A dissertation submitted in partial fulfilment of the requirements for B.Sc Honours in Maths and Computing for Sciences,

Thomas Vian, 2002 - 2003

Acknowledgements

I would like to take this opportunity to thank the following;

Dr Mike Newton, my personal tutor, for all his help, support and guidance.

I would like to thank the four schools I have spent time with, and have participated in the development process;

Millbrook Junior School, Kettering.

Westglade Infant School, Nottingham.

Victoria Infants School, Wellingborough.

The Holy Family Primary School, Bristol.

I would also like to thank my friends and family for their support.

Contents Page

List of Programs

Abstract

The purpose of this project was to design an interactive website to help the teaching of addition of nine, by the method of addition of ten, then subtraction of one, to Year Two, Key Stage One children.

Java 2 was used to write an applet, which was then suitably embedded within HTML and JavaScript in order to make a complete graphical user interface that was highly machine independent.

Research within schools, looking at the government guidelines, interviews with teachers and questionnaires helped in producing this website. The final website was then taken into a local school in order to observe children working upon the website and to evaluate it fully.

1 Introduction

This chapter discusses the aims of the project. It gives an introduction to the Internet and the World Wide Web and how they work. It gives an introduction to how a website can be designed and a brief description of some possible programming languages and software that could achieve this.

1.1 Aim.

To design an interactive website to help the teaching of addition of nine, by the method of addition of ten, then subtraction of one, to Year Two, Key Stage One children.

1.2 What are the Internet and the World Wide Web?

The Internet is a number of computers located all over the world that are all connected together. They can be connected in many different ways, which are becoming more and more diverse every day. A user on these computers 'explores' the World Wide Web. The World Wide Web is an ethereal environment with an ever-changing collection of varied documents, which can contain many different types of information, sounds, videos, pictures and programs.

1.3 How does the World Wide Web work?

Computers on the Internet are mostly classified as one of two types, a server or a client. A server is a remote system that holds information. A client requests a file or document from the Server and displays or stores it. The connection between the two is determined by protocols.

Protocols are a set of rules that govern the formatting of data transmission between computers. In order for a client computer to access the World Wide Web it must have a 'browser'. A browser is a program used to surf (slang term meaning 'to access') the World Wide Web. A browser uses a Uniform Resource Locator (URL) to make a request to the relevant sever for a certain piece of information. If the information is found correctly then it will appear within the Client's browser.

1.4 How can web pages be designed?

A web page is designed to be opened by a browser. Many web pages can be combined together to form a website. There are countless different types of websites on the Internet ranging from just text, through to fully interactive sites with movies and sound. There are a few different programming languages available from which a website could be designed, but the main one is HTML (see 1.7). The advantages and applications of using HTML will be discussed later on in the document, but briefly HTML is the main programming language for the design of web pages and sites because HTML programs are saved as a text file. This means that HTML is highly versatile as virtually any one that has a computer can view and write a HTML file.

There are many pieces of software available that make it very easy to write a web page in HTML. Most of these applications let you program HTML without seeing any code, just by clicking a few buttons. There are also dedicated sites on the Internet that allow you to choose the content and then generate the code for your web page.

A disadvantage of HTML is its low degree of interactivity. To resolve this, HTML can be combined with other languages in order to achieve a greater degree of interactivity within a program.

1.5 Java.

Java is one of these languages. Java enables you to build a separate special program, called an applet, which by using HTML you can embed within a web page, to give it a degree of interactivity.

1.6 JavaScript.

JavaScript is an example of a scripting language. It is a program included on an HTML page to put interactivity into a page. JavaScript code is written along side HTML within <script> tags.

1.7 HTML.

HTML (HyperText Markup Language) has two essential features, hypertext and universality. Hypertext allows you to create a link in a piece of text that leads the user to a different part of the document or documents.

Universality means that HTML documents are saved as ‘text only’ files, so it doesn’t matter what type of computer or browser the user has, they can still use a HTML document.

1.8 What is a browser?

A browser is a program used to view, download, upload, surf, or otherwise access documents (for example, web pages) on the Internet. Netscape Navigator and Microsoft Internet Explorer are well-known "web browsers" that enable you to view and interact with websites.

1.9 What is animation?

Animation is creating the appearance of movement with drawn objects, just as in cartoons, animation gives life to websites by displaying a variety of moving objects in a browser's screen: images, words and pictures.

1.10 An introduction to Adobe LiveMotion.

The official description of LiveMotion from the Adobe website is:

> "Adobe® LiveMotion™ software is an essential tool for professional Web designers and developers who need to create dynamic, interactive content in a variety of formats, including Macromedia® Flash™ (SWF) and QuickTime™. Support for ActionScript, combined with design, coding and debugging tools, allows you to create versatile, animated content for the Web and other media."
> (Adobe Systems Incorporated, 2003)

In summary

LiveMotion is a timeline driven program, where certain properties of an object are set to different states according to the place in the timeline.

2 Review of Literature

This chapter reviews relevant previous work/programs and theoretical background of websites.

In addition to the time that I have spent in schools, I also looked at the computer programs in use by them or recommended to me. One of these programs was (NUMBER BOX 2. 2000.).

This program starts with a 'menu box' to select different options. One of these options is to 'use quick sheets', which gives the user an opportunity to choose from a selection of pre-made tasks. There is a choice of four different levels of difficulty: Yellow (Easy), Green, Blue, Red (Hard).

In each level there are different choices of pre-made tasks. The Yellow level corresponds to Year Two, Key Stage One. 'Adding Machine' is one choice on the Yellow level. It asks the child to input a number of their choice and then the child has to 'guess' what number added to this equals one hundred. When the 'calculate' button is clicked the computer finds the answer. There is no space for a child to input a second number as a 'guess', so there is no feedback on whether the child has got it right.

There are various other windows available. The 'help' window contains lengthy sentences and it looks a lot to read. The 'recalculate' window holds commands e.g. play or stop. These commands don't appear to achieve anything obvious. There is also a 'hint' window that appears when the mouse

pointer is over an object on screen, which displays an explanation of what the object is. These explanations are quite complex e.g. “this cell contains a whole number”, I think these are meant for a teacher; as a Year Two Key Stage Two child would be unlikely understand this. These windows appear over the top of the task screen. This could be confusing because it is not focussed what to do.

The other four existing programs are less applicable to addition of nine but they still hold a valuable reference as to what Year Two, Key Stage One software there is available.

