circle instead of a square.

As we can see from this screenshot the drawEllipse code which I just input has worked correctly and it now displaying a circle.

I am now looking to add a piece of code which will calculate the area of this circle the exact piece of

code which I will input is

area= (size\*size)\*(Math.PI);

As a result I will be presented with a circle and the area will be correct.

As we can see from this screenshot the area of circle has been calculated and it is correct. This additionally means that the code which I have input is correct and is functioning correctly.

Exercise 1.4 - complete code

var canvas;

canvas = openGraphics();

var size;

size = 100;

var area;

area= (size\*size)\*(Math.PI);

canvas.drawEllipse( 10, 10, size, size );

// drawRect, short for drawRectangle, requires four values:

// drawRect( x, y, width, height );

// x and y are the coordinates of the top left corner

// width and height are what they say!

// Setting width and height to the same value draws a square

canvas.drawString( "Square of size:", 10, 250 );

canvas.drawString("Area Of circle",10,270);

canvas.drawString( size, 120, 250 );

canvas.drawString( area, 120, 270 );

canvas.paint();

Exercise 1.4 – My thoughts.

I felt that this exercise was both hard and simple in some parts. The exercise was going very well until I had to find the area of the circle. I was using area = radius2 X pi but for whatever reason it was not working. Instead I used the area= (size\*size)\*(Math.PI); as the practically calculate the same way but different processes.

Exercise 1.5 Interest Rates

Simple interest rates can be calculated by multiplying together the amount to be saved (or borrowed!), the interest rate and the number of years over which the money is to be saved (or borrowed). Dividing the final result by 100 gives the amount of interest:

amount X rate X years

interest = ---------------------

100

Exercise 1.5 – Initial code

var amount;

amount = 250;

// declare, and initialise, identifiers for rate and time

var interest;

// input a formula to calculate the interest

alert( interest );

Exercise 1.5 – Task 1

My first task was to load the initial code

Exercise 1.5 - Task 2

My second task was to add two new variables alongside my already existing variable “amount”.

var rate;

rate = 0;

var time;

time = 0;

var years;

years = 1;

This was the code that I input into the initial code. These three variables will be used to calculate the interest value.

Exercise 1.5 – Task 3

My next task was to add a formula in which the program could use to calculate the amount of interest. The exact formula and code I used was this.

interest = (amount \* rate \* years)/(100);

Exercise 1.5 – Task 4

I ran this program and code and I was presented with a value.

var amount;

amount = 250;

var rate;

rate = 10;

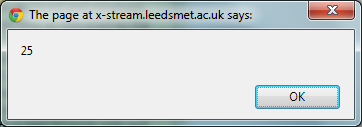
var years;

years = 1;

var interest;

interest = (amount \* rate \* years)/(100);

alert( interest );

As we can see from this screenshot the code has worked correctly and has calculated the value and interest of the value I input into the exercise program.

Exercise 1.5 – Task 5

I copied this work and information into my log book.

Exercise 1.5 – complete code

var amount;

amount = 250;

var rate;

rate = 10;

var years;

years = 1;

var interest;

interest = (amount \* rate \* years)/(100);

alert( interest );

Exercise 1.5 – my thoughts

I thought that this task was a very simple as all it required was the addition of two values and the implementation of a formula. I felt that the task was simple and I didn’t experience much difficulty completing it.

Exercise 1.6 framing a picture

You are going to display a digital image (stored as a JPEG file) and frame it like a printed image might appear. Load up the inital program and run it. You should see a picture of some honeysuckle. The image files are fairly sizeable and the first load might take a few seconds depending on the speed of your network link.

Exercise 1.6 - Initial code

var canvas;

canvas = openGraphics();

var imageName;

var xPosition;

var yPosition;

var width;

var height;

imageName = "Honeysuckle.jpeg";

xPosition = 0;

yPosition = 0;

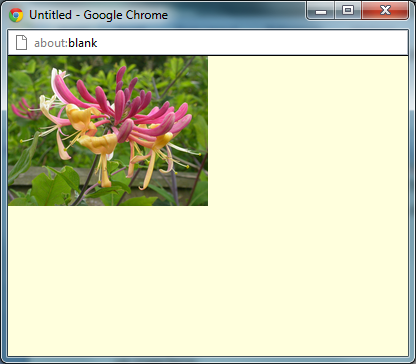
width = 200;

height = 150;

canvas.drawImage( imageName, xPosition, yPosition, width, height );

canvas.paint();

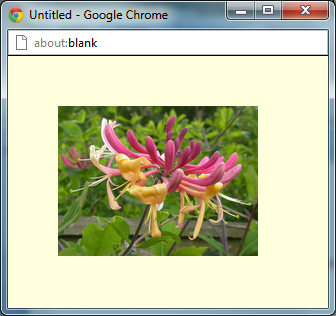
Exercise 1.6 Task 1

My first task on this exercise was to move the location of the output image. This can be easily done changing the values of the variables named “X” and “Y”.

As we can see from this screenshot from the unaltered code the picture remains in the top left hand corner of the screen.

xPosition = 0;

yPosition = 0;



However, when I changed the values of X and Y.

xPosition = 50;

yPosition = 50;

The picture has noticeable moved from the corner to the centre of the output box; this means that the change has been successful.

Exercise 1.6 Task 2

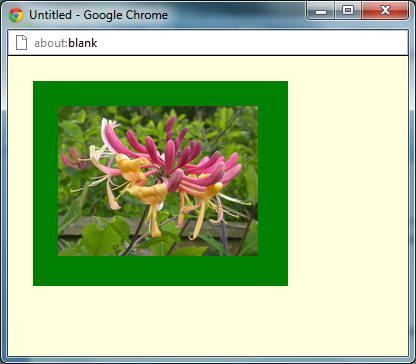
My second task on this exercise was to create a fill rectangle around my picture to act as a background.

The way in which you can create a background is through the fillRect code. The exact code which I used to create this back ground was

canvas.setColor( "green" );

canvas.fillRect( 25, 25, 255, 205 );

This code would create a green background around the perimeter of my picture.

As we can see from this screenshot the code

canvas.setColor( "green" );

canvas.fillRect( 25, 25, 255, 205 );

has worked and there is a green background to my picture.

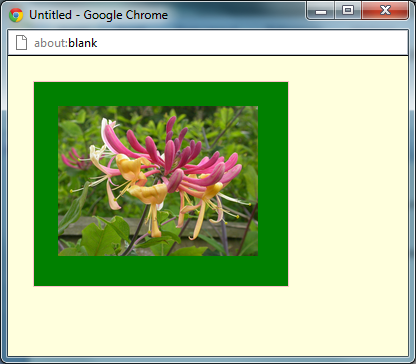
Exercise 1.6 Task 3

My next task in this exercise was to create a frame around my background. This can be done using the drawRect instead of the fillRect code.

I started this task by writing the code required to complete the task.

The code I used was

canvas.setColor( "pink" );

canvas.drawRect( 25, 25, 255, 205 );

As you can see from this screenshot the code has worked and I now have an image which has a border and that boarder is now framed.

Exercise 1.6 Task 4

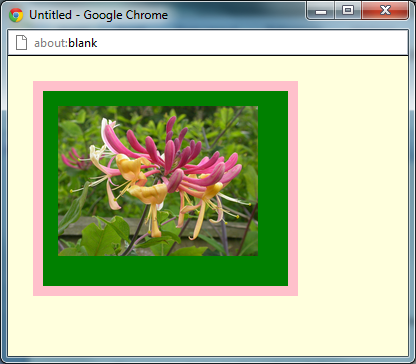
This next task involved me increasing the thickness of the frame as currently the frame I have now is very difficult to see.

They way in which I do this is through the code canvas.setstroke(n). This code will increase or decrease the thickness of the frame depends of the value which I put instead of “n”.

canvas.setStroke(10);

canvas.setColor( "pink" );

canvas.drawRect( 25, 25, 255, 205 );

This was the code which I was left and will make the changes I require to complete the task.

As we can see from this screenshot the new code

canvas.setStroke(10);

has worked correctly and the thickness of the boarder has increase and is now quite noticeable.

Exercise 1.6 Task 5

I recorded the work into my Log when I was happy enough with the results.

Exercise 1.6 Complete code

var canvas;

canvas = openGraphics();

var imageName;

var xPosition;

var yPosition;

var width;

var height;

imageName = "Honeysuckle.jpeg";

xPosition = 50;

yPosition = 50;

width = 200;

height = 150;

canvas.setColor( "green" );

canvas.fillRect( 25, 25, 255, 205 );

canvas.setStroke(10);

canvas.setColor( "pink" );

canvas.drawRect( 25, 25, 255, 205 );

canvas.drawImage( imageName, xPosition, yPosition, width, height );

canvas.paint();

Exercise 1.6 My thoughts

I felt that this task was very easy. Most of the help that I received was the from the instructions panel. Most of the commands which I used in this exercise I received from the instructions panel. It wasn’t a difficult task just a lot of the same thing which was creating squares or filling squares.

Exercise 2.1 – Getting input from the user.

Exercise 2.1 – initial Code

// Ex2-1: Based on Ex1-2, Graphical Welcome

// All graphics programs require a canvas on which to draw

var canvas;

canvas = openGraphics();

// comment

var phrase1;

phrase1 = "Hello, ";

// comment

var phrase2;

phrase2 = ",<br />welcome to the Programming module.";

// comment

var name;

name = prompt( "Please enter your name:" );

// comment

var message;

message = phrase1 + name + phrase2;

// comment

canvas.setFont( "comic sans ms", "15px", Font.BOLD );

// comment

canvas.setColor( "darkred" );

// comment

canvas.drawString( message, 30, 50 );

// All graphics programs must instruct the browser to

// carry out the commands and render the output

canvas.paint();

Exercise 2.1 – Task 1

In this exercise I had to write code so that the user has input and is able to change the text to his preferences.

My first task was to annotate the entire program; this task will help get an idea about what is going on in each piece of code.

// Ex2-1: Based on Ex1-2, Graphical Welcome

// All graphics programs require a canvas on which to draw

var canvas;

canvas = openGraphics();

// This phrase says hello

var phrase1;

phrase1 = "Hello, ";

// This phrase says welcome to the Programming module

var phrase2;

phrase2 = ",<br />welcome to the Programming module.";

// This is promp which ask the user for his name

var name;

name = prompt( "Please enter your name:" );

// This is a compilation of all the phrases and inputs into one long message

var message;

message = phrase1 + name + phrase2;

// This determines the font of the text

canvas.setFont( "comic sans ms", "15px", Font.BOLD );

// This determines the color of the text

canvas.setColor( "darkred" );

// This determines the location of the text

canvas.drawString( message, 30, 50 );

// All graphics programs must instruct the browser to

// carry out the commands and render the output

canvas.paint();

Exercise 2.1 – Adding Flexibility

Exercise 2.1 task 1

In this task I had to add three new identifiers. These identifiers I picked were

// Prompts the user to enter a font style,.

var fontname;

fontname = prompt( "Please enter a font name:" );

// Prompts the user to enter the size of the font.

var fontsize;

fontsize = prompt( "Please enter a font size:" );

// Prompts the user to enter a colour for the text.

var fontcolor;

fontcolor = prompt( "Please enter a colour:" );

As we can see from these this code t each of the identifiers have been identified.

Exercise 2.1 task 2

My second task is add suitable prompts in which the user will able to change the font style size and colour

// Prompts the user to enter a font style,.

var fontname;

fontname = prompt( "Please enter a font name:" );

// Prompts the user to enter the size of the font.

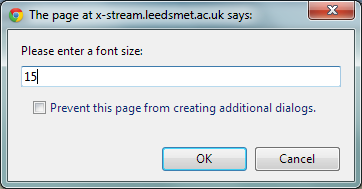
var fontsize;

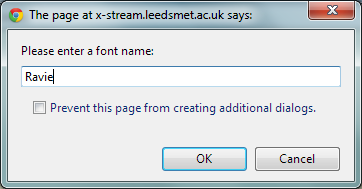
fontsize = prompt( "Please enter a font size:" );

// Prompts the user to enter a colour for the text.

var fontcolor;

fontcolor = prompt( "Please enter a colour:" );



As we can see from these three screenshot the prompt code have worked correctly and we have resulted in these three outboxes been presented to me.



As we can see from this screen shot on the right the prompts have worked correctly and I have been able to alter the way in which the text is displayed.

Exercise 2.1 complete code

// Ex2-1: Based on Ex1-2, Graphical Welcome

// All graphics programs require a canvas on which to draw

var canvas;

canvas = openGraphics();

// This phrase says hello

var phrase1;

phrase1 = "Hello, ";

// This phrase says welcome to the Programming module

var phrase2;

phrase2 = ",<br />welcome to the Programming module.";

// This is promp which ask the user for his name

var name;

name = prompt( "Please enter your name:" );

// Prompts the user to enter a font style,.

var fontname;

fontname = prompt( "Please enter a font name:" );

// Prompts the user to enter the size of the font.

var fontsize;

fontsize = prompt( "Please enter a font size:" );

// Prompts the user to enter a colour for the text.

var fontcolor;

fontcolor = prompt( "Please enter a colour:" );

// This is a compilation of all the phrases and inputs into one long message

var message;

message = phrase1 + name + phrase2;

// This determines the font of the text

canvas.setFont( fontname,fontsize, Font.BOLD );

// This determines the color of the text

canvas.setColor( fontcolor );

// This determines the location of the text

canvas.drawString( message, 30, 50 );

// All graphics programs must instruct the browser to

// carry out the commands and render the output

canvas.paint();

Exercise 2.1 My thoughts

I felt that this was a very fun exercise to do as I could see the effects of what I was inputting into the prompts. I felt that the commands that I had to input were very straight forward and I rarely faced any troubling parts in this exercise.

Exercise 2.2 – Numerical Input

Exercise 2.2 – Initial Code

// Ex2-2 : based on Ex1-4

var canvas;

canvas = openGraphics();

var value;

value = prompt( "Enter size of the square" );

var size;

size = parseInt( value, 10 );

canvas.drawRect( 10, 10, size, size );

// drawRect, short for drawRectangle, requires four values:

// drawRect( x, y, width, height );

// x and y are the coordinates of the top left corner

// width and height are what they say!

// Setting width and height to the same value draws a square

canvas.drawString( "Square of size:", 10, 250 );

canvas.drawString( size, 120, 250 );

canvas.paint();

Exercise 2.2 – Task 1

In this 1st task I had to modify the program to accept two values: width and height.

var height;

height = prompt( "Enter the height of the square" );

var width;

width = prompt( "Enter the width of the square" );

// "parseInt" changes the text in the prompt to a number format, this is important when it comes to working out values for the size.

height = parseInt( height, 10 );

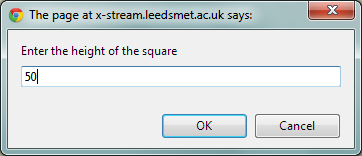
width = parseInt( width, 10 );

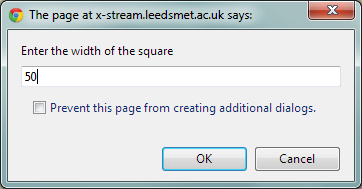
Exercise 2.2 – Task 2

My second task was to replace the two references to size on the drawRect function with width and height variables.