‘Jellies’ is a data handling program where children in the class choose which is their favourite jelly flavour. After all the data has been collected a graph may be produced. There are also options to print or change the level. These buttons are clear and easy to click on.

‘Pet Survey’ enables a child to produce some data handling information about the class’ favourite pets. Again this has similar options to ‘Jellies’, with the options to draw a graph and print it and the data.

‘Simple Survey’ gives many different options to make a survey on, e.g. weather. It gives the child the choice of what categories to have within their survey. It does not give the option for more, unusual categories to be added.

‘Taking Away’ is similar to the ‘Adding Machine’ although the child has the option to enter two numbers and find the difference (e.g. the first subtract the second). The answer is shown when the ‘calculate’ button is clicked. Just like the ‘Adding Machine’ program there is nowhere for the child to enter a ‘guess’, therefore there is no feedback to either the teacher or the child.

The ‘Number Box 2’ programs are all very colourful and interesting (see 11.1). Although at times there are too many things going at once. From looking at this program I believe that two children working on this with an adult helper would be an ideal teaching set up, because they could all discuss their guesses and then prompt the computer to reveal the answer.

The Black Cat Company has many different software applications, which span the whole range of ICT (Information and Communication Technology) skills, Literacy skills, and Numeracy skills. I choose to look at ‘Number Box 2’ because it is the most relevant to this project.

I also looked at ‘Monty’. Monty is a snake that hides some numbers of a number square under his body (see 11.2). The child has to guess the hidden numbers. There is one of eight different number squares each time the program is started. The child could use his/her memory to play this task, because the child sees the numbers before ‘Monty’ covers them up. Otherwise the child would have to use mental strategies (similar to that of ‘Addition of Nine’) to solve the puzzle.

The program does not give a response if the child gets it right or wrong. Nor does it give feedback to the teacher. The program makes the puzzle harder by moving the snake for a long time thus making it harder to remember the solution and also by having a more complex number square. There is no way to select which number square to use. The program is brightly coloured and 'Monty' is entertaining, so hopefully maintaining the children's attention for longer.

Another program I looked at was 'Dive into Maths One'. My first impression of this program was that it is colourful and bright; it has clear instructions that are informative and concise. The program contains different sections, which include the ability to draw pictographs, bar graphs both vertically and horizontally; it has sections on length, in centimetres, more or less than, heavy or light and also some problem solving.

The program was introduced with some characters as part of a story. The story was kept going throughout the whole program. It was used to help explain when a child got it wrong. The story also provided an incentive to progress through the tasks.

2.1 Drawbacks.

At times the program was slow and time consuming and the characters spoke with strange accents, which at times made them hard to understand. The program took some time to complete tasks; this would have made it difficult for a teacher to include this program within a lesson, as it would have taken

more than the 30 minutes working time. The program is not specific enough just to use it to explore a method covered in the lesson.

The last application I looked at was ‘Mad About Maths Number’. This program has a heavy story line. It has an opening introduction, which introduces four different characters. The duration of the introduction is greater than 4 minutes. One child I observed using this application started clicking the mouse after about 5 seconds and when he got no response he reported that his computer had broken. Once through the introduction the characters go off in different directions and the user has to go and help each one in turn. It gave no feedback to the child, apart from general comments embedded in the story.

This type of program would be ideal for a child to use at home where the child can treat it as a game and take advantage of using available adult help.

This program, as with the others, I have looked at gave no explanation of why a child had got an answer wrong. Neither have they suggested strategies to help them answer the question right the second time around.

There are two ways in which I could design a program to achieve my brief (at the top of the page). I could design an application or I could design a website. An application would be a compact program that would run in its own window and be totally self contained. A website would be opened using an applicable

browser e.g. Microsoft Internet Explorer, or Netscape Navigator and would run within a normal window.

The advantage of using a website over an application is because the Internet is so widely available that most machines are made compatible. This means a website will run on any machine. It would also give an opportunity to publish the page on the Internet therefore making it easily available to schools across the world. By using a familiar browser the teacher will find it easy to work with and the children will start to use the skills needed in using the browser and the Internet.

There are many different programming languages in which a website could be written. I will just assess the main ones, Java, JavaScript and HTML.
HTML has a few different languages associated will it namely XHTML and XML these are all different combinations of HTML will different properties so I will treat them as one language.

Within Java it is possible to make applications, frames, applets and objects. Applets are implemented into web pages by using HTML to call the Applet to the page. The Applet is embedded into the page. It is for this reason that HTML and Java Applets work well together. Java and JavaScript provide an interactivity to a website. So the HTML is the basis to the page and either Java or JavaScript provides something to do. Java is not the same as JavaScript, Java is used to write whole programs within a separate package

where as JavaScript is a way of using scripting to implement interactivity in to a website.

In conclusion I believe that the experience obtained from observing teachers implementing the National Numeracy Strategy in the schools I have been fortunate to visit; the discussions held with teachers; the trialling of existing software packages and websites have contributed to the development of my ideas for my own mathematics program. I look forward to developing my ideas further and, with the kind permission of the head teachers of schools in Key Stage One, trialling them with children who will be the program's greatest critics.

3 Specification

Describes the problem and what is wrong at the moment. Gives a feasibility study and my objectives for the project. It gives the results and findings of an initial investigation into the problem.

3.1 The Problem

The purpose of this project is to design an interactive website to help in the teaching of maths at Key Stage One. I will look at a particular part of the Numeracy Curriculum, addition of nine.

3.2 Clarification of The Problem

I plan to design and develop it for Year Two, Key Stage One. I will use a special method of ‘addition of nine’ where you add ten to a starting number and then you subtract one from the result. For example if you start with three. You would add ten, which would make thirteen and then you would subtract one, to get to twelve.

Children have been taught the mathematical skills for subtraction and addition earlier in the year. They have practised the method for adding on ten. My program would make the children combine these existing skills so that they could add nine more efficiently. Once the children have learnt this method they will be able to extend it to work with eleven. It could be extended for subtraction of both these numbers. This method is one of the more advanced methods that the Key Stage One scheme has and thus tends to be taught

towards the end of Year Two. The challenge of this project is how to present this method so that a child of all abilities would be able to understand it fully.

3.3 Analysis of the problem

Initially I visited two schools, Victoria Infants School, Wellingborough and the Holy Family Primary School, Bristol.