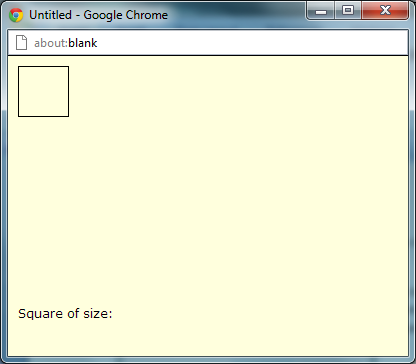
canvas.drawRect( 10, 10, width, height);

The result of this would mean that the rectangle will be draw depending on what values I put into the prompts.

We can see this prompt has worked correctly and I have input a value of 50 for the height of the square.



I can see the second prompt has also worked correctly and I have input a value of 50 for the width of the square.

From this screenshot we can finally see that both prompts have worked together and we have been displayed with a perfect 50 by 50 square.

Exercise 2.2 – Task 3

The third task in this exercise was to modify the output at the bottom of the window so that the width and height are output with suitable messages.

The way in which I did this was through a simple drawstring code.

The exact code that I used was

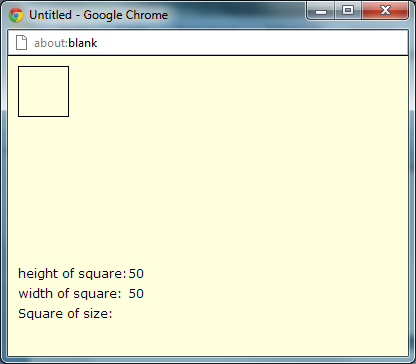
canvas.drawString( "height of square:", 10, 210 );

canvas.drawString( height, 120, 210 );

canvas.drawString( "width of square:", 10, 230 );

canvas.drawString( width, 120, 230 );

canvas.drawString( "Square of size:", 10, 250 );

As we can see from this screenshot the modification of the code has worked and two additional titles have been displayed.

These additional titles display the height of the square and the width of the square.

Exercise 2.2 – Task 4

In this fourth task I had to calculate the area of the square and output the value at the bottom of the output box.  
The way in which I did this was by adding an additional variable, the variable being.

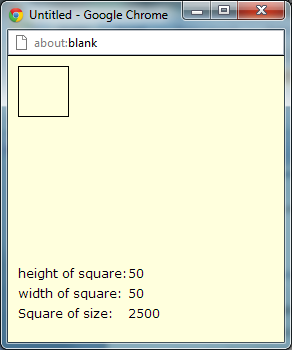
var area;

area = width \* height;

Also I had to use the drawstring code so that the value which was calculated could be attached to the canvas.

canvas.drawString( "Square of size:", 10, 250 );

canvas.drawString( area, 120, 250 );



This screenshot is the end result of both of the code. We can see that the correct “Size of square” has been calculated as the code States that width \* height = area and 50 \* 50 = 2500.

Exercise 2.2 – Going round again

In this task I had draw a circle and calculate the area of that circle

This is the code which I used to calculate the size and area of the square. The code has been taken from exercise 1.4 in parts.

var canvas;

canvas = openGraphics();

var radius;

radius = prompt( "Please Enter a radius of the circle:" );

var x;

x = prompt( "Please Enter the x-coordinate of the center of the circle:" );

var y;

y = prompt( "Please Enter the y-coordinate of the center of the circle:" );

var area;

area = (radius \* radius) \* Math.PI;

radius = parseInt( radius, 10 );

canvas.drawEllipse( (x - radius), (y - radius) , (radius\*2) , (radius\*2) );

canvas.drawString( "X-Coordinate:", 10, 210 );

canvas.drawString( x, 180, 210 );

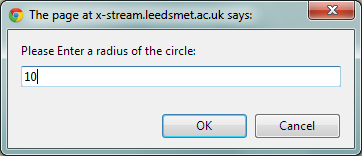
canvas.drawString( "Y-Coordinate:", 10, 230 );

canvas.drawString( y, 180, 230 );

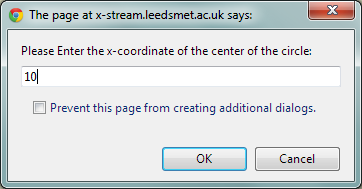
canvas.drawString( "area of circle:", 10, 250 );

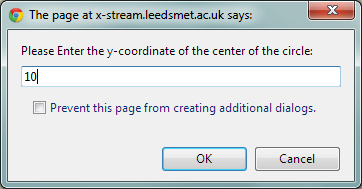
canvas.drawString( area, 120, 250 );

canvas.paint();

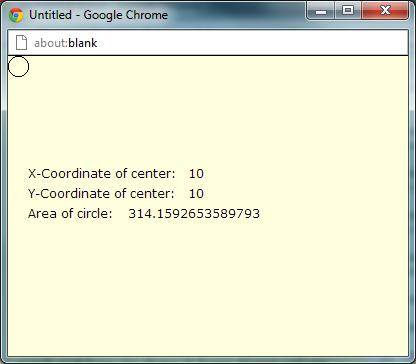
As a result of this code I was left with a fully functioning programme.

We can see that the prompt for the radius of the circle has worked correctly and I have input a value of 10.

We can see that the prompt for the x coordination of the circle has also worked and I have also input a value of 10 for that as well.



We can also see that the third prompt which was the y coordinate of the circle has also worked and I have yet again input another value of 10.



I am finally presented with a output box which is an accumulation of all the information which I have put into the prompts. We can see that all the information I have input into the prompts is being displayed and the

area = (radius \* radius) \* Math.PI;

radius = parseInt( radius, 10 );

has worked correctly and has calculated the “area of the circle.”

Exercise 2.3 – Personal Details

In this exercise I had to build a program that will ask the user for his/her Name, Mobile Number, Email address, Day/month/year of their birth. The DOB will be presented in three different values such as day, month and year. The program will then present this information in a way of my choosing. The way it will be presented depends on the font size type and colour of my choice. I will alos implement a function which will allow display the approximate age of the user in years. For example if a person was born in 1993 there approximate age would be 19 years old.

var canvas;

canvas = openGraphics();

var name;

name = prompt("Please enter your name:");

var mobilenumber;

mobilenumber = prompt("Please enter your mobile number:");

var emailaddress;

emailaddress = prompt("Please enter your email address:");

var day;

day = prompt("Please enter the day of your birthday:");

var month;

month = prompt("Please enter the month of your birthday (in number format):");

var year;

year = prompt("Please enter the year of your birthday");

var age;

age = (2012 - year);

var dob;

dob = day + month -1 + year;

canvas.setFont( "cambria", "30px", Font.ITALIC );

canvas.setColor( "blue" );

canvas.drawString( "Name:", 20, 40 );

canvas.drawString( name, 110, 40 );

canvas.drawString( "Mobile Number:", 20, 70 );

canvas.drawString( mobilenumber, 230, 70 );

canvas.drawString( "Email:", 20, 100 );

canvas.drawString( emailaddress, 105, 100 );

canvas.drawString( "Date of Birth:", 20, 130 );

canvas.drawString( dob, 195, 130 );

canvas.drawString( "Your approximate age is:", 20, 160 );

canvas.drawString( age, 340, 160 );

canvas.paint();

 As we can see from this screenshot the code which I input into the program has worked correctly. I have the name, the mobile number, the email address, the date of the birth and the approximate age all displayed.

The Approximate age was the result of this code.

var dob;

dob = day + month -1 + year

I can defiantly say that the code I have input has worked.

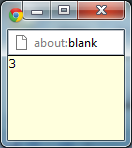
Exercise 2.3 – My Thoughts

I felt that this was a difficult exercise as it required many variables which ultimately meant that a lot of the information had to remain the same. This was a huge problem for me as I sometimes capitalised words and as a result a huge segment of my time was spent shifting through loads of code and comparing them to previous ones. Another problem which I encountered during this exercise was working out the formulae to calculate the approximate age. This was due to the months being one less than what I had anticipated. After the afternoon lecture that day I discovered the reason why and I quickly made changes.

Exercise 2.4 String operations

aString.length

var name = "Anj";

canvas.drawString( name.length ); 🡨 This code will display the number of characters in my name.

As you can see from this screenshot the code aString.length has presented me with the correct length of my name.

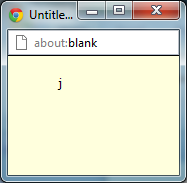
aString. substr(2)

var canvas;

canvas = openGraphics();

var name = "Anj";

canvas.drawString( name.substr(2), 50, 20 ); 🡨 This code subtracts characters out of my name depending on the value I put into the brackets..

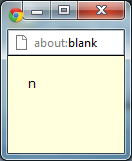
canvas.paint();

As we can see from this screenshot on the left, I have subtracted 2 of the characters in my name leaving me with the j at the end of my name.

aString.charAt(1)

var name = "anj";

canvas.drawString( name.charAt(1), 20, 20 ); 🡨 this code will takes a character and put it in a output box depending on the value in the brackets



As you can see from this screenshot the code has picked the second character from my name as I assigned the programme to do so. This code has worked correctly.

aString.substr(1) and astring .charAt(1)

var canvas;

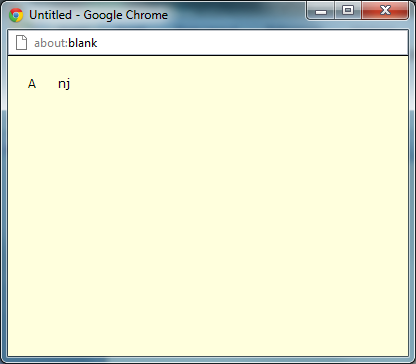
canvas = openGraphics();

var name = "Anj";

canvas.drawString( name.charAt(1), 20, 20 ); 🡨 This code changes the location of the character depending on the value

canvas.drawString( name.substr(1), 50, 20 );

canvas.paint();



As you can see from this screenshot there is a space in between the character in my name.

Exercise 2.4 My Thoughts

I felt that this was a very simple exercise the only tedious part of this exercise was the abundance of reading that I had to do. This was not a really bad exercise it just required a huge amount of ready to use the commands to complete the tasks.

Exercise 2.5 – Date operations

In this exercise I had to

Exercise 2.5 Initial Code

// Ex2-5

var canvas;

canvas = openGraphics();

var value;

var aDate;

aDate = new Date();

canvas.drawString( "The date is: " + aDate, 10, 20 );

value = aDate.toLocaleString();

canvas.drawString( "toLocalString: " + value, 10, 40 );

// value = aDate.get .......

// canvas.drawString( "getYear: " .......

canvas.paint();

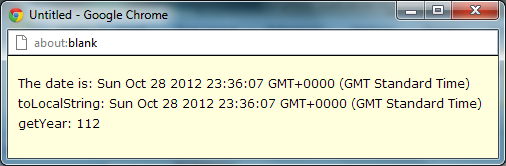
Exercise 2.5 Task1 – getYear();

In this task I had to find out the year through the code

value = aDate.getYear();

canvas.drawString( "getYear: " + value, 10, 60);

The result of this code can be seen on this output box screenshot.

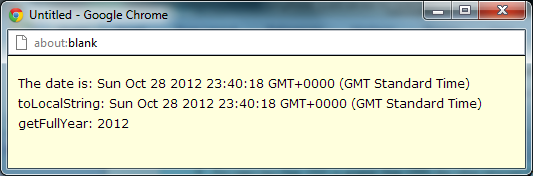
As you can see from this screenshot the getYear code has displayed the year as 112. This is because we are in the year 2012. Years abouve 200 are represented in numbers 100 or greater.

This is the same for years between 1900 and 1999. getYear for these years is between 0 and 99 so if the year was 1912 the number would be 12.

Exercise 2.5 Task2 – GetFullYear();

value = aDate.getFullYear();

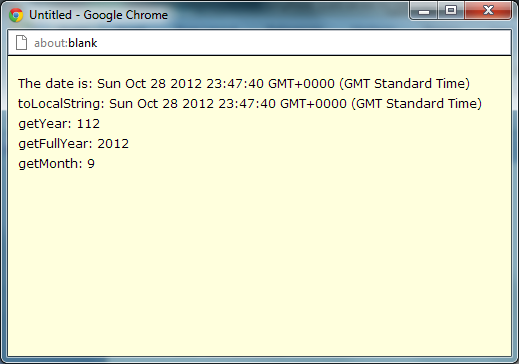
canvas.drawString( "getFullYear: " + value, 10, 80);

As we can see from this screenshot the number is now determined by a full number which is the exact same number as our year.

This is due to getFullYear is an whole number or an absolute number. Whereas getYear only returns numbers in the hundreds or below.

Exercise 2.5 task 3 – GetMonth();

value = aDate.getMonth();

canvas.drawString( "getMonth: " + value, 10, 100);

As we can see from this screen we now have a getMonth value. This shows that the code I have input has worked correctly.

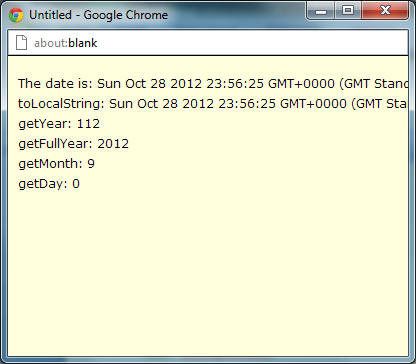
As for GetMonth. January starts at 0 and every other month is one less than it should be. We can see that the month is 9 when in fact it should be 10 for October.

Exercise 2.5 task 4 – GetDay ();

value = aDate.getDay();

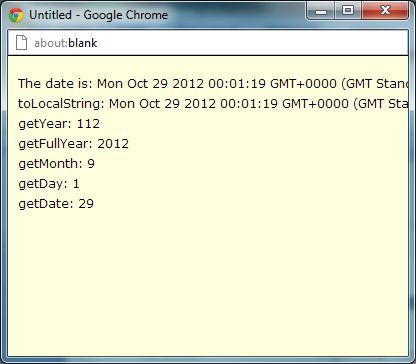
canvas.drawString( "getDay: " + value, 10, 120);

The value for get day is determined by a integer starting at 0 for Sunday, 1 Monday etc.

We can see from this screenshot that the day has also worked correctly. We can see that the value in the getDay is set at 0 so we can say that the date must be Sunday.

Exercise 2.5 task 5 – GetDate ();

The value output for the getDate is a number between 1 and 31. However not all months has 31 days in then so there is a chance that the month flips over to the next one.

We can see that the number on the getDate is 29 which ultimately means that it is the 29th day of that month.

Exercise 2.5 – Finished Code

// Ex2-5

var canvas;

canvas = openGraphics();

var value;

var aDate;

aDate = new Date();

canvas.drawString( "The date is: " + aDate, 10, 20 );

value = aDate.toLocaleString();

canvas.drawString( "toLocalString: " + value, 10, 40 );

value = aDate.getYear();

canvas.drawString( "getYear: " + value, 10 ,60);

value = aDate.getFullYear();

canvas.drawString( "getFullYear: " + value, 10, 80);

value = aDate.getMonth();

canvas.drawString( "getMonth: " + value, 10, 100);

value = aDate.getDay();

canvas.drawString( "getDay: " + value, 10, 120);

value = aDate.getDate();

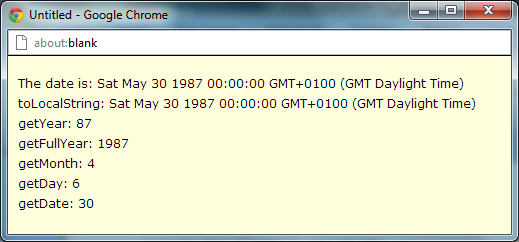
canvas.drawString( "getDate: " + value, 10, 140);

canvas.paint();

Exercise 2.5 task 6 – aDate = new Date(1987, 4, 30);

After we alter the date in the code like the exercise instructs us to. We can independently change todays date, as a result the rest of the code changes aswell.