I spent two weeks working in Victoria Infants School where I observed Year One and Year Two lessons with various different teachers and one week working in The Holy Family Primary School where I observed lessons mostly by Mrs Whiteside. I also planned and taught, (with the assistance of the class teacher,) a data-handling lesson using computers at the Holy Family Primary School.

I have found that tasks given to children need to have clear and concise instructions. They must also be progressive and the 'National Curriculum' states that the exercises must be "*differentiated*". I also found that it essential to build in extension tasks to motivate children with higher levels of ability.

It is also necessary to ensure that exercises cope with different children using a variety of strategies, for example in Mrs Kellend's data-handling class, the children had collected data from fellow class members and wanted to plot the data on to a bar graph. Some children worked their way down the list of names colouring each square according to the name whereas other children

went down the list and counted up all the same types, (for example seven bananas, and then coloured seven banana blocks in).

Mrs Kellend's data-handling lesson also showed that tasks needed to be confined and controlled. Although these children were not misbehaving they wasted a lot of time in inefficiently collecting the data. A comparison lesson by Miss Pallet on data handling showed that by restricting the lesson to either data collection or data representation (not both) worked better as the children had a clear objective and there was a defined learning outcome. This also showed me that a clear objective within an exercise must also be defined and clear ideas of what skills the child is going to use within the exercise is also essential.

During Miss Pallet's data-handling lesson I notice that children who had a lower level of literacy than their level of Numeracy experienced difficulty in understanding the questions. When the questions were explained to them they could answer them easily.

The tasks I design must, therefore, be very easy to understand even for children who have a low level of literacy.

The National Curriculum states that lessons should follow a 10-minute introduction as a class, followed by 30-minute of differentiated exercises, followed by a 10-minute plenary again as a class. I plan for my software to be used after direct, whole-class teaching has taken place. There will, however,

still be a need for some introduction to enable the children in to access the program.

Whilst in Mrs Gardener's Class where number bonds were being taught I observed that a good introduction to the lesson is essential because the children became actively engaged in the tasks and worked hard and efficiently as a result of the good clear and concise introduction.

During the week I was in The Holy Family Primary School, Bristol, I spent time in Reception, Year One, Year Two and Year Three.

During a Year Three computer lesson I observed that the computer and its surroundings very easily distracted the children. It is therefore essential that the tasks engage the children and retain their attention.

In most of the computer-orientated lessons that I observed the children found the use of computer vocabulary hard to understand, I found that by using easy to remember phrases like *"The Magic Arrow Button"* to mean *'Tab'*, the children understood better. It is equally important that the children use and understand computer vocabulary. Guidelines from the Qualifications and Curriculum Authority (QCA) suggest that appropriate vocabulary is used.

In all the lessons that I observed, where computers were used, children found difficulty in entering text. This difficulty was in three main ways; firstly, and most simple to solve, is the keyboard. Some children couldn't recognise the

letters on the keyboard as they were in upper case. Replacing the keyboard with a special board engraved with lower case letters easily solves this, although this represents a cost to the school as most computers come with the standard keyboard from new. It is possible to stick letters over the top of the keys on a standard keyboard, but these deteriorate quickly. Secondly there is a difficulty associated with the font. It is important to select a suitable font so that different forms of letters like a/a and g/g don't confuse a child. I have been advised to use *'Sassoon Primary Infant'* as the letters reflect the style most commonly used and written for this age group.

I need to ensure that this does not affect any machine independence.

Thirdly, children tended to struggle with hand-to-eye co-ordination. The children often found it hard to press the correct key or move the mouse accurately. *(TALKING ABOUT INFORMATION COMMUNICATIONS TECHNOLOGY, A GUIDE FOR TEACHERS. 1999. Pg 65*) highlights this as criteria for evaluating software, stating; *"how precise does the mouse control need to be?"* This needs to be taken into account when programming, the buttons must be big enough so that a child with poor mouse control can click on them. The program must also be able run slowly enough for them to be able to maintain control of the program.

Similarly, the children did not think to use capital letters and full stops. (FOX et al. 2000. Pg 64.) undertook a case study of children working on a mathematics software package. Within one of these case studies a researcher asks one of the children *"why don't you use strategies when you*

work at the computer?" the child replied, *"you use these when you're working in your books. We don't have things like paper and pencil or a ruler by the computer."* Clearly children don't associate typing as handwriting and thus they did not think to include punctuation. Children at Year Two, Key Stage One find punctuation and grammar very difficult at this early stage of their development in literacy. Ideally I would want the child to have little or no writing to do. The program that I will design will test numeracy skills and will therefore require minimal literacy skills.

During Mrs Whiteside's addition and subtraction lesson a puppet called 'Fussy Peg' was used to help teach the children. This puppet helped keep the children's attention. I would like to include in my program a character whose purpose is to help, give information or feedback to the child. Animations also make it fun for the child and make it easier for them to learn. Also (CICCO et al. 1999. Pg 129.) states that *"the addition of moving pictures – animation – to a Website can greatly enhance its presentation"*.

I perceived while evaluating "Dive into Maths One" and "Mad about maths Numbers" that the animations were long and tedious so I need to make any animations quantitative.

During Sister Mary's adding and subtraction lesson I noticed how much the children responded to music. It kept them really interested and on task. As a result of my observations of this lesson there is a need to include music, if possible, within my program so that it gives an added element in to the

program retaining the children's interest and makes it multi-sensory. (TALKING ABOUT INFORMATION COMMUNICATIONS TECHNOLOGY, A GUIDE FOR TEACHERS. 1999. Pg 65) also points out that how long a piece of software holds a child's "attention, interest and motivation" is a key factor in a successful piece of software.

I have found that the children wasted a lot of time on the computer. I think this was because it is like a new toy, a game and this makes it hard for them to work with a computer. At this level, children are being shown that a computer is not only something to play on but also a tool. This goes back to the fundamental reason why we use computers as (FOX et al. 2000. Pg 54) says "…to do things quickly which would have otherwise be too time consuming…"

3.4 What is wrong with the present system?

At the present time the method of addition of nine is taught, in the four schools I visited, by the use of worksheets. There is no use of any IT equipment or skills. These worksheets can be unclear and are not standardised throughout the years, e.g. each new 'Year Two' is using different worksheets. Another problem associated with the use of worksheets is that the teacher has to produce differentiated exercises to meet the National Curriculum standards. It results in the teacher having to produce one than more worksheet, to meet the needs of the class.

3.5 Feasibility Study

3.5.1 Technical Feasibility

Is it technically possible for me to build a website to meet the aim set out in 1.1?

In answer to this question I considered these different solutions:

- Designing a website solely using HTML programming.
- Using a piece of website designing software, e.g. Adobe Go Live, to build the site.
- Using HTML with embedded JavaScript to provide the interactivity.
- Using HTML with an embedded Java applet to provide the interactivity.

All of these could equally work and each has their advantages and disadvantages. I have discussed in detail the choice I have made and why in 5.1.

Is it possible to design a program that meets with the current content requirements of the National Curriculum?

The National Curriculum has strict guidelines for what the children should learn. It should be possible to design a program that meets these requirements as long as this is kept in mind.

Is it viable to fit the program into the time constraints of the National Numeracy Strategy?

A child would have to be able to complete my program within the 30 minutes in order to fit the guidelines of the National Numeracy Strategy. It would be hard to make a program that:

a) Lasted for thirty minutes and,

b) Kept the interest of a six or seven year old child for thirty minutes. It would have to be the case that if the child had not finished when the thirty minutes had concluded, then the program would have to save the progress of the child. Making the program have levels and rewarding their achievement would help keep the attention of the child.

Are there facilities for trialling the program?

Westglade Infant School, Nottingham has very kindly let me trial my program within the school. This does mean that there are certain time constraints because trials of the program must be at an appropriate time for both the school and children.

3.5.2 Social Feasibility

How would the use of this program affect the teacher and support staff?

The use of this program could affect them differently according to the school's resources and how the teacher chose to run the lesson. For example if the school had a large computer suite in which a class lesson could be taken, then a teacher could choose to teach the whole class at once. This would need a different skill set compared with teaching in the classroom. Another example, if the teacher only wished one group to go

on the computers. If this was in another room then the children would have to be supervised and this may mean less support in the classroom. Whatever the resources of the school the teacher would have to understand the program in able to teach a child to use it.

How would this program affect the children?

Using a computer to do what they could have done on paper gives the children valuable extra time on ICT skills. It also helps keep them motivated. Children find a computer interesting and are keen to work on one. However, my observations described in page 16 showed that 'children were very easily distracted by the computer and it surroundings'. Therefore it is important to design the program to motivate and retain interest and not distract the children.

Will there need to be training for the adults and/or children?

The teacher or support staff that will be introducing the children to the program will need to know how to work it. If the program is easy to understand then less introduction by the adult will have to be undertaken and the more work the child can do.

3.5.3 Economic Feasibility

Are there adequate funds within the schools to provide relevant hardware and software requirements of the program?

A website would run in a browser, e.g. Microsoft Internet Explorer. This comes as standard with most new machines. All schools with an Internet

connection would have Internet Explorer so the school would not have to buy new software. Again as a browser comes as standard, it would not entail purchasing new hardware.

Are there enough funds to provide any extra staff support, if necessary?

If, as in a case described earlier, where a member of staff would be required to supervise a group working on the computers then would the school have to employ an additional member of staff to cover them or just have one less in the classroom? Teachers are able to change the focus of their support staff working alongside them but are not able to have extra staff, as this needs long-term decision-making by the school. I am not in a position to answer yes or no, it would depend on the situation, the children, the school and many more things. The school would have to balance the benefits of the program against the extra cost of staff. I can only make my program fit the criteria as closely as possible so that the school thinks that it would be beneficial to use my program.

Are there going to be any running costs?

The program would not use any peripheral hardware e.g. a printer that would cost money. The only running costs would be maintaining the computer, which is not solely the responsibility of my program.

3.6 Questionnaire

I carried out a questionnaire to teachers to ask their opinions about my initial ideas and plans (see 11.5). These are my results.

Half the teachers that answered the questionnaire said that on average a child within their class uses a computer for numeracy lessons once a month. The other half said that a child uses a computer once a week.

All of the teachers said that on average a child uses a computer for any part of their schooling, once a month.

All of the teachers thought that animation is a useful medium to help children answer questions and explain instructions.

All of the teachers felt that music would help in keeping the attention of children.

An interactive ‘help’ was thought by all teachers, to be useful.

There was a variation between schools about how many children thoroughly understand the concept of addition of nine. Half the teachers felt that ten percent understood and the other half felt that it was around twenty percent.

ICT resources varied from school to school and this influenced the teacher’s response to the questionnaire. One school described their ICT resources as

having some computers in each classroom, with a computer suite as well. One school had a small computer suite per two classrooms and one school had a large main computer suite, with one computer in each classroom.

The level of adult supervisors per class also fluctuated greatly depending on the school and the class. One teacher said one or less, one teacher said one or two, two teachers said two and two teachers said two to four.

All teachers felt that a medium to high level of adult supervision would be needed for a child to work on this type of program. Also all teachers felt that this level of supervision was ideal.

None of the teachers that were surveyed knew what ‘drill and practise’ was and so didn’t answer the corresponding question.

Two out of the three schools surveyed said they had a computer club.

All of the teachers felt that around half of their children had the use of computers at home.

All the teachers felt that a program, as described, has a place within the classroom.

The mathematical software that is used by the schools I surveyed is:

Sorting and counting

Number Crew

Connections

Number Train

Number Plane

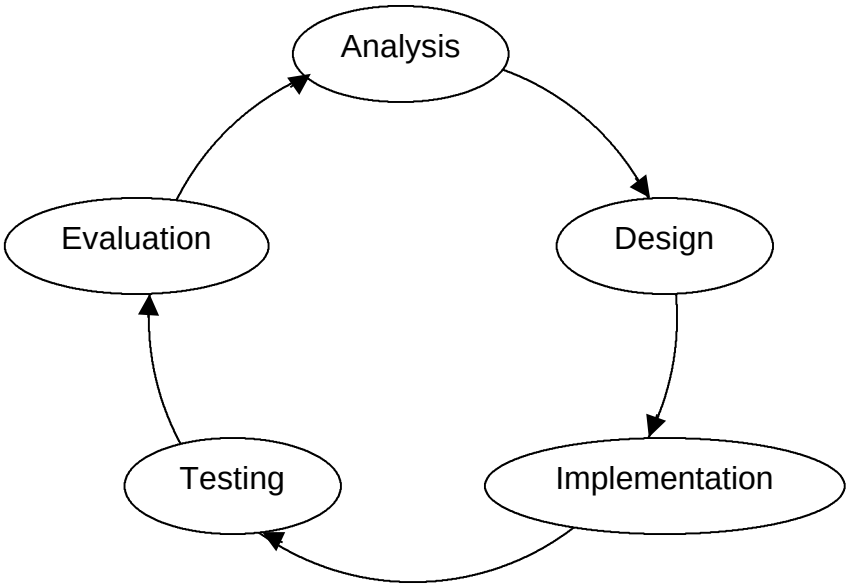
Black Cat – number box 2

Monty

4 Methodology

This chapter includes a description of the System Life Cycle, appropriate psuedo code, a system flowchart, an Event Table and corresponding label GUI (Graphical User Interface) diagram.