This code is necessary for this to work

aDate = new Date(1987, 4, 30);

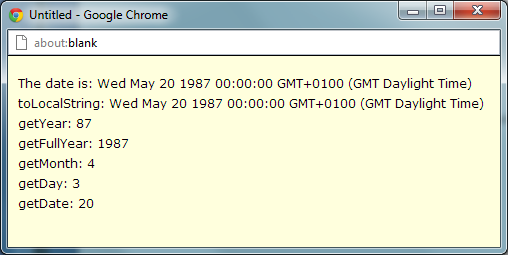
As we can see from this screenshot the code which I just input has worked and the result is this screenshot.

Exercise 2.5 task 7 - setDate();

In this task I had to independently set the date. This was the code that I used.

aDate = new Date(1987, 4, 30);

aDate.setDate(20);

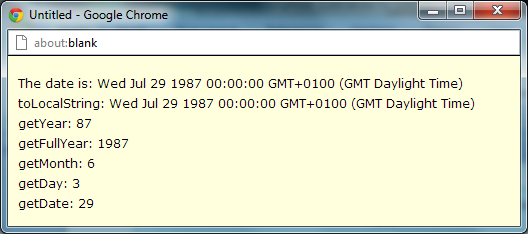
As we can see from this screenshot the date has changed due to how I have set the value in the setDate variable.

Exercise 2.5 task 8 - setMonth();

In this task I had to independently change the month. This is the code I used.

aDate = new Date(1987, 6, 29);

aDate.setMonth(6);

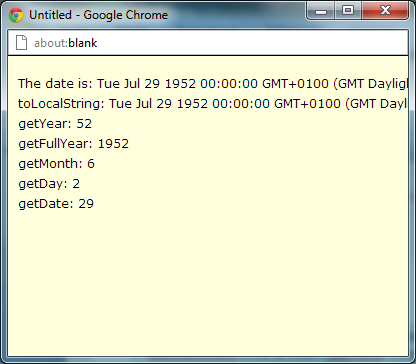
As you can see from this screenshot you can see that the month has changed to what I set it to and the date as moved according to that value.

Exercise 2.5 task 9 - setYear();

In this task I had to independently change the year. This is the code I used.

aDate = new Date(1987, 6, 29);

aDate.setYear(1952);

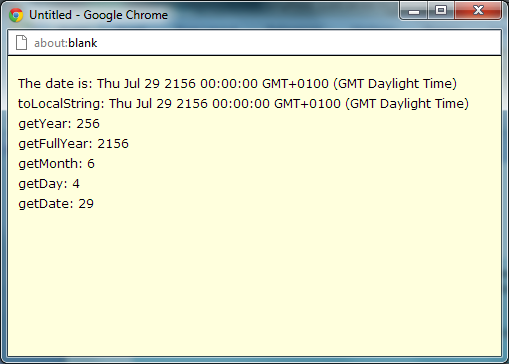
As you can see from this screenshot the year had changed, I have set the year to 1952 and the day and also changed according to the value I input.

Exercise 2.5 task 10 - setFullYear();

In this task I had to independently change the full year. This is the code I used.

aDate = new Date(1987, 6, 29);

aDate.setFullYear(2156);

As you can from this screenshot the full year has changed. I have set the full year to 2156 and as a result the remaining variable have also changed to match.

Exercise 2.5 – complete code

// Ex2-5

var canvas;

canvas = openGraphics();

var value;

var aDate;

aDate = new Date(1987, 6, 29);

aDate.setFullYear(2156);

canvas.drawString( "The date is: " + aDate, 10, 20 );

value = aDate.toLocaleString();

canvas.drawString( "toLocalString: " + value, 10, 40 );

value = aDate.getYear();

canvas.drawString( "getYear: " + value, 10 ,60);

value = aDate.getFullYear();

canvas.drawString( "getFullYear: " + value, 10, 80);

value = aDate.getMonth();

canvas.drawString( "getMonth: " + value, 10, 100);

value = aDate.getDay();

canvas.drawString( "getDay: " + value, 10, 120);

value = aDate.getDate();

canvas.drawString( "getDate: " + value, 10, 140);

canvas.paint();

Exercise 2.5 – My thoughts

I felt that this was a very long and tedious exercise. As for the code it was relatively simple to use and work with but the recording the log each step was very jarring.

Exercise 2.6 How Old are you?

In this exercise I have to find out the age of the user.

Exercise 2.6 Initial Code

var canvas;

canvas = openGraphics();

var day;

var month;

var year;

var birthday;

// get user's birthday details

// create two dates

var age;

// calculate age in mS and convert to days and then years

canvas.paint();

Exercise 2.6- Task 1

My first task in this exercise was to add appropriate lines of code to the initial program that prompts the user for their day, month and year of birth.

The way in which I went about doing this task was by first by implementing my command prompts. The code which I was left with was

var day;

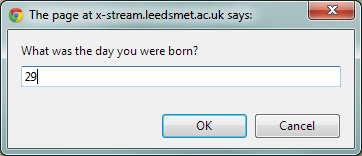
day = prompt("What was the day you were born?" );

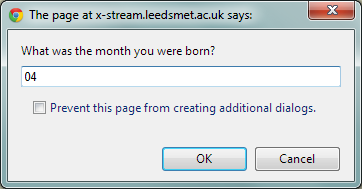
var month;

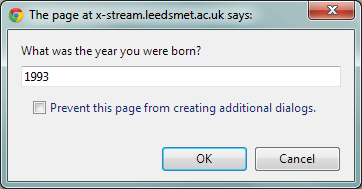
month = prompt("What was the month you were born?" );

var year;

year = prompt("What was the year you were born?" );

As you can see from this screenshot the prompt for the day of my birthday has worked and I have input 29 as the day of my birth

  
You can see that this screenshot for the month of my birthday has also worked and I have input 04 as the month of my birth.



As we can again see here the prompt for the year of my birth has again worked this means that all the prompting codes work correctly.

Exercise 2.6- Task 2

In this task was to create a Date from these values. The way in which I did this was utilising the variable know as “birthday”

var birthday;

birthday = new Date( year, month -1 ,day );

This will compile all the information I have placed in the input box into one complete day. Hopefully, my birthday.

Exercise 2.6- Task 3

My third task on this exercise was to make a suitable output message. I did this through the canvas.drawString codes and a combination of phrases and messages. I have previously used before.

The code I used to present and display this message was

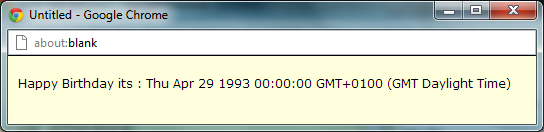
var phrase1;

phrase1 = "Happy Birthday its : " ;

var message;

message = phrase1 + birthday;

canvas.drawString( phrase1 + birthday, 10, 20 );

As we can see from this screenshot the code which I have input befor has worked correctly and I have displayed my birthday “29th april 1993” with a suitable message.

Exercise 2.6- Find the users age

Exercise 2.6- Task 1

I started this task off by doing what the exercise instructed me to do which was to create a new variable called today.

The way In which I did that was creating a new pieces of code. The exact code I use was

var value;

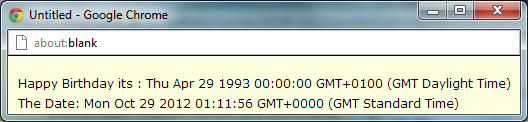
var today;

today = new Date();

canvas.drawString("The Date: " + today, 10, 40);

value = today.toLocaleString();

The code above shows the current date.

As you can see from this screenshot the code I have input has worked and I am now being displayed the current date.

Exercise 2.6 – Task 2

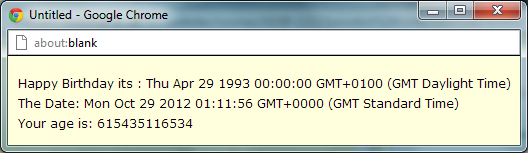
The next task was very simple; it involved creating a variable called age and the finding out the age of the user through a simple subtraction of today’s date away from my birthday.

This is the code that I used to find my age.

var age;

age = ( today - birthday );

canvas.drawString(" Your age is: " + age, 10 , 60);

As you can see from this screenshot the code above has worked and I am now being displayed my age is seconds.

Exercise 2.6 – Task 3

I outputted the value as you can see from the previous screenshot.

Exercise 2.6 – Task 4

I now need to divide that number so I am ultimately left with the number of years I have been living for.

The way in which I did this was through huge amounts of mathematical code. The exact code that I used was

var seconds = age/1000;

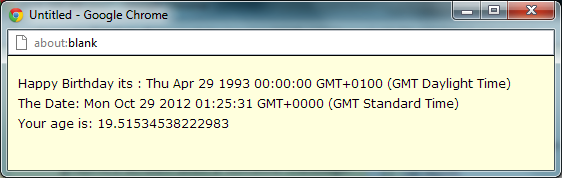
var minutes = seconds/60;

var hours = minutes/60;

var days = hours/24;

var years = days/365;

age = years;



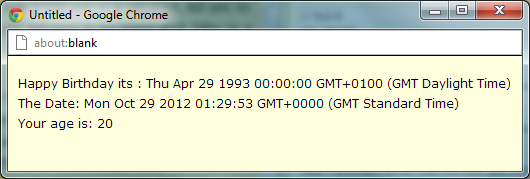
As you can see from this screenshot the code above have resulted in the age being calculated at 19 year old which is correct. So as a result I can saftly say that this code works.

Exercise 2.6 – Task 5

My task now was to round the value that I divided it to, to a whole number. The way in which I did that was using the ode

age = Math.round(age);

Hopefully this code will round my age to the near number.

As you can see from this screenshot the age = Math.round(age); code has round my age to the nearest whole number and in my situation it has rounded my age to 20 which is correct as I am closer to 20 then I am 19.

Exercise 2.6 – Complete code

// Ex2-6

var canvas;

canvas = openGraphics();

var day;

day = prompt("What was the day you were born?" );

var month;

month = prompt("What was the month you were born?" );

var year;

year = prompt("What was the year you were born?" );

var birthday;

birthday = new Date( year, month -1 ,day );

var phrase1;

phrase1 = "Happy Birthday its : " ;

var message;

message = phrase1 + birthday;

canvas.drawString( phrase1 + birthday, 10, 20 );

var value;

var today;

today = new Date();

canvas.drawString("The Date: " + today, 10, 40);

value = today.toLocaleString();

var age;

age = ( today - birthday );

// calculate age in mS and convert to days and then years

var seconds = age/1000;

var minutes = seconds/60;

var hours = minutes/60;

var days = hours/24;

var years = days/365;

age = years;

age = Math.round(age);

canvas.drawString(" Your age is: " + age, 10 , 60);

canvas.paint();

Exercise 2.7 Events timer

Exercise 2.7 initial code.

// Ex2-7: Event Timer

// This program calculates the difference in time

// between two events: the user clearing alerts

// The difference is a measure of the time it

// Takes to make two keypresses or mouse clicks

var canvas;

canvas = openGraphics();

// Put a Title on the output screen

// uncomment the next line to see the title

// canvas.paint();

// create a Date object and store the current time

alert( "Press \"OK\" to start the timer." );

// create another Date object and store the time now

alert( "Press \"OK\" to stop the timer." );

// calculate the difference between the two times

// output a suitable message to the user

canvas.paint();

Exercise 2.7 – Task 1

In this task I had to create the title for my event timer. The way in which I did this was through code such as setFont, setColour.

The exact code that I used was

canvas.setColor( "crimson" );

canvas.setFont( "chiller", "50px", Font.BOLD );

canvas.drawString("Event Timer" , 10, 10); As you can see the code works correctly

The result of this code was this screenshot

Exercise 2.7 – Task 2

The second task was to create two objects to store the times when the button “OK” is clicked.

The way in which I went about doing this using the variable date1 and date2 which record the current time after each click. The exact code looks like this

var date1 = new Date();

alert( "Press \"OK\" to start the timer." );

var date2 = new Date();

alert( "Press \"OK\" to stop the timer." );

Exercise 2.7 task 3

The third task was to calculate the difference between the two times and display them in a suitable message to the user.

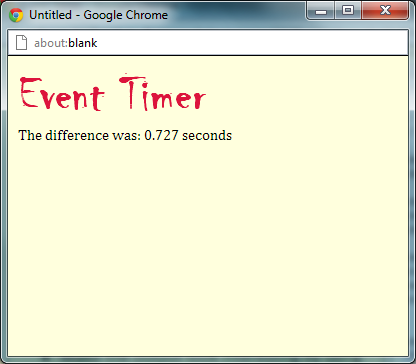
var difference;

difference= date2 - date1;

var seconds = difference/1000;

canvas.setColor( "black" );

canvas.setFont( "cambria", "15px" );

canvas.drawString( "The difference was: " + seconds + " seconds", 10, 70 );

As you can see from the screen shot the code above has worked and the message which I wrote out has been display. This is a suitable message to have when determining the difference between two points in time.

Exercise 2.7 Finished Code

// Ex2-7: Event Timer

// This program calculates the difference in time

// between two events: the user clearing alerts

// The difference is a measure of the time it

// Takes to make two keypresses or mouse clicks

var canvas;

canvas = openGraphics();

canvas.setColor( "crimson" );

canvas.setFont( "chiller", "50px", Font.BOLD );

canvas.drawString("Event Timer" , 10, 10);

var date1 = new Date();

alert( "Press \"OK\" to start the timer." );

var date2 = new Date();

alert( "Press \"OK\" to stop the timer." );

var difference;

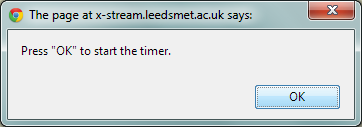
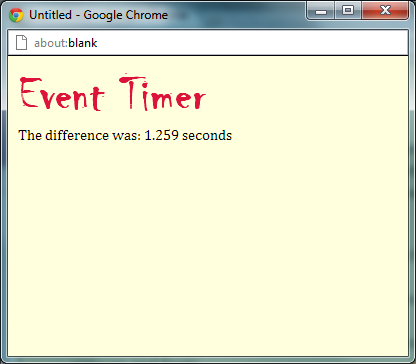
difference= date2 - date1;

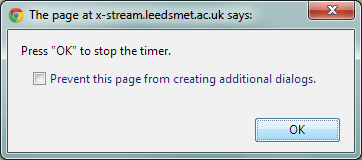
var seconds = difference/1000;

canvas.setColor( "black" );

canvas.setFont( "cambria", "15px" );

canvas.drawString( "The difference was: " + seconds + " seconds", 10, 70 );

canvas.paint();



Exercise 2.7 My thoughts.

I very much enjoyed this exercise I like the creativity side of it. I felt that coding complication for this exercise was also very easy and all it really required was a good think through. I didn’t really encounter any problems when performing the tasks.