4.1 The Systems Life Cycle



4.1.1 Analysis

In the analysis the problem is identified and a small study is performed to decide if the problem is worth investigating. Once it has been decided that something will be gained if a study is undertaken a more detailed study is undertaken to determine the practicality of the possible study. This feasibility study would look at three main areas - Technical feasibility, Social feasibility and Economic feasibility

Technical feasibility looks at whether it would be viable technically to solve the problem. It looks at whether it is possible to solve all of the objectives and

aims or just some of them. Social feasibility is concerned with the effect on the people involved. Economic Feasibility is about the cost of any possible changes that might be made. It is also concerned with on going cost e.g. running costs or maintenance costs. It looks into possible economic benefits as well. This has been explored in 3 (Specification).

Upon completion of the feasibility study the designer/designers should have a clear and concise set of boundaries concerning human interaction, technical requirements and cost for the possible solution.

4.1.2 Design

In this section the process of the design will be looked at. Things that will need to be taken into consideration are the output, input, files, processing, security, testing strategies and the hardware requirements. A large part of the input and output consideration are the GUI.

A good graphical user interface is an essential part of a successful system. In designing a GUI the designer must take in to consideration four important points, as explained in (A LEVEL COMPUTING. 1997. pg 83).

- Who is going to use the finished system?
- What tasks the computer is going to perform?
- The environment, in which the computer will be used, e.g. is it hazardous or do you need special requirements for the user to utilize the computer.
- What is technologically possible?

“In particular, careful screen design can make a huge difference to the usability of a system. When designing an input screen, the following points should be borne in mind:

- The display should be given a title to identify it.
- It should not be too cluttered. Spaces and blanks are important.
- It should indicate the size and format of data entry in each field; i.e. don’t just put ‘Date: _____’
- Items should be put into a logical sequence to assist the user.
- Colour should be carefully used.
- Default values should be written in where possible.
- Help facilities should be provided where necessary.
- The user should be able to go back and correct entries before they are accepted.” (A LEVEL COMPUTING. 1997. pg 83)

At this stage the designer/designers will also design the program. This involves writing detailed specifications, using structure diagrams e.g. flowcharts, and the writing of psuedo code. Psuedo code is a list of clear statements made in English in the order the program would run it. It is a brief description of the processes the program/system will go through in roughly that order. It is useful to note that pseudo code does not have to follow the grammar rules of English, as long as the statement make sense to the reader.

Another stage in the design process is prototyping. Prototyping is where a small working model is built in order to put it the final situation. This allows the

designer to test initial stages of the design process. It also gives a chance for the user to makes suggestions. Prototypes can either be enhanced into the final system or discarded.

The final task for the designer is to design how the system is going to be tested. The testing will take place later on in the systems life cycle, but the designer needs to set the conventions for the testing so that consistency is kept at that time.

There are three types of testing top-down, bottom-up and ends-in. The first two are well known and well documented. Top-down is a method by where the programmer sets the global skeleton structure of the program and then tests this. Next the programmer would 'fill in' the details in some sections and then test it again and so on until the final program was competed and tested. The advantage of this method is that the programmer is always looking at the whole problem in one go, so all the parts of the problem are related to each other.

Bottom-up is a method by where the programmer splits the overall problem in many different and smaller parts. These parts are then programmed and tested separately, with the end task being to put it all together. The advantage of this method is you know that each individual part works fully and is tested fully. One disadvantage of this method is that it needs careful planning so that all the separate parts fit together at the end.

The ends-in method has come about recently from professional programmers and meets the need to find quicker ways of finishing a project and saving time. This method basically means parts of both the previous methods combined. A programmer might start with skeleton frame, but fill in and test important sections as and when they are determined. This also leave the scope from the use of previous work, so instead of replicating a section of code already produced and tested from another project, the programmer could just adapt it and implement it from the start.

4.1.3 Implementation

Implementation deals with how the final solution will fit into the existing system. In the context of replacing a computer system of a business the cost of rewiring the building, any new furniture that would needed and hardware installation would all need to be taken into consideration. In the case of this project whether the computers in schools would be able to run the program. Questions that would need to be answered relate to the amount of space used, the speed, and the ease of opening and closing the program.

4.1.4 Testing

This would be putting the planned testing into action. Depending on which style of testing the designer chose would determine how the testing was to be performed. The testing would need to test all possible forms of input and output. It would need to test the security of the program, whether the code would be easily accessed and changed, making the program not work. It

would have to test thoroughly everything that could possibly occur even to the extent of power cuts and hardware failing.

4.1.5 Evaluation

The evaluation is to review the final production. It looks at whether the final solution actually solved the aims and objectives set earlier on. This need to find out if the final solution is fulfilling expectations.

The Systems Life Cycle is never ending. Each time a cycle is completed it begins again. The points that are raised in the evaluation are then analysed, re-designed, re-implemented, re-tested and re-evaluated. These processes can take many years and with new hardware and software being developed constantly the cycle to improve the system is indefinite.

4.2 Psuedo Code

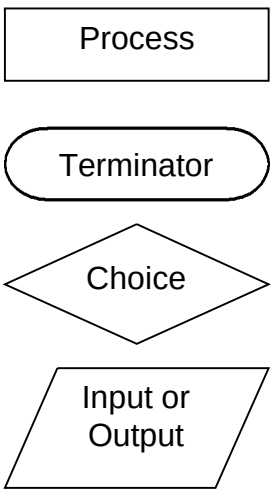
```
START
        Initialises and adds the screen objects
        Set number of questions
        Declare variables
        Calculate some unique random integers & then Stores number.
        When enter is pressed in textbox1
          If not all quest are completed & it is textBox1
              if answer is right
                    Say well done
                    Move to next 2nd part of the question
                    Stores the score
              Else
                Say sorry its wrong try again

        When enter is pressed in textBox2
              If not all quest are completed & it is textBox2
              if answer is right
                    If last quest
                        Well done you've completed all the questions
                    Else
```

Say well done
Move to next question
Else
Say sorry its wrong try again
When setLevel function is called externally level is set, corresponding range values , object on screen are set
When BgColor is called externally the background colour is set
When setFileName is called externally the filename is set
Writes the score to a file as a string
END

4.3 A System Flowchart

A flowchart is a way of representing the flow of data through a system or program diagrammatically. The System flowchart is made of symbols that each has its own meaning. These symbols are arranged to represent the system and labels are put on them.

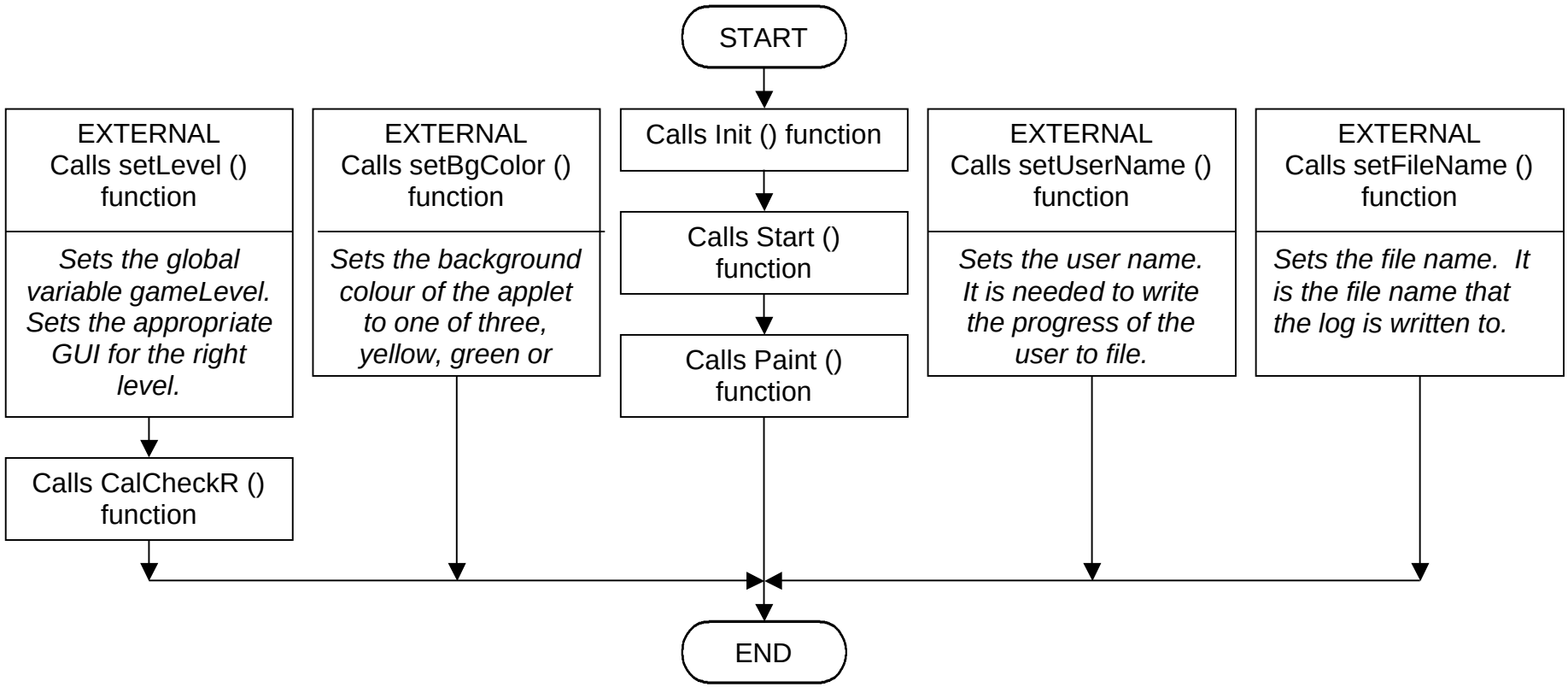
The main symbols of the flowchart are:



4.3.1 System Flowchart for this problem

To draw a system flowchart for my program is difficult because nearly the whole program is event driven. By event driven I mean that when the user does something, e.g. clicks the mouse or presses a key on the keyboard, this

is called an event. When an event happens it triggers a piece of code within the program so that it can react. I have chosen to write an event driven program rather than a program that flows from start to finish because it gives the program more scope to respond to the child's actions.



It can be seen from the flowchart that certain functions within the flowchart have no start. This is because these functions are event triggered so it doesn't mean they will at start a particular point.