Exercise 3.1 Grading Work

In this exercise I had to grade student work through classifications depending on the percentage they gained in an exam.

Exercise 3.1 – Intro code

// Ex: 3-1 Grading Work

var mark;

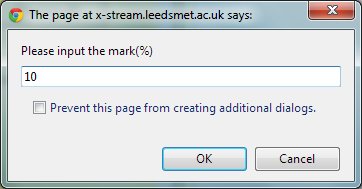
var grade;

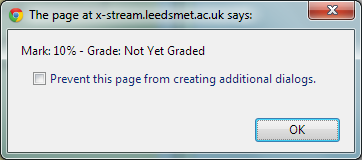
grade = "Not Yet Graded";

mark = prompt( "Please input the mark(%)" );

mark = parseInt( mark, 10 ); // see comment

alert( "Mark: " + mark + "% - Grade: " + grade );

As you can see the initial prompt is functioning correctly and I have input a value of 10%



You can also see that the secondary output box is functioning correctly and the 10% I input early has not worked as I haven’t put any function in.  
Exercise 3.1 – Task 1

My first task was to load up the initial code an run it which has already been done.

Exercise 3.1 – Task 2

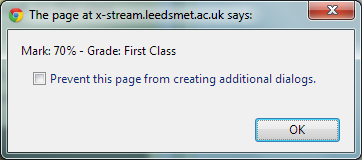
My second task was to add the following lines to the program so that marks of 70% and above are classified as first class:

if( mark >= 70 )

{

grade = "First Class";

}

The code above means that any percentages which are greater than or equal to 70% the classification of that mark is “First Class”.

And we can see that it has worked as we are presented with the correct output and classification of mark.

Exercise 3.1 – Task 3

The next task involved me doing the same for mark which were greater than or euqla to 60. This is the classification under 1st class.

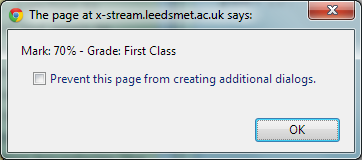
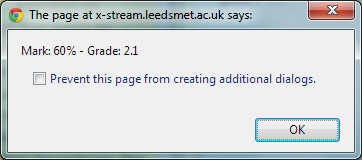
The code which I used to make this classification for marks of 60 or higher was this.

else if ( mark >= 60 )

{

grade = "2.1";

}



As we can see from these two screenshot both “ First Class” classification and “ 2.1” classification are both function correctly and displaying their appropriate classifications.

Exercise 3.1 – Task 4

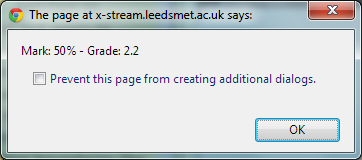
My next Task was to input more code so that the next grade “2.2 “is taken into account and that it works too. The code was relatively simple as it mimics the code from previous classifications.

else if ( mark >= 50 )

{

grade = "2.2";

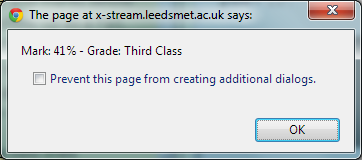
}

We can now see that this classification has also worked. As the screen shot displays both the percentage mark and the classification correctly.

Exercise 3.1 – Task 5

My last task on this section was to complete the classifications for both “third class” and “fail” classifications.

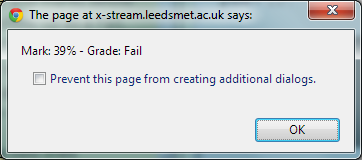
The code which I input to implement these changes were

else if ( mark >= 40 )

{

As we can see from these two screenshots both of the marks within the screenshot is corresponding to the correct classification. This means that the code I have input is correct also

grade = "Third Class";

}

else if ( mark < 40 )

{

grade = "Fail";

}

Exercise 3.1 – Extending the programme

You might have observed that values of over 100% (which are not really valid marks) will be awarded a First Class grade and marks below zero (also not really valid) show up as a Fail.

You need to put all of the if(..) else if(..) code into the body of another if statement that will only execute if the mark is greater then, or equal to 0 AND less than or equal to 100.

Exercise 3.1 – Extending the programme – Task 1

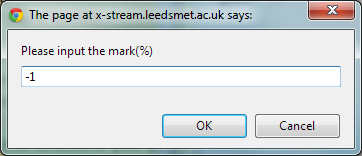
I had to add the two line of code that the exercise gave to me and put them in my programme.

These two lines of code which we were told to put into our programme are.

if(mark <0 || mark >100 )

{

alert("Invalid mark,outside 0-100");

}

If I was to put an invalid make it. For example if I was to put -1 in the input back this would be an invalid mark as it isn’t within the range which I just programmed it for.



So as a result I am presented with an error box like I should be. This error message tells me that the code has worked correctly.

Exercise 3.1 – Complete Code

// Ex: 3-1 Grading Work

var mark;

var grade;

mark = prompt( "Please input the mark(%)" );

mark = parseInt( mark, 10 ); // see comment

if( mark >= 70 )

{

grade = "First Class";

}

else if ( mark >= 60 )

{

grade = "2.1";

}

else if ( mark >= 50 )

{

grade = "2.2";

}

else if ( mark >= 40 )

{

grade = "Third Class";

}

else if ( mark < 40 )

{

grade = "Fail";

}

if(mark <0 || mark >100 )

{

alert("Invalid mark,outside 0-100");

}

alert( "Mark: " + mark + "% - Grade: " + grade

Exercise 3.1 – My thoughts

This exercise was quite challenging. With enough time I worked it out and completed it on my own. The only problem which I encountered in the exercise was the location of the If(true) statement. I was putting it before any of the input were put in as a result the complication never ran through it and the if true statement was never taken into account.

Exercise 3.2 – Date validation

In Exercise 2-3, you created a program that asked a user for their date of birth and did some manipulation to show, for example, how many days they had been alive.

Find that program in your log and copy it into the code pane. As it stands, the program does not check to see if the user has entered a valid date. The JavaScript Date object will do its best with duff data. Entering values for 32-3-06 will probably end up with 1st April 1906.

The code I used for 2.6 was

// Ex2-6

var canvas;

canvas = openGraphics();

var day;

day = prompt("What was the day you were born?" );

var month;

month = prompt("What was the month you were born?" );

var year;

year = prompt("What was the year you were born?" );

var birthday;

birthday = new Date( year, month -1 ,day );

var phrase1;

phrase1 = "Happy Birthday its : " ;

var message;

message = phrase1 + birthday;

canvas.drawString( phrase1 + birthday, 10, 20 );

var value;

var today;

today = new Date();

canvas.drawString("The Date: " + today, 10, 40);

value = today.toLocaleString();

var age;

age = ( today - birthday );

// calculate age in mS and convert to days and then years

var seconds = age/1000;

var minutes = seconds/60;

var hours = minutes/60;

var days = hours/24;

var years = days/365;

age = years;

age = Math.round(age);

canvas.drawString(" Your age is: " + age, 10 , 60);

canvas.paint();

Exercise 3.2 – Task 1

I have copied the solution from 2.6 into this exercise.

Exercise 3.2 – Task 2

I have checked that the code still works and it does.

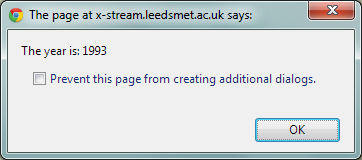
Exercise 3.2 – Task 3

I have input the code which was given to me in the instructions panel

The exact code was

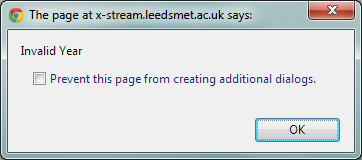
I was presented with this text box validating the year that the year I put in was in fact 1993.

I then re ran the program and made an incorrect year purposely to check to see it the validation worked on the opposite way round and I was presented with this output box state that it was an invalid year.

birthday.getFullYear();

if (year != birthday.getFullYear())

{

 alert("Invalid Year");

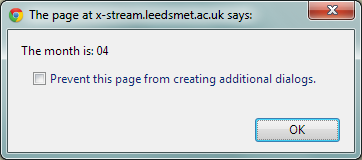
}

else

{

alert("The year is: " + year, 10, 120);

}

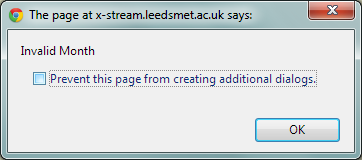
I also did the same with the month and the day birthday.getMonth();

As we can see from this screenshot the month that I have put in has been validated by this output box this means that the code I input was correct.

I also checked to see if the invalid month alert appeared, it did.

if(month != birthday.getMonth()+1)

{

 alert("Invalid Month");

}

else

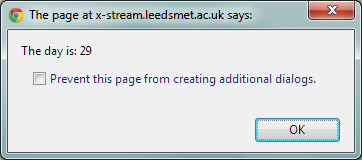
{

alert("The month is: " + month, 10, 120);

}

I finally did the same with the date.

The code I input to put validation on the date was.

birthday.getDate();

if(day!= birthday.getDate())

As we can see from this screenshot the code I input for the date validation has worked successfully and validated my date of birth.

{

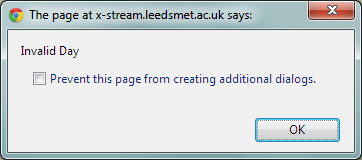
alert("Invalid Day");

}

else

{

alert("The day is: " + day, 10, 120);

}

As we can see from this screenshot the incorrect day I put in purposely has resulted in an alert being initialised. This means that the code I have input is correct.

Exercise 3.2 – Complete code

// Ex3-2

var canvas;

canvas = openGraphics();

var day;

day = prompt("What was the day you were born?" );

var month;

month = prompt("What was the month you were born?" );

var year;

year = prompt("What was the year you were born?" );

var birthday;

birthday = new Date( year, month -1 ,day );

var phrase1;

phrase1 = "Happy Birthday its : " ;

var message;

message = phrase1 + birthday;

canvas.drawString( phrase1 + birthday, 10, 20 );

var value;

var today;

today = new Date();

canvas.drawString("The Date: " + today, 10, 40);

value = today.toLocaleString();

var age;

age = ( today - birthday );

// calculate age in mS and convert to days and then years

var seconds = age/1000;

var minutes = seconds/60;

var hours = minutes/60;

var days = hours/24;

var years = days/365;

age = years;

age = Math.round(age);

canvas.drawString(" Your age is: " + age, 10 , 60);

birthday.getFullYear();

if (year != birthday.getFullYear())

{

alert("Invalid Year");

}

else

{

alert("The year is: " + year, 10, 120);

}

birthday.getMonth();

if(month != birthday.getMonth()+1)

{

alert("Invalid Month");

}

else

{

alert("The month is: " + month, 10, 120);

}

birthday.getDate();

if(day!= birthday.getDate())

{

alert("Invalid Day");

}

else

{

alert("The day is: " + day, 10, 120);

}

canvas.paint();

Exercise 3.2 My thoughts

I thought that this was a very long exercise. However, I am happy to know that we were able to use the code from exercise 2.6. This meant that the amount of code that I actually had to put in myself was greatly reduced the only bit of code that I had to put in was given to use. I didn’t find any problems when progressing through this task.

Exercise 3.3 Coloured Event time

In this task we had to change the colour of the text depending on how long it took us to click both of the “OK “buttons.

Exercise 3.3 initial code

// Ex3-3: Event Timer

// This program calculates the difference in time

// between two events: the user clearing alerts

// The difference is a measure of the time it

// Takes to make two keypresses or mouse clicks

var canvas;

canvas = openGraphics();

canvas.setColor( "crimson" );

canvas.setFont( "chiller", "50px", Font.BOLD );

canvas.drawString("Event Timer" , 10, 10);

var date1 = new Date();

alert( "Press \"OK\" to start the timer." );

var date2 = new Date();

alert( "Press \"OK\" to stop the timer." );

var difference;

difference= date2 - date1;

var seconds = difference/1000;

canvas.setColor( "black" );

canvas.setFont( "cambria", "15px" );

canvas.drawString( "The difference was: " + seconds + " seconds", 10, 70 );

canvas.paint();

This is the initial code from exercise 2.7; the initial code of this exercise is the same.

Exercise 3.3 – Task 1

I found the solution and I have it within the program.

Exercise 3.3 – Task 2

I then added the appropriate line of code into the solution

The full code then looked like this

if ( difference < 150 )

{

canvas.setColor( "green" );

}

else if ( difference < 350 )

{

canvas.setColor( "orange" );

}

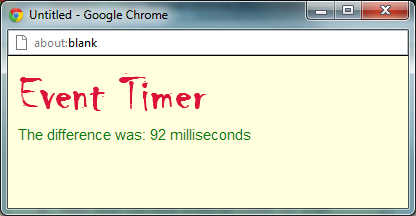
else if ( difference > 350 )

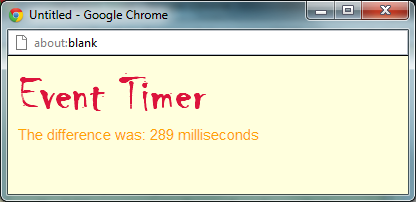
{

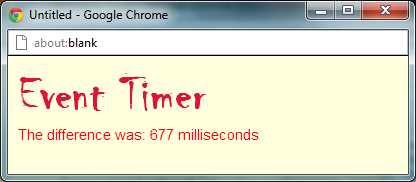
canvas.setColor( "red" );

}

The result of these additional pieces of code has resulted in three different output colours occurring.

The green text means that the difference was less than 150 milliseconds

The orange text means that the difference was less than 350 milliseconds



The red means that the difference was about 350 milliseconds

Exercise 3.3 – complete code

// Ex3-3: Event Timer

// This program calculates the difference in time

// between two events: the user clearing alerts

// The difference is a measure of the time it

// Takes to make two keypresses or mouse clicks

var canvas;

canvas = openGraphics();

canvas.setColor( "crimson" );

canvas.setFont( "chiller", "50px", Font.BOLD );

canvas.drawString("Event Timer" , 10, 10);

var date1 = new Date();

alert( "Press \"OK\" to start the timer." );

var date2 = new Date();

alert( "Press \"OK\" to stop the timer." );

var difference;

difference= date2 - date1;

if ( difference < 150 )

{

canvas.setColor( "green" );

}

else if ( difference < 350 )

{

canvas.setColor( "orange" );

}

else if ( difference > 350 )

{

canvas.setColor( "red" );

}

canvas.setFont( "arial", "15px" );

canvas.drawString( "The difference was: " + difference + " milliseconds", 10, 70 );

canvas.paint();

Exercise 3.3 – My thoughts.

I thought that this exercise was fairly easy as the code had already been completed for us as we completed it in a previous exercise. I also felt that the if formulas weren’t as hard as I had initially thought as we had again already done these in previous exercises and I had had some practise with them so I knew exactly how to work them. I didn’t find anything to strenuous in the exercise.

Exercise 3.4 – Truth Tables

In this exercise I had to complete the truth table using the code and matching it so I am left with the appropriate response.