4.4 An Event Table

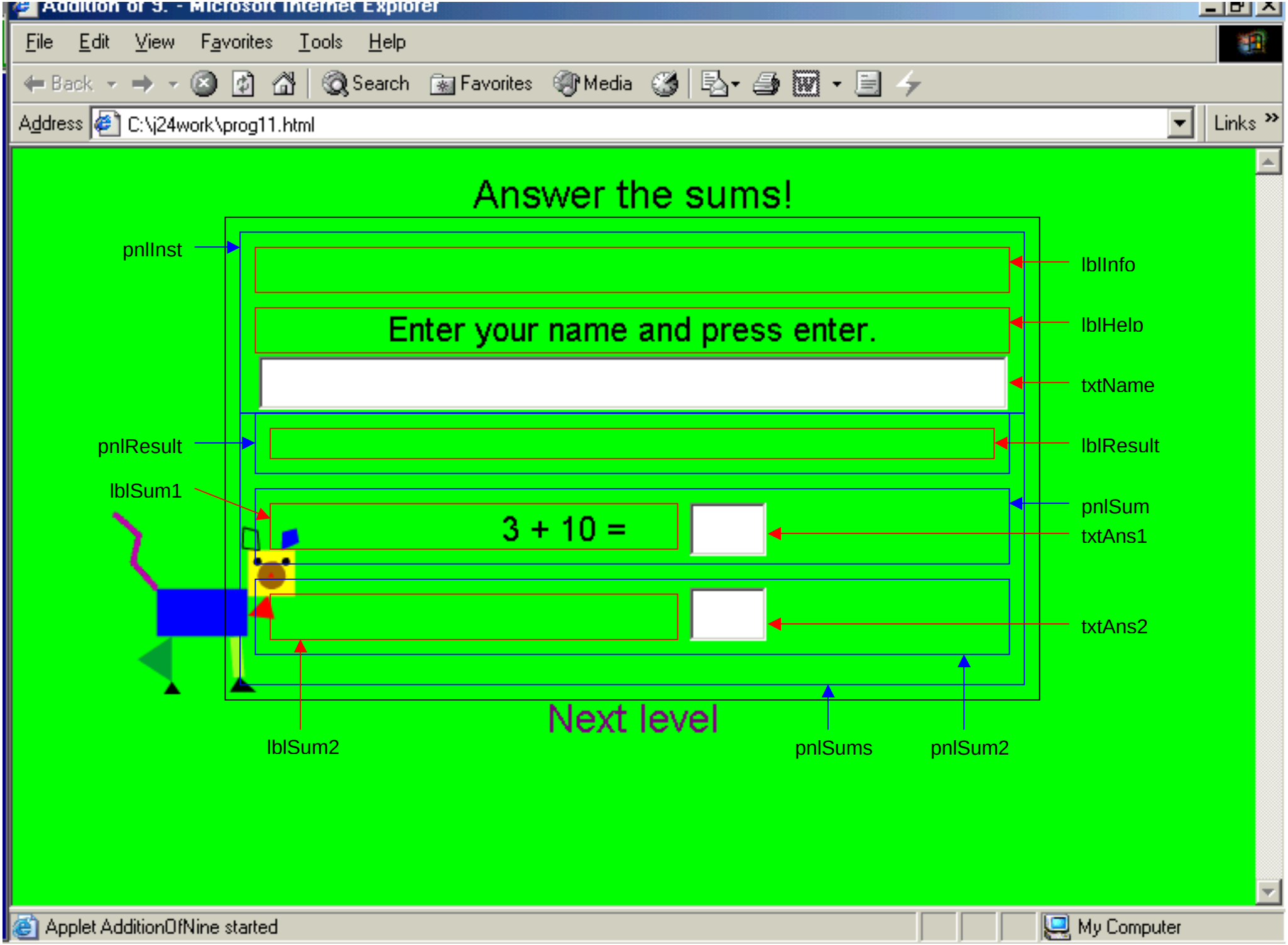
When a large part or parts of a program/system flowchart is event driven it can be worthwhile representing the events in an Event Table. An Event Table describes each individual action. An Event Table describes the method by which the event is triggered, the conditions of the event, the action that is taken and the links with the rest of the program/system.

An Event Table for my program follows. Note: variable names are used throughout the System flowchart and the Event Table. A fully annotated diagram follows showing clearly each variable and its relative position on the screen.

Event	Condition	Action	Call Function
butRestart – 'on click'	if checkIfAllRight=true	Makes a string – the 'User' has got all their questions correct first time.	checkIfAllRight()
	else	Makes a string – the 'User' has got some wrong.	
	try	Making as new data buffered file output stream. Write the string. Flushes the stream. Closes the stream.	
	catch SecurityException	Sends error to lblInfo.	
	catch IOException	Sends error to lblInfo.	
butHelp – 'on click'		Asks how can I help you.	
ButNextLevel – 'on click'		Sets count=0. Calls setLevel passing in gameLevel +1.	setLevel()

txtAns1 – 'on complete'	if it's the 1st level or 2nd level & not all the questions have been completed & the answer is correct.	Get the answer. Stores the score.	checkIfAllRight()
	if it's the last question	Clears everything, check to see if all right then prints message accordingly.	
	else	Says well done and moves on the next question.	
	else	Says sorry you've got it wrong, try again. Resets the question.	
	if it's the 3rd level or 4th level & not all the questions have been completed & the answer is correct.	Get the answer. Stores the score.	
	if it's the last question	Clears everything, check to see if all right then prints message accordingly.	
	else	Says well done and moves on the next part of the question.	
	else	Says sorry you've got it wrong, try again. Resets the question.	
txtAns2 – 'on complete'	if not all the questions have been completed & the answer is correct.	Get the answer. Stores the score.	checkIfAllRight()
	if it's the last question	Clears everything, check to see if all right then prints message accordingly.	
	else	Says well done and moves on the next question.	
	else	Says sorry you've got it wrong, try again. Resets the question.	

txtName – ‘on complete’		setUserName with the contents of txtName. Hides the text box. Gives the initial message. Focuses on txtAns1	setUserName


Addition of 9. - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address C:\j24work\prog11.html
Links
Answer the sums!
pnlInst
lblInfo
Enter your name and press enter.
lblHelp
txtName
pnlResult
lblResult
lblSum1
pnlSum
3 + 10 =
txtAns1
txtAns2
Next level
lblSum2
pnlSums
pnlSum2
Applet AdditionOfNine started
My Computer

5 Design

This chapter includes a discussion on suitable means to design a website, browsers and plug-ins, animation techniques, writing to local disk and the design of the GUI.

5.1 HTML

The HTML language works by a method of tags. A tag is a way of classifying something. For example the title of the page would use the title tag, *<title>Tom's web site</title>*. Tags usually come in two parts, an open tag and a close tag and are contained within greater than (<) and less than (>) symbols. The close tag is usually identical to the opening tag except that it proceeded by a slash (/). There are a great many tags and they mean that a web page can have things like tables, frames, pictures, text, titles, headers, different sized fonts, different styles of fonts and lists. It is these tags that enable the majority of what you can see on a website. An attribute of HTML is that the tags are not case sensitive, so title, TITLE and TitLE all mean the same. This is a disputed point about HTML. It has the advantage that it makes it easier for beginner programmer to write HTML code, but it gets them into bad practices. It also causes problems when HTML wants to interface with other programming languages. This is due to a lack of consistency with the variable names.

5.2 Java

Java was designed by a Sun Microsystems developer James Gosling. He was unhappy whist working on a c++ project, so he designed a new language

to overcome his problems. It is a programming language that needs a highly organised approach and in the recent years it has become a big part of the Internet.

As the Java website says:

> *"The Java programming language is robust and versatile, enabling developers to:*
>
> - *Write software on one platform and run it on another.*
> - *Create programs to run within a web browser.*
> - *Develop server-side applications for online forums, stores, polls, processing HTML forms, and more.*
> - *Write applications for cell phones, two-way pagers, and other consumer devices."* (SUN MICROSYSTEMS INC. 2003)

Within the scope of this project I am only interested in the abilities of Java with respect to Internet but it interesting to note how many possible applications Java has.