Exercise 3.4 initial Code

// Ex: 3-4

var canvas;

canvas = openGraphics();

var operation;

operation = "AND";

var heading;

heading = "Truth Table for the " + operation + " operation.";

canvas.drawStringRect( heading.big(), 0, 5, 400, "center" );

var message;

message = "An entry in the table is the result of the operation ";

message = message + "A " + operation + " B";

canvas.drawStringRect( message, 0, 30, 400, "center" );

message = "i.e. result = a && b;";

canvas.drawStringRect( message, 0, 65, 400, "center" );

var x; // left edge of table

var y; // top of table

x = 50;

y = 100;

// draw table outline

canvas.setStroke( 3 );

canvas.drawLine( x, y, x+300, y );

canvas.drawLine( x, y, x, y+150 );

canvas.drawLine( x, y+50, x+300, y+50 );

canvas.drawLine( x, y+150, x+300, y+150 );

canvas.drawLine( x+100, y, x+100, y+150 );

canvas.drawLine( x+300, y, x+300, y+150 );

canvas.setStroke( 1 );

canvas.drawLine( x, y+100, x+300, y+100 );

canvas.drawLine( x+200, y, x+200, y+150 );

canvas.setStroke( Stroke.DOTTED );

canvas.drawLine( x, y, x + 100, y + 50 );

// Put in column and row labels

canvas.setFont( "lucida console", "20px", Font.BOLD );

canvas.setColor( "darkred" );

canvas.drawStringRect( "A", x+15, y+20, 50, "left" );

canvas.drawStringRect( "B", x+5, y+5, 80, "right" );

canvas.drawStringRect( "true", x, y+50+15, 100, "center" );

canvas.drawStringRect( "false", x, y+100+15, 100, "center" );

canvas.drawStringRect( "true", x+100, y+15, 100, "center" );

canvas.drawStringRect( "false", x+200, y+15, 100, "center" );

// Now to put the results in the table

var result;

var a;

var b;

a = true;

b = true;

result = a && b;

canvas.drawStringRect( result, x+100, y+50+15, 100, "center" );

// COMPLETE THE TABLE

canvas.paint();

Exercise 3.4 – Task 1

I found the line relating to the instruction. These lines were

a = true;

b = true;

result = a && b;

canvas.drawStringRect( result, x+100, y+50+15, 100, "center" );

Exercise 3.4 – Task 2

I then replicated the code above three times and changed the values so that I could complete the table

a = true;

b =false;

result = a && b;

canvas.drawStringRect( result, x+200, y+50+15, 100, "center" );

a = false;

b = true;

result = a && b;

canvas.drawStringRect( result, x+100, y+100+15, 100, "center" );

a = false;

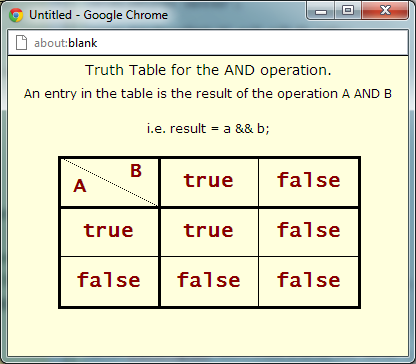
b = false;

result = a && b;

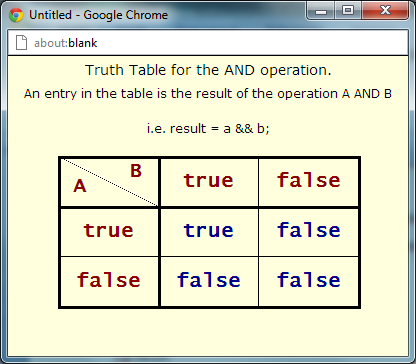
canvas.drawStringRect( result, x+200, y+100+15, 100, "center" );

Exercise 3.4 – Task 3

I checked to see if the code had worked correctly

As we can see from this screenshot the duplicated pieces of information which I altered have worked correctly and the location which I also alter are also correct.

Exercise 3.4 – Task 4

The next task was to change the colour of the output from dark red to dark blue

canvas.setColor( "darkblue" );

was the piece of code I needed to changed in order to change the colour.

as we can see from this screenshot the code has worked correctly and the output colour is now in dark blue instead of dark red.

Exercise 3.4 – The OR operation.

I had to change my programme so that I could complete the task.

a = true;

b = true;

result = a && b;

canvas.drawStringRect( result, x+100, y+50+15, 100, "center" );

a = false;

b =true;

result = a || b;

canvas.drawStringRect( result, x+200, y+50+15, 100, "center" );

a = true;

b = false;

result = a || b;

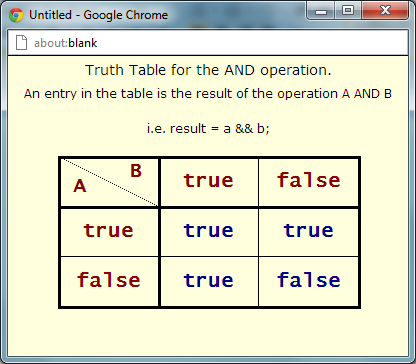
canvas.drawStringRect( result, x+100, y+100+15, 100, "center" );

a = false;

b = false;

result = a || b;

canvas.drawStringRect( result, x+200, y+100+15, 100, "center" );

As we can see from this screenshot there is only one false remaining this means that the code I have input was correct and the instruction panel states that there should only be one false panel remaining.

Exercise 3.4 – Final Code.

// Ex: 3-4

var canvas;

canvas = openGraphics();

var operation;

operation = "AND";

var heading;

heading = "Truth Table for the " + operation + " operation.";

canvas.drawStringRect( heading.big(), 0, 5, 400, "center" );

var message;

message = "An entry in the table is the result of the operation ";

message = message + "A " + operation + " B";

canvas.drawStringRect( message, 0, 30, 400, "center" );

message = "i.e. result = a && b;";

canvas.drawStringRect( message, 0, 65, 400, "center" );

var x; // left edge of table

var y; // top of table

x = 50;

y = 100;

// draw table outline

canvas.setStroke( 3 );

canvas.drawLine( x, y, x+300, y );

canvas.drawLine( x, y, x, y+150 );

canvas.drawLine( x, y+50, x+300, y+50 );

canvas.drawLine( x, y+150, x+300, y+150 );

canvas.drawLine( x+100, y, x+100, y+150 );

canvas.drawLine( x+300, y, x+300, y+150 );

canvas.setStroke( 1 );

canvas.drawLine( x, y+100, x+300, y+100 );

canvas.drawLine( x+200, y, x+200, y+150 );

canvas.setStroke( Stroke.DOTTED );

canvas.drawLine( x, y, x + 100, y + 50 );

// Put in column and row labels

canvas.setFont( "lucida console", "20px", Font.BOLD );

canvas.setColor( "darkred" );

canvas.drawStringRect( "A", x+15, y+20, 50, "left" );

canvas.drawStringRect( "B", x+5, y+5, 80, "right" );

canvas.drawStringRect( "true", x, y+50+15, 100, "center" );

canvas.drawStringRect( "false", x, y+100+15, 100, "center" );

canvas.drawStringRect( "true", x+100, y+15, 100, "center" );

canvas.drawStringRect( "false", x+200, y+15, 100, "center" );

// Now to put the results in the table

var result;

var a;

var b;

canvas.setColor( "darkblue" );

a = true;

b = true;

result = a && b;

canvas.drawStringRect( result, x+100, y+50+15, 100, "center" );

a = false;

b =true;

result = a || b;

canvas.drawStringRect( result, x+200, y+50+15, 100, "center" );

a = true;

b = false;

result = a || b;

canvas.drawStringRect( result, x+100, y+100+15, 100, "center" );

a = false;

b = false;

result = a || b;

canvas.drawStringRect( result, x+200, y+100+15, 100, "center" );

// COMPLETE THE TABLE

canvas.paint();

Exercise 3.4 – My thoughts.

When I first approached this exercise I was very much overwhelmed by the massive amount of information which was display. I instantly thought that this was going to be a hard task but when I read through the instructions the solution to this exercise was quite simple and straight forward.

I didn’t have any problems with this exercise.

Exercise 3.5 Experiments with graphics

The purpose of this exercise is to experiment with the loop control value and other variables when drawing squares. The starting point is the last part of the demonstration program − load the initial code now and run it to remind you of the output.

Exercise 3.5 initial Code

// Ex3-5 : Graphics Experiments

// Part of demo program:

var canvas;

canvas = openGraphics();

var x;

var y;

var size;

x = 10;

y = 10;

size = 200;

while( size > 0 )

{

canvas.drawRect( x, y, size, size );

x = x + 10;

y = y + 10;

size = size - 20;

}

canvas.paint();

Exercise 3.5 – Task 1

My first task on this exercise was to change the locations values of the “X” and the “Y” coordinates so that the location of the image id different and also change the value of the size variable so that the size of the image is also different.

x = 10;

y = 10;

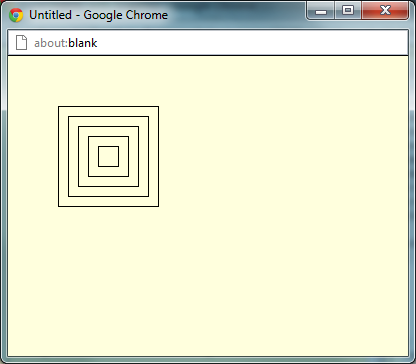
size = 200;

I changed it to

x = 50;

y = 50;

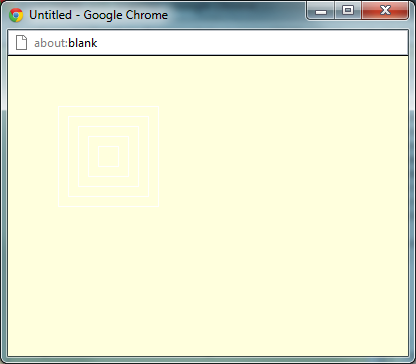
size = 100;

If you compare these two screenshots you can see that the secondary screen is much different from the initial unaltered image. The location has changed and the size is also great than that of the secondary image.

Exercise 3.5 – Task 2

My next task was to change the colour of the square from black to white. I did this by writing the standard

canvas.setColor("White");

code and as a result you are left with this screenshot.

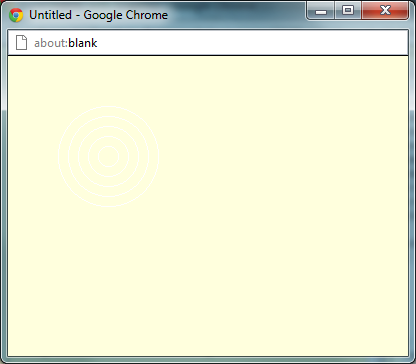
The colour of the square has changed from black to white. It is difficult to see but it is there.

Exercise 3.5 – Task 3

This task involves the use of the

canvas.drawEllipse( x, y, size, size );

code and as a result you are left with a circle image instead of a square.

This can be seen through this screenshot.

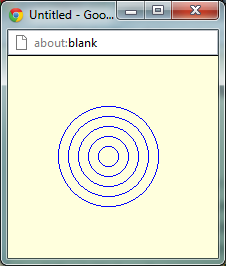
As we can see from this screenshot the square has change to a circle. This means that the code I implemented earlier has worked correctly.

Exercise 3.5 – Task 4

In this task I had to change the image from white to blue. This is a simple as possible all it involves is changing the colour within the setColor brackets

canvas.setColor("blue");

As a result I am now left with this image.

As we can see from this screenshot the colour of the cirlc ehas changed from white to blue.

Exercise 3.5 – Task 5

I removed the variable ofsize and made two new variables which we “width” and “height”.

var height;

var width;

height = 75;

width = 40;

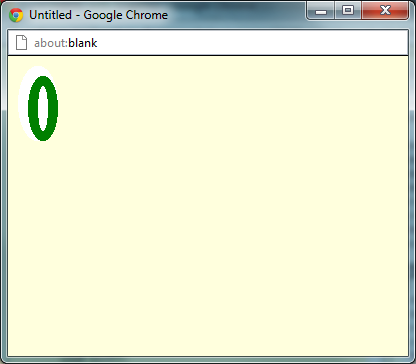
I then implemented the while formula

while( (width > 0 ) && (height > 0) )

{

Exercise 3.5 – Task 6

I then altered the height and the width until I found a shape which I was happy with.

This was my final shape

Exercise 3.5 – Final code

// Ex3-5 : Graphics Experiments

// Part of demo program:

var canvas;

canvas = openGraphics();

var x;

var y;

var height;

var width;

x = 10;

y = 10;

height = 75;

width = 40;

while( (width > 0 ) && (height > 0) )

{

canvas.setColor("white");

canvas.fillEllipse( x, y, width, height );

x = x + 10;

y = y + 10;

height = height - 20;

width = width - 20;

canvas.setStroke(10);

canvas.setColor("green");

canvas.drawEllipse( x, y, width, height );

x = x + 10;

y = y + 10;

height = height - 20;

width = width - 20;

}

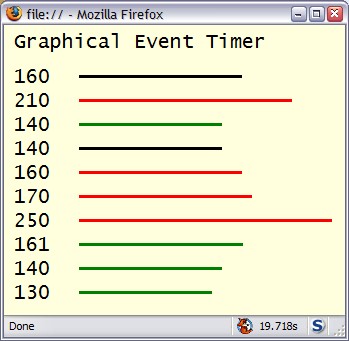
canvas.paint();

Exercise 3.5 – My Thoughts

I really like this exercise is was similar to what have been doing in the past with shapes. I like this exercise as it repeated some of the commands which have seen in this past like setColour and setStroke which I think is a good thing.

I didn’t really think I had any problems when tackling this exercise.

Exercise 3.6 Graphical Events Timer

This program is going to take the core of the event timer that you built for Exercise 2-7. The event timer will be run a fixed number of times and a graphical representation of the interval shown as a line graph. What we are aiming for is something like this:

The core of the events timer Exercise was.

var canvas;

canvas = openGraphics();

canvas.setColor( "crimson" );

canvas.setFont( "chiller", "50px", Font.BOLD );

canvas.drawString("Event Timer" , 10, 10);

var date1 = new Date();

alert( "Press \"OK\" to start the timer." );

var date2 = new Date();

alert( "Press \"OK\" to stop the timer." );

var difference;

difference= date2 - date1;

var seconds = difference/1000;

canvas.drawString( "The difference was: " + seconds + " seconds", 10, 70 );

canvas.paint();

Exercise 3.6 – Task 1

Put a loop in place that will execute 3 times and put into the loop body the code that captures the event time. When you run this program you should get five opportunities to measure the event before it stops. You may use either a while or a for loop - the choice is yours it doesn't matter which one you use.

I have input some code so that it loops around the three times

var loop = 1;

while(loop <= 3)

{

canvas.setColor( "crimson" );

canvas.setFont( "chiller", "50px", Font.BOLD );

canvas.drawString("Event Timer" , 10, 10);

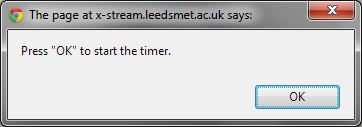
var date1 = new Date();

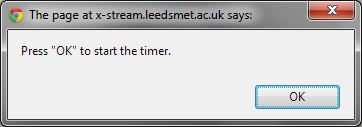
alert( "Press \"OK\" to start the timer." );

var date2 = new Date();

alert( "Press \"OK\" to stop the timer." );

loop = loop + 1;

}

🡨🡪

X3

This process is looped around three times.

Exercise 3.6 – Task 2

We need identifiers for the x and y positions of the text and the start values for the lines so declare four suitable identifiers.

var loop = 1;

var textX = 20;

var textY = 30;

var lineX = 60;

var lineY = (textY + 5);

// some possible starting values

textX = 20;

textY = 30;

lineX = 60;

lineY = textY + 5;

while(loop <= 3)

{

This was the code which I input it the program. These are my x and y position identifiers

Exercise 3.6 – Task 3

Use the textX and textY values to output the interval times similar to those shown in the example. You will need to increase the value of textY each time around the loop. Move thecanvas.paint() function call INTO the loop body, but at the very end, so that the text is output each time the loop executes.

var canvas;

canvas = openGraphics();

var loop = 1;

var textX = 20;

var textY = 30;

var lineX = 60;

var lineY = (textY + 10);

while(loop <= 3)

{

alert( "Press \"OK\" to start the timer." );

var date1 = new Date();

alert( "Press \"OK\" to stop the timer." );

var date2 = new Date();

var difference;

difference= date2 - date1;

lineY = (lineY + 20);

loop = loop + 1;

canvas.paint();

}

Exercise 3.6 – Task 4

For each event, we need to draw the line. The drawLine function requires four values, the (x,y) coordinates for the start and the end. So it will look something like:

canvas.drawLine( lineX, lineY, lineX + time, lineY );

var canvas;

canvas = openGraphics();

var loop = 1;

var textX = 20;

var textY = 30;

var lineX = 60;

var lineY = (textY + 10);

while(loop <= 3)

{

alert( "Press \"OK\" to start the timer." );

var date1 = new Date();

alert( "Press \"OK\" to stop the timer." );

var date2 = new Date();

var difference;

difference= date2 - date1;

canvas.drawString( difference, 5, (lineY - 8));

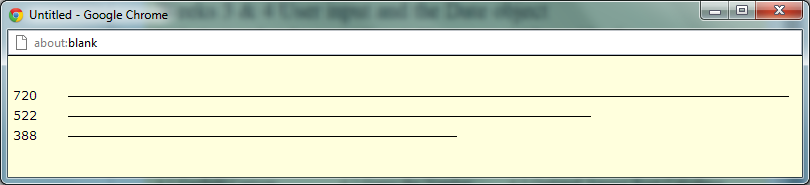
canvas.drawLine( lineX, lineY, lineX + difference, lineY );

lineY = (lineY + 20);

loop = loop + 1;

canvas.paint();

}

As we can see from this screen shot the program has looped three times and at each interval it has recorded it on the graph.

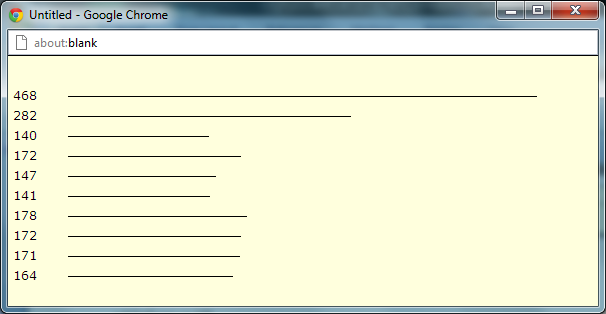
Exercise 3.6 – Task 5

I then was tasked with the instruction to make the program loop 10 times. The way in which I implements this change was a simple altering of the loop variable.

while(loop <= 3)

was changed to

while(loop <= 10)

as a result I was left with

as you can see from this screenshot the change has worked the graph had recorded 10 different loops.

Exercise 3.6 – Task 6

The 6th task in this exercise instructed me to make a title of this event timer. This was a simple piece of code which I had used before. The exact code that I used was

canvas.setColor( "crimson" );

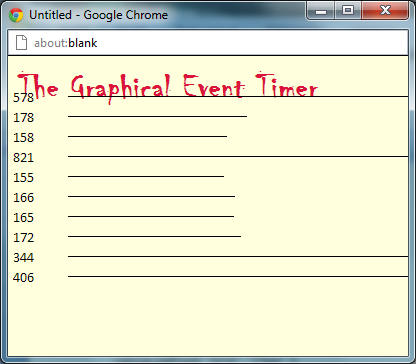
canvas.setFont( "chiller", "35px", Font.BOLD );

canvas.drawString("The Graphical Event Timer" , 10, 10);

canvas.setColor( "black" );

canvas.setFont( "cambrian", "15px" );

These two pieces of code resulted in this screenshot



As you can see from this screenshot the code has worked correctly as the title has appeared and it matches my preferences which I input. So as a result I can safely say the code has worked.

Exercise 3.6 – Extention

You will need another identifier so that you can store both the current interval time, and the last interval. For all times through the loop, check if the current time is less than the previous time and change the line colour to green. If the time is longer, change the colour to red. If the two times are the same, leave the line colour the same as the text (black in the example).

The code that I imput so that the lines change colour depending on thedifference are

if(lastvalue == current)

{

canvas.setColor("black");

canvas.setStroke("2");

canvas.drawLine( lineX, lineY, lineX + current, lineY );

}

else if(lastvalue > current)

{

canvas.setColor("green");

canvas.setStroke("2");

canvas.drawLine( lineX, lineY, lineX + current, lineY );

}

else if(lastvalue < current)

{

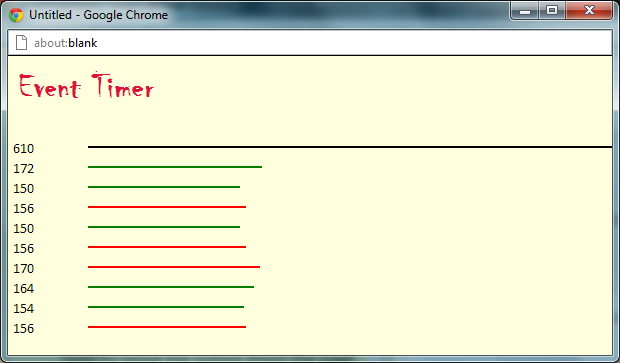
canvas.setColor("red");

canvas.setStroke("2");

canvas.drawLine( lineX, lineY, lineX + current, lineY );

}

lastvalue = current;

As a result I am left with this screenshot which has line of different colours. As a result I can safely say that this code I have implemented has worked

Exercise 3.6 –My thoughts.

I didn’t like this exercise at all it was very complex and too challenging. I required huge amounts of help from previous pieces of code like and it took me a very long time to complete.

Exercise 3.7 Sudoku Grid

The purpose of this program is to draw the outline for a blank Sudoku grid.

Exercise 3.7 Initial Code

// Ex2-7 : Blank Sudoku Grid

var canvas;

canvas = openGraphics();

var x;

var y;

var gap;

x = 20;

y = 20;

gap = 25;

// so not much help here!

canvas.paint();

Exercise 3.7 Task 1

I had to outline to Sudoku boxes.

var canvas;

canvas = openGraphics();

var linex, liney, xline, yline;

var gap = 25;

var gap1 = 75;

var ver = 1;

var width = 222;

var hor = 50;

var horizontal = 125;

xline = 50;

yline = 1;

linex = 50;

liney = 25;

canvas.setFont( "Chiller", "20px", Font.BOLD );

canvas.drawString ("Master Suduko Grid", 103);

for(ver=1; ver<=10; ver=ver+1)

{

yline = yline +25;

canvas.drawLine (xline, yline, xline +width, yline);

canvas.drawLine (hor, 26, hor, 250);

hor = hor + gap;

canvas.setStroke(1);

canvas.paint();

}

for(ver=1; ver<=2; ver=ver+1)

{

canvas.setStroke(5);

liney = liney +75;

canvas.drawLine (linex, liney, linex +width, liney);

canvas.paint();

}

for(hor=1; hor<=2; hor=hor+1)

{

canvas. setStroke(5);

canvas.drawLine (horizontal, 26, horizontal, 250);

horizontal = horizontal +gap1;

canvas.paint();

}

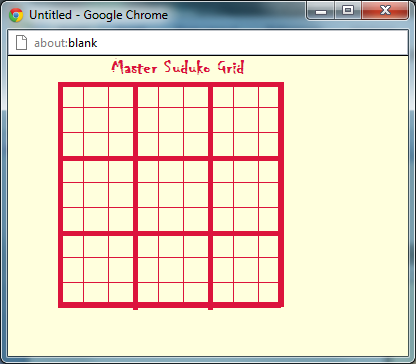
var x = 50;

var y = 26;

var size = 220;

canvas.drawRect(x,y, size, size);

canvas.paint();



As we can see from this screenshot, the code which I have created has worked and the boarders of the Sudoku have been outline and thickened. This is due to the setStoke function as used in previous exercises. I have also created my own personalise title. The font is “chiller” and I have also changed the colour of the Sudoku box itself. I changed it to “crimson” colour

Exercise 4.1 Garfield Cartoon

In this task I had to complete the complete the Garfield comic strip and apply the text as well.

Exercise 4.1 Intro Code

// Ex4-1: Garfield Cartoon

var canvas;

canvas = openGraphics();

canvas.paint();

The way in which I did this was starting with the variables as you would normally. I then added some code in the programme so that all three images could be displayed. The exact code that I used was

var imageName;

var xpoisition = 20;

var ypoisition = 20;

imageName = "garfield1.gif";

canvas.drawImage( imageName, xpoisition, ypoisition, 199, 177);

imageName = "garfield2.gif";

canvas.drawImage( imageName, xpoisition+199, ypoisition, 203, 177);

imageName = "garfield3.gif";

canvas.drawImage( imageName, xpoisition+199+203, ypoisition, 199, 177);

This code displayed 3 separate images of the comic strip

As you can see from this screenshot. The code that I just imput has worked correctly and you can see three different images. This means that the code has worked correctly.

I then had to add the text for the second part of this exercise. The way in which I did this was with a simple canvas.drawString code. The exact code that I used was

canvas.setFont("comic sans ms", "10px", Font.BOLD);

canvas.drawStringRect ("There's A Good Scary Movie on TV Tonight", 32, 25, 150, " center");

canvas.drawStringRect ("Yeah Sure", 140, 67, 50, " center");

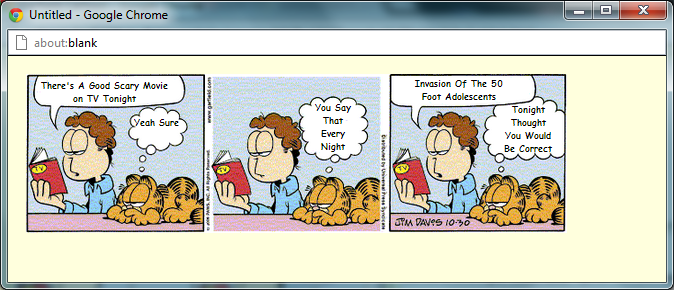
canvas.drawStringRect ("You Say That Every Night", 335, 50, 50, " center");

canvas.drawStringRect ("Invasion Of The 50 Foot Adolescents", 445, 23, 110, " center");

canvas.drawStringRect ("Tonight Thought You Would Be Correct", 548, 52, 60, " center");

The third value determines the centre location ^

This code will display text and with the application of the X and Y variables which I implemented earlier I was able to change the text so that it could fit into the thought bubbles exactly.

As a result of this code I was then presented with what I hope was a cartoon strip with text in their allocated thought bubbles.

As we can see from this screenshot there are 3 separate images and there is text inside each of the thought bubbles.

Exercise 4.1 Final Code

// Ex4-1: Garfield Cartoon

var canvas;

2

canvas = openGraphics();

var imageName;

var xpoisition = 20;

var ypoisition = 20;

imageName = "garfield1.gif";

canvas.drawImage( imageName, xpoisition, ypoisition, 199, 177);

imageName = "garfield2.gif";

canvas.drawImage( imageName, xpoisition+199, ypoisition, 203, 177);

imageName = "garfield3.gif";

canvas.drawImage( imageName, xpoisition+199+203, ypoisition, 199, 177);

canvas.setFont("comic sans ms", "10px", Font.BOLD);

canvas.drawStringRect ("There's A Good Scary Movie on TV Tonight", 32, 25, 150, " center");

canvas.drawStringRect ("Yeah Sure", 140, 67, 50, " center");

canvas.drawStringRect ("You Say That Every Night", 335, 50, 50, " center");

canvas.drawStringRect ("Invasion Of The 50 Foot Adolescents", 445, 23, 110, " center");

canvas.drawStringRect ("Tonight Thought You Would Be Correct", 548, 52, 60, " center");

canvas.paint();

Exercise 4.1 My thoughts

Didn’t think this exercise was all that bad. It only required two different commads which were display the image and write the text and I knew how to do this. The only problem which I encountered was moving the code around the canvas to match up with the location of the thought bubbles.

Exercise 4.2 Guess the number

In this task I had to create a numbers guessing game. This was done by making a function in which you had to guess the exact number, and the game would say if the number was higher or lower than your guess.

Initial Code

/ Ex4-2: Guess the Number

var canvas;

canvas = openGraphics();

var max;

max = 100;

var numberToGuess;

numberToGuess = 50; // use a known value for testing

// (1) canvas.drawString( ... suitable title ......

var guess;

// (2) get a guess from the user and make sure it is a numerical value

var message;

// (3) if too high .....

// (4) if too low .....

// (5) "just right" said Goldilocks .....

// (6) canvas.drawString( message ......

canvas.paint();

Exercise 4.2 Task 1

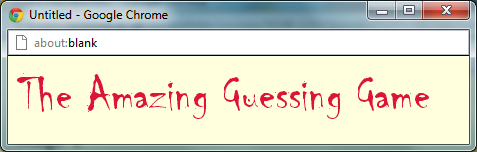
My first task was to load up the initial code and implement code to output a suitable title. This was doing using the simple drawString/setFont/setColor commands. The exact code that I used was

canvas.setFont ( "chiller", "50px", "set.FONT");

canvas.setColor("crimson");

canvas.drawString("The Amazing Guessing Game", 10, 10);

As a result I was presented with this output box.

As we can see from this screenshot the cod e that I have implemented has resulted in an appropriate name, font and colour for my guessing game.

I also needed to imput the code again below the drawstring so that the text after this title were not displayed in the same, colour, font and size.

canvas.setFont ( "cambria", "15px", "set.FONT");

canvas.setColor("black");

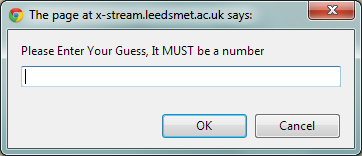
Exercise 4.2 Task 2

My next task was to add the code needed to get a numerical input from the user.

This was done using a simple prompt code.

var guess = prompt ( "Please Enter Your Guess, It MUST be a number");

The code above provided the user with an input box in which to place a guess. I also stated in the code that the guess must be a numerical value or it would not work.

 As we can see from this screenshot the code I input above has worked correctly and looking from this screenshot the prompt has stated that the input guess must be a number.

Exercise 4.2 Task 3

My third task was to add three if() statements that will set the value of message to appropriately as a result of comparing the values of guess and numberToGuess.

The way in which I did this was I made three if statements which their own conditions and if the user input matched the conditions of one of the if statements than they would be presented with the corresponding answer. The answer being the Guess was too high, the guess was too low or the guess was correct.

The exact code which I used was

if(guess > numberToGuess)

{

canvas.drawString("Your Guess Was Too High", xposition, yposition);

}

canvas.paint();

if(guess < numberToGuess)

{

canvas.drawString("Your Guess Was Too Low", xposition, yposition);

}

canvas.paint();

if(guess == numberToGuess)

{

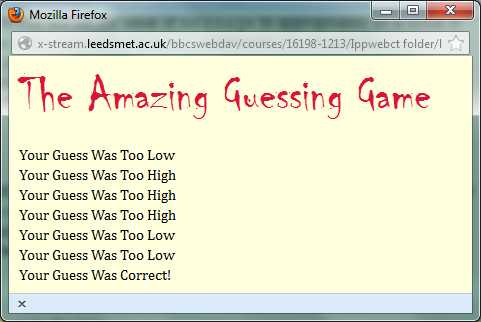
canvas.drawString("Your Guess Was Correct!", xposition, yposition);

guessed = true;

}

canvas.paint();

As a result of this code I will be presented with either “too high”, “too low”, or “correct” as my responses.

As we can see from this screenshot there are numerous results which have been presented. As a result I can see that the code has worked correctly and all three if statements work correctly.

Exercise 4.2 Task 4

My fourth task was to create an output line

canvas.drawString("Your Guess Was Correct!", xposition, yposition);

This was the code that I input to create my output.

This can be seen from the previous screenshot

Exercise 4.2 Task 5

My fifth task in this section was to implement a loop into the equation. The exact code that I used to create a loop was

These two lines of code were placed at the beginning of the if Statements.

loop = 10; 🡨

while (loop >= 0)

This line of code were placed at the end of the if Statements. This code was to stop the loop continuing for an infinite amount of time

loop= loop - 1; 🡨

As a result I was presented with a prompt after prompt asking for a value until the value input was correct.

Exercise 4.2 Section 2

Exercise 4.2 Task 1

In this section I had to declare another identifier, called guessed and assign it the value false. This declaration should be near the declaration for guess.

The exact code that I used was

var guessed;

I put this code beside all my other variables as the top of the code.

Exercise 4.2 Task 2

I then had to Put ALL the code from the section that gets the user's input to the very end of the program into the body of a while loop where the condition is:

while( !guessed )

Include canvas.paint(); inside the loop.

This code here would result in the loop continuing until the number input matched the number you had to guess.

This is what the code looked like when I added the two pieces of code above

while( !guessed ){

var guess = prompt ( "Please Enter Your Guess, It MUST be a number");

yposition = yposition + 20;

if(guess > numberToGuess)

{

canvas.drawString("Your Guess Was Too High", xposition, yposition);

}

canvas.paint();

if(guess < numberToGuess)

{

canvas.drawString("Your Guess Was Too Low", xposition, yposition);

}

canvas.paint();

if(guess == numberToGuess)

{

canvas.drawString("Your Guess Was Correct!", xposition, yposition);

guessed = true;

}

canvas.paint();

}

canvas.paint();

The program will continue to loop until the correct value is found.

Exercise 4.2 Task 3

I then had to add code so that the programm could detect if the answer was correct.

The exact code which I input was

if(guess == numberToGuess)

{

canvas.drawString("Your Guess Was Correct!", xposition, yposition);

guessed = true;

}

canvas.paint();

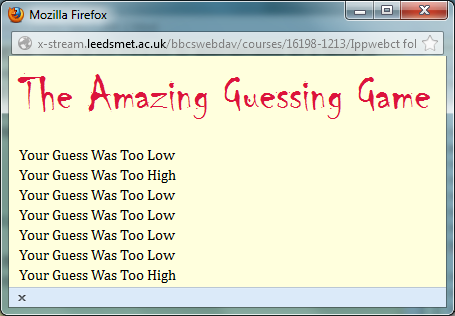
This code would then stop the loop from repeating as the code above tells stops the program as the correct value has been found.

Exercise 4.2 Task 4

I has to change the location of each of the output responses or else they would just remain in one area. The way in which I stopped this from happening was by putting in some new code which meant that the location of the output from the continuous loop was different.

The exact code I used was

yposition = yposition + 20;

This code results in the current location of the y position text which was 70 having an additional 20 on the scale and as a result the output moves down the output box.

As we can see from this screenshot the output responses move continuously down the y position of the output box.

Exercise 4.2 Task 5

In this task I had to input code so that the program would pick a number randomly and so the game could not be easily beaten by the user who set the number to guess.

The code which I input was

numberToGuess = Math.floor( Math.random() \* max ) + 1;

The code above replaced the standard

numberToGuess = 50;

As a result the program chooses the number and the game is now completely and fully operational.

Exercise 4.2 Extensions

I then decided to change the colour of the output response. For example if the guess was too high the colour would be displayed in a red colour. If the guess was to low then the colour would be displayed in a blue colour and if the prediction was correct then the colour of the text would be green.

This was a simple setColour code which was input in each of the if statements for example if(guess < numberToGuess)

{

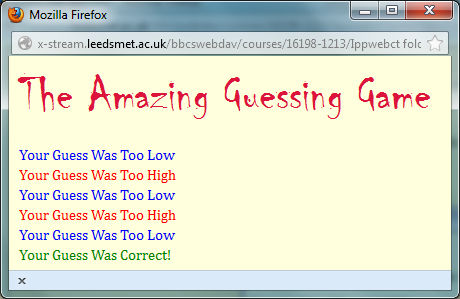
canvas.setColor("blue");

canvas.drawString("Your Guess Was Too Low", xposition, yposition);

}

canvas.paint();

This code would mean that the output information would be displayed in a blue.



As we can see from this screenshot the code which I have input in each of the if statements has worked correct and all 3 if statements has worked correctly as the low guesses are blue, high guesses are red and correct guesses are green.

Exercise 4.2 Final Code

// Ex4-2: Guess the Number

var canvas;

canvas = openGraphics();

var max;

max = prompt("what do what want to maximum number to be?");

var xposition = 10;

var yposition = 70;

var loop;

var guessed;

var numberToGuess;

numberToGuess = Math.floor( Math.random() \* max ) + 1;

canvas.setFont ( "chiller", "50px", "set.FONT");

canvas.setColor("crimson");

canvas.drawString("The Amazing Guessing Game", 10, 10);

var message;

canvas.setFont ( "cambria", "15px", "set.FONT");

canvas.setColor("black");

loop = 10;

while( !guessed ){

var guess = prompt ( "Please Enter Your Guess, It MUST be a number");

yposition = yposition + 20;

if(guess > numberToGuess)

{

canvas.setColor("red");

canvas.drawString("Your Guess Was Too High", xposition, yposition);

}

canvas.paint();

if(guess < numberToGuess)

{

canvas.setColor("blue");

canvas.drawString("Your Guess Was Too Low", xposition, yposition);

}

canvas.paint();

if(guess == numberToGuess)

{

canvas.setColor("green");

canvas.drawString("Your Guess Was Correct!", xposition, yposition);

guessed = true;

}

canvas.paint();

}

canvas.paint();

Exercise 4.2 my thoughts

I thought this exercise was very simple the only problem which I encountered in this exercise was my location of specific code. I forgot to input guessed = true; code inside the if statement which determined the correct response for my game what going round on a continuous loop not ending the game and not knowing if the answers we gave were even correct in the first place.

Exercise 4.3 – BMI Calculator

In this exercise I had to produce a BMI calculator. I did this in a long series of instructions.

Exercise 4.3 Initial Code

// Ex4-4: BMI Calculator

Exercise 4.1 Task 1 – The simplest BMI calculator

In this task I had to declare three identifiers. These identifiers were height, weight and BMI. The exact code which Input was.

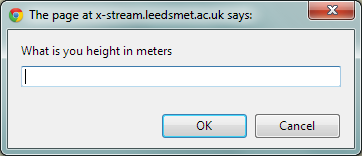
var height;

var weight;

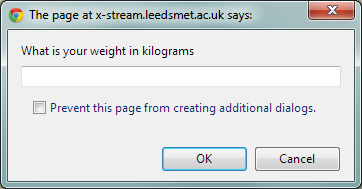
var BMI;

Secondly I had to Assign values to the height and weight; pick values from the test data table in Step 3 of the Introduction. This was easily done using prompt commands. The exact code which I used was.

Height = prompt( "What is you height in meters");

Weight = prompt( "What is your weight in kilograms");

As you can see from these two screenshots the code has worked correctly and the prompts have been displayed as I programmed them to do.

The value in the two prompts boxes which them be calculated by the formula which I input and ultimately a BMI will be calculated.

I then had to create an expression to calculate the BMI and assign it to the BMI identifier. I did this by input a formula into my program. The exact piece of code that I used to calculate the BMI was.

BMI = weight / (height \* height);

This formula determines the BMI from the values which I input in my height and weight prompts.

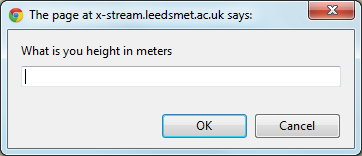
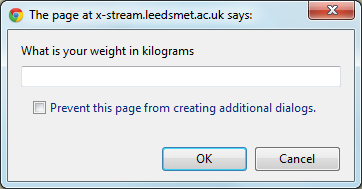
Exercise 4.1 Task 2 – Add user imput

I then had to change the code for the values so that the user can have an input. This was done using simple prompt code. The exact code that I used was

height = prompt( "What is you height in meters");

weight = prompt( "What is your weight in kilograms");

This code would display two output boxes asking for the weight and the height of the user.



As you can see from these screenshot above, the code has worked and I have been presented with two prompts. The values that the user put in these boxes will determine the value which is presented in the BMI calculator.

I then had to add further code so that the users input can be calculated has a whole number. This can be done using the code parseFloat.

height = parseFloat(height);

weight = parseFloat(weight);

Exercise 4.1 Task 3 – Classify the BMI

I then had to write more code into my program, so that the values determine by the BMI formula I input are classified. These way in which I did this was through standard if formulas. The exact code that I used was.

if(BMI<18.5) 🡨I changed this value many times to include all values in the BMI

{

canvas.drawString( "is Underweight", 130, 70); 🡨 This drawstring code changed as well.

canvas.paint();

}

Exercise 4.1 Final Code

// Ex4-4: BMI Calculator

var canvas;

canvas = openGraphics();

var height;

var weight;

var BMI;

height = prompt( "What is you height in meters");

weight = prompt( "What is your weight in kilograms");

height = parseFloat(height);

weight = parseFloat(weight);

BMI = weight / (height \* height);

canvas.drawString("Your Height is :" + height, 10, 10);

canvas.drawString("Your Weight is :" + weight, 10, 30);

canvas.drawString("Your BMI is: " + BMI , 10, 50);

canvas.drawString("Your Classification: " , 10, 70);

if(BMI<18.5)

{

canvas.drawString( "is Underweight", 130, 70);

canvas.paint();

}

else if(BMI>18.5 && BMI<= 24.9)

{

canvas.drawString( "is Ideal", 130, 70);

canvas.paint();

}

else if(BMI>=25 && BMI<= 29.9)

{

canvas.drawString( "is Overweight", 130, 70);

canvas.paint();

}

else if(BMI >=30)

{

canvas.drawString( "is Obese", 130, 70);

canvas.paint();

}

else if(BMI > 40)

{

canvas.drawString( "is Very Obese", 130, 70);

canvas.paint();

}

canvas.paint();

Exercise 4.1 My thoughts

I felt that this program was fairly easy; I thought that due to the fact that the code was similar to the “Grading Work” I was able to complete the work quite quickly, the only time consuming task was the research I had to do to find out about the requirements for each classification. The if statements were very easy to complete as I had already done them before. The only problem which I did encounter was the parseFloat code. I didn’t know how to write them so as a result I research them and found the correct way.

Exercise 5.1 - Weather Data Graph

In this task I had to create a program that will accept some of this data, input by the user, and to produce a graph that is suitably formatted.

Exercise 5.1 - Initial Code

var canvas;

canvas = openGraphics();

//You may have to decide where this is best placed

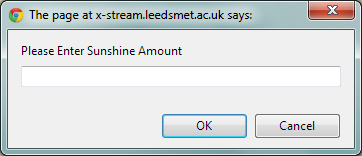
canvas.paint();

Exercise 5.1 Task 1 – Input from the user

In order to record the information on my graph I first need to gain the information to calculate the size of the bars in my graph. The values are from the user and inputting a standard prompt code I can easily get this information. The exact code that I used was

var value = prompt("Please Enter Sunshine Amount");

value = parseInt(value,10);

As we can see from this screenshot the code has worked and as a result I have been presented with this prompt box.

Exercise 5.1 Task 2 - Drawing my Graph

I first had to declare my variables so that I can then write more code to draw my axis in which my graph will rest on. The exact code that I used to declare my variables was.

var canvas;

canvas = openGraphics();

var xpos = 30;

var ypos = 40;

var height = 225;

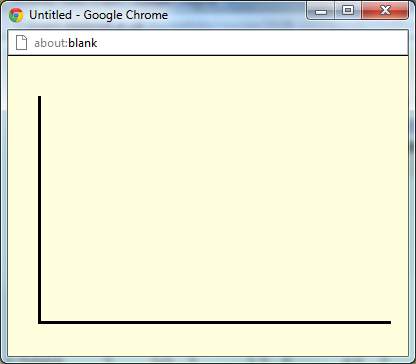
var width = 350;

canvas.setStroke(3);

canvas.drawLine(xpos, ypos + height, xpos, ypos);

canvas.drawLine(xpos, ypos + height, xpos+ width, ypos + height);

canvas.paint();

As a result, after I declared my variables and wrote the code which displayed the axis for my graph I was presented with an X position line and a Y position line. As we can see from this screenshot the code that I have input has worked correctly and I have been presented with this axis. This means that the code has worked correctly. The code now provides me with space to add bars and record the values that I input into my program.

Exercise 5.1 Task 3 - Naming my axis

I then had to write some code so that the x axis and the y axis had values to that any input I put in has meaning. The way in which I did this was inputting code after my drawLine code. The exact code that I used was.

// the code below is the values on the y axis going up in 45 until it reaches 225

var ylabel = 0;

while(ylabel <= 225)

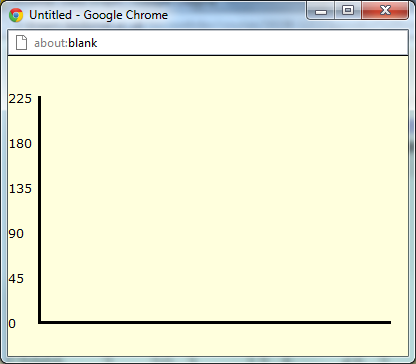
{

canvas.drawString(ylabel, 0, ypos + height - ylabel - 5);

ylabel += 45;

}

This code meant that the Y axis had values on it.

As you can see from this screenshot, the Y axis has values and they are increasing in 45s until it reaches 225.

I then had to name my X axis; this consisted of the months of the year. The way in which I did this was again through code. The code which I used to name the x axis was this code.

//the code below is the values on the x axis going from January to December.

var months = "JFMAMJJASOND";

var xlabel = xpos + 10;

var barwidth = width/12;

for (var loop = 0; loop < 12; loop++)

{

var nextLetter= months.charAt(loop);

canvas.drawString(nextLetter, xlabel, ypos + height + 10);

xlabel = xlabel + barwidth;

}

xpos = xpos + 3;

This code above names the x axis. As you can see from this screenshot below we can see from this screenshot the x axis has been named so that means the code has worked correctly.



We can now see from the screenshot above that both the x axis and the y axis are names and have values on them.

Exercise 5.1 Task 4 – Creating the bars

I then had to create a loop so that the bar could represent the 12 months of the year. The way in which I did this was through a standard loop. The exact code that I used was

// the code below creates the bars and gives them specific shapes, outlines and colours

for (var month = 0; month < 12; month++)

{

var value = prompt("Please Enter Sunshine Amount");

value = parseInt(value,10);

canvas.setStroke(Stroke.Dotted);

canvas.setColor("pink");

canvas.fillRect( xpos, ypos + height - value, barwidth, value);

canvas.setColor("black");

canvas.drawRect( xpos, ypos + height - value, barwidth, value);

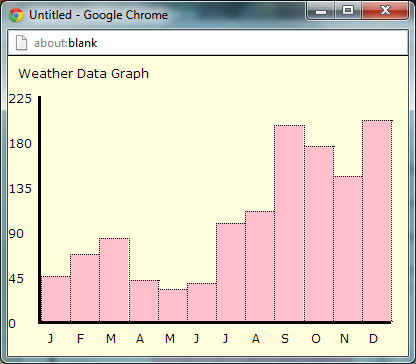
xpos = xpos + barwidth;

canvas.paint();

}

The code above determines the size, colour and shape of the bars which display the users inputted values.

The screenshot below shows the effects of that code.

As we can see from the screenshot each of the bars is divided perfectly across the x boarder. Due to the code

var barwidth = width/12

This code means that the bar is divided across 12 months

Exercise 5.1 – Final Code.

//exercsie 5.1

var canvas = openGraphics();

var xpos = 30;

var ypos = 40;

//the code below creates the axis for my graph to rest on.

var height = 225;

var width = 350;

canvas.setStroke(3);

canvas.drawLine(xpos, ypos + height, xpos, ypos);

canvas.drawLine(xpos, ypos + height, xpos+ width, ypos + height);

// the code below names the y axis. going up in 45s until 225

var ylabel = 0;

while(ylabel <= 225)

{

canvas.drawString(ylabel, 0, ypos + height - ylabel - 5);

ylabel += 45;

}

canvas.drawString("Weather Data Graph", 10 , 10);

//The code below names the x axis. a letter for each letter of every month

var months = "JFMAMJJASOND";

var xlabel = xpos + 10;

var barwidth = width/12;

for (var loop = 0; loop < 12; loop++)

{

var nextLetter= months.charAt(loop);

canvas.drawString(nextLetter, xlabel, ypos + height + 10);

xlabel = xlabel + barwidth;

}

xpos = xpos + 3;

// the code below creates the bars and gives them specific shapes, outlines and colors

for (var month = 0; month < 12; month++)

{

var value = prompt("Please Enter Sunshine Amount");

value = parseInt(value,10);

canvas.setStroke(Stroke.Dotted);

canvas.setColor("pink");

canvas.fillRect( xpos, ypos + height - value, barwidth, value);

canvas.setColor("black");

canvas.drawRect( xpos, ypos + height - value, barwidth, value);

xpos = xpos + barwidth;

canvas.paint();

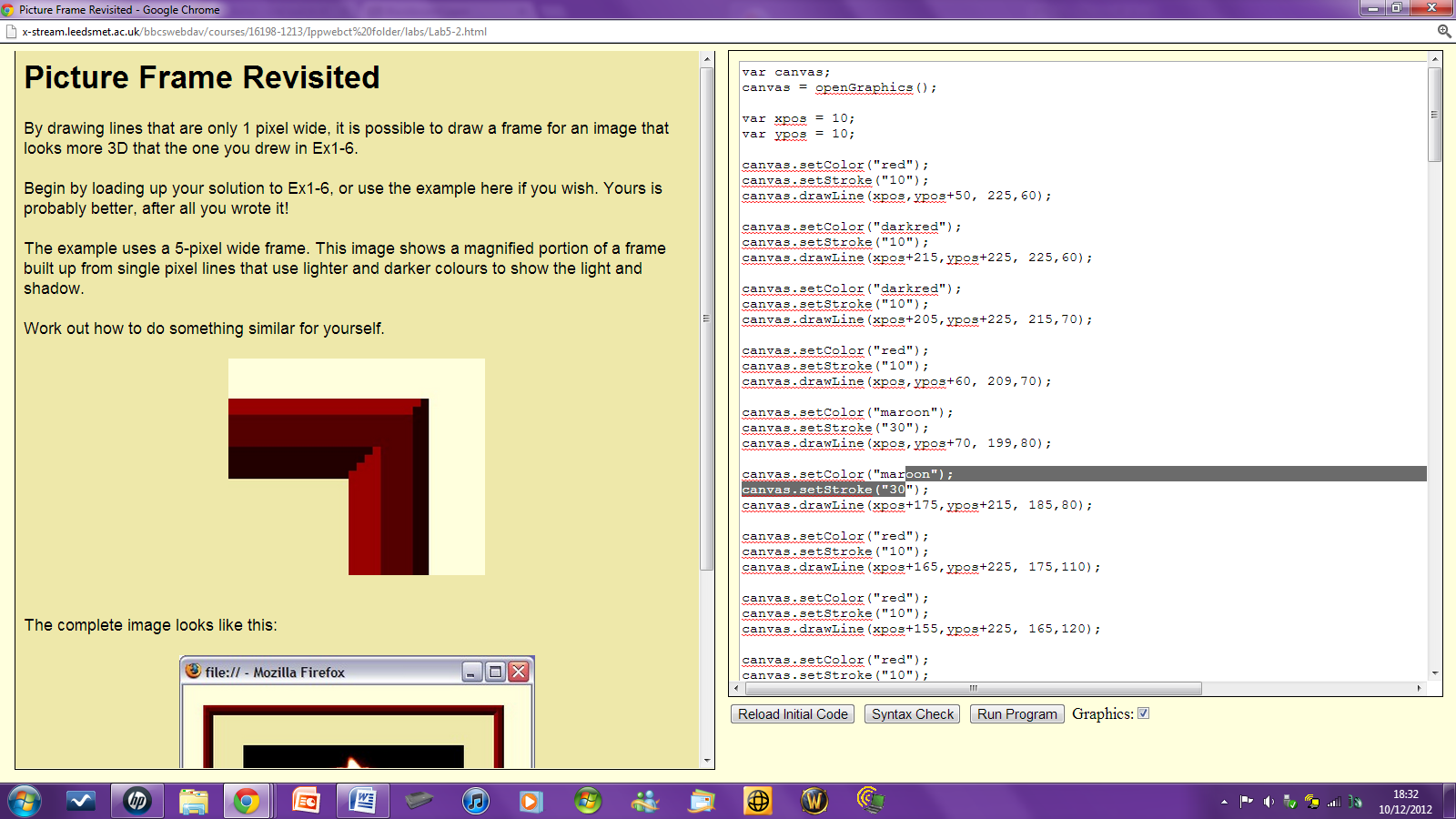
}

Exercise 5.1 – My thoughts.

I felt that this exercise was a very hard exercise. Due to the placing and the creation of the bars, the code which you had to write so that the names of the axis were positioned correctly and the axis itself. The majority of the code was quite complex and at some times challenge, the task was time consuming when it came to understanding what everything did.

Exercise 5.2 – Picture Frame Revisited.

In this exercise I had to use code to create this image below.

The way in which I went about writing the code and creating this image was through the code known as canvas.drawLine.

I started this exercise with the simple canvas.drawLine code and declaring all my variables which I required.

The exact code that I used was this.

var canvas;

canvas = openGraphics();

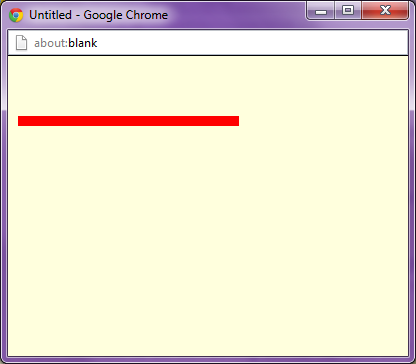
var xpos = 10;

var ypos = 10;

canvas.setColor("red");

canvas.setStroke("10");

canvas.drawLine(xpos,ypos+50, 225,60);

The result of this code was that I was then presented with this image.

As we can see from this screenshot the code has worked and I have been provided with a red line which is exactly what I wanted and provides the foundation for the next part of the exercise and will undoubtedly help me finishing it.

Exercise 5.2 – Complete Code

I replicated the first lines of code over and over and changed the values so that the location and size of the lines would also change as it appears in the example picture. The finished code which provided me with exact image was.

var canvas;

canvas = openGraphics();

var xpos = 10;

var ypos = 10;

canvas.setColor("red");

canvas.setStroke("10");

canvas.drawLine(xpos,ypos+50, 225,60);

canvas.setColor("darkred");

canvas.setStroke("10");

canvas.drawLine(xpos+215,ypos+225, 225,60);

canvas.setColor("darkred");

canvas.setStroke("10");

canvas.drawLine(xpos+205,ypos+225, 215,70);

canvas.setColor("red");

canvas.setStroke("10");

canvas.drawLine(xpos,ypos+60, 209,70);

canvas.setColor("maroon");

canvas.setStroke("30");

canvas.drawLine(xpos,ypos+70, 199,80);

canvas.setColor("maroon");

canvas.setStroke("30");

canvas.drawLine(xpos+175,ypos+215, 185,80);

canvas.setColor("red");

canvas.setStroke("10");

canvas.drawLine(xpos+165,ypos+225, 175,110);

canvas.setColor("red");

canvas.setStroke("10");

canvas.drawLine(xpos+155,ypos+225, 165,120);

canvas.setColor("red");

canvas.setStroke("10");

canvas.drawLine(xpos+145,ypos+225, 155,130);

canvas.setColor("red");

canvas.setStroke("10");

canvas.drawLine(xpos+135,ypos+225, 145,140);

canvas.setColor("darkred");

canvas.setStroke("10");

canvas.drawLine(xpos+160,ypos+100, 10,110);

canvas.setColor("darkred");

canvas.setStroke("10");

canvas.drawLine(xpos+150,ypos+110, 10,120);

canvas.setColor("darkred");

canvas.setStroke("10");

canvas.drawLine(xpos+140,ypos+120, 10,130);

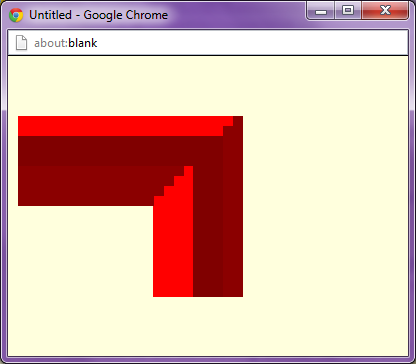
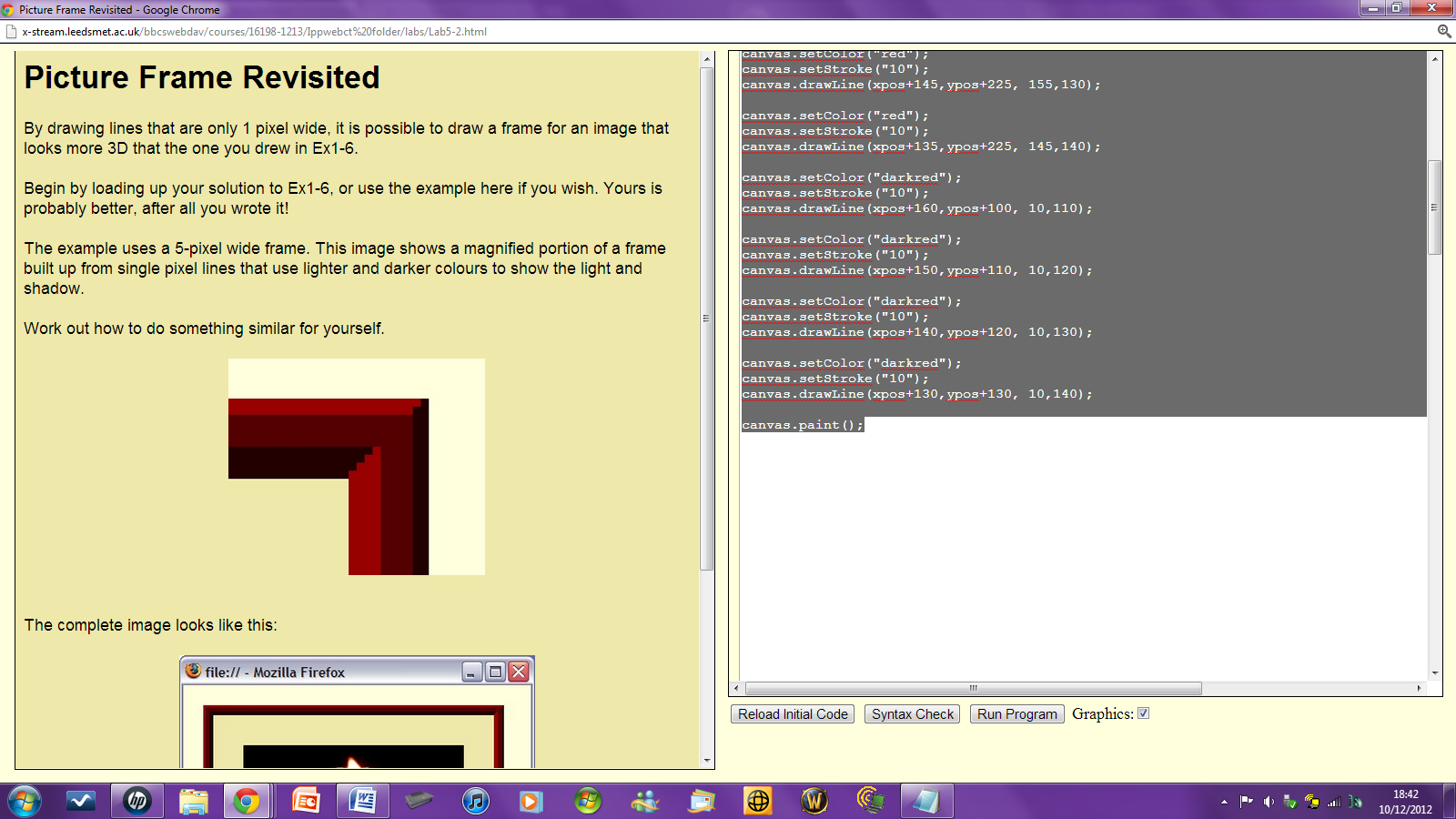
canvas.setColor("darkred");

canvas.setStroke("10");

canvas.drawLine(xpos+130,ypos+130, 10,140);

canvas.paint();

This complete code provided me with this image.



As we can see the image on the left which is my image is exactly the same as the example which was presented to me befor the exercise. This means that the code that I used is correct and functioning correctly.

Exercise 5.2 - My thoughts

I felt that this exercise was very simple the only comands which were required in this exercise were the drawLine commands. The only problem which I encountered was that the process of moving the location of the line and their location became a very tedious and time consuming task.