As previously mentioned in 1.5, in order to use a Java program with a browser, a special type of Java program is built, called an applet. An applet follows a basic layout of:

Only initialise and paint are needed to make an applet work. Each part is called at different times. Initialise is called when the web page first loads up. Start is called when the applet is started. Paint is called when something needs to be put on the screen. It essentially 'paints' what is needed on to the screen. Stop is called when the applet is stopped and Destroy is called when the web page is closed.

The advantages of using Java in order to get a level of interactivity on a web page are that Java is an Object-Orientated language. Object-Orientated is where the data, and the methods that act upon that data, are grouped together. The advantage of this is that any thing in real life can be classed as an object. An object has properties and behaviours. Properties are described by using data, and behaviours are described by using methods. An Object-Orientated program provides greater flexibility, modularity and reusability. Another advantage of Java is that it is a 'program once, run anywhere' language, so it is very portable. Java also has built in facilities to create multimedia. By multimedia I mean Sounds, Images, Graphics and Video. Multimedia demands incredible computing power, and an advanced language to manipulate them.

The main disadvantage of Java is its lack of speed, due to the way that Java code is translated into the code that is run on the computer. When choosing to program in Java it is a trade-off between having the advanced capabilities of Java and it doing it slowly.

A Java applet is run from the HTML code by the use of two possible tags, one is the <applet> tag and the other is the <object> tag. The <applet> tag is depreciated in favour for the <object> tag, which can be applied for a great many different types of files not just applets (Note these tags are for use within Microsoft Internet Explorer not Netscape Navigator. See 5.5 for explanation).

5.3 JavaScript

As stated previously JavaScript is written alongside HTML, contained within <script> tags. It is also a language that is written in text-only files and this has the same advantages as HTML. JavaScript also is non-case sensitive, which again causes problem with interfacing between languages.

5.4 Java, HTML and JavaScript compared

I looked at each of these possibilities in 3.4.1 in order to decide the best method to design my website. Taking the first point, HTML has a very low level of interactivity. I feel I would need more interactivity within my program. This ruled out this option.

Secondly looking at pieces of web designing software each one has a different and highly personalised way of writing the same page. Some of these applications use a method called CSS (Cascading Style Sheets) by where one or many ‘Styles’ from a external file can be applied to a whole website. This external file has to be linked to the website. This would entail a

great deal of superfluous code, as I am planning my website to be relatively small.

Thirdly, I looked into a combination of HTML and JavaScript. The problems I found with this method was that although the JavaScript code worked well simultaneously with the HTML, the JavaScript code didn't allow a complex, powerful interface to be designed coherently.

This is where Java came into its element. Java is very powerful. It also has the advantage that the source code is hidden so it cannot be changed accidentally. Java applets are designed for use within a web page. Applets have a low size when complied into the class file, which is the file run from the browser. A disadvantage of a Java applet is that it doesn't interface well with HTML.

It was with these findings that I decided that I would use a Java applet embedded into HTML. To solve the problem of interfacing between Java and HTML I chose to use JavaScript to communicate with Java.

5.5 Browsers and Plugins

The two well-known browsers are Netscape Navigator and Microsoft Internet Explorer (IE). Browsers allow extensions to their basic methods by the use of Plug-ins. A Plug-in is a software program that enhances the capabilities of a browser. A Plug-in is needed to run up-to-date multimedia files e.g. Java or Flash files.

Different companies make these two browsers. This can cause some problems when the browsers are handling other languages such as Java or Flash. On some occasions a different plugin and code is required for each browser. This is the case with the use of Flash files (see 5.6) when using IE the <object> tag is used but when using Netscape Navigator the <embed> tag has to be used. This means that in order for a website to work with both browsers both tags would have to be used.

5.6 Animation

Throughout my observations and the questionnaire I have found that having animations would greatly enhance my program. However, they would have to be quantitative, as described on page 18. Animation is a series of pictures that are played sequentially at speed that gives the illusion of contents of the pictures moving.

It would be possible to use Java or JavaScript to animate some pictures although this mean having each picture in each frame downloaded on to web site. This method would take up a lot of space.

An alternative method would be to use a piece of animation software, for example Adobe LiveMotion. The advantage of using LiveMotion is that only one animation file is needed instead of many picture files. This saves significantly on the amount of files to download on the website, so it keeps the download time low and therefore reduces cost of Internet connection time.

As described earlier, LiveMotion is a timeline driven program, where certain properties of an object are set to different states according to the place in the timeline. LiveMotion allows you to set the start and finish positions of an object. It then calculates the trajectory between the two. This is very powerful and means that many different intermediate positions can also be set giving highly complex movements.

A good example of the capabilities of LiveMotion is 'EG3.swf' contained on the attached CD.

5.7 Writing to local disk

Another major observation is that a teacher would like to have a record of a child's work. I therefore plan to write a log file, which contains the names of the children who had worked on the program, what they had got right and wrong.

This log would have to be saved on the local hard drive of the computer that the child was working on. This could cause some problems because the security of IE will not allow an applet to write a file unless it has a signed certificate. (SRINIVAS, R. 2003) gives a good explanation of why this happens. (MUELLER, M., 2003.) also gives an example of this. This problem will be discussed in greater detail during the testing of this problem.

5.8 The Graphical User Interface (GUI)

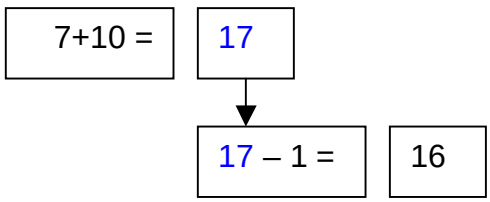
The GUI forms a major part of this project. My observations suggest that it needs to be simple, so that a child can use it. It needs to be colourful and interesting so that it keeps a child's attention. Therefore it will have quantitative animations.

I looked at the worksheets that were used with class, in one school, (see 11.4 & 11.3). I decided I would keep the same format as the worksheets because a qualified teacher has designed them.

- This was a first level of adding just ten,
- The second level of subtracting just one,
- A third level of combining the two on a range of zero to twenty and
- Finally a last level of the combination of the two on a range of zero to one hundred.

I wanted to align the boxes so that it visually looked like the first sum moved into the second.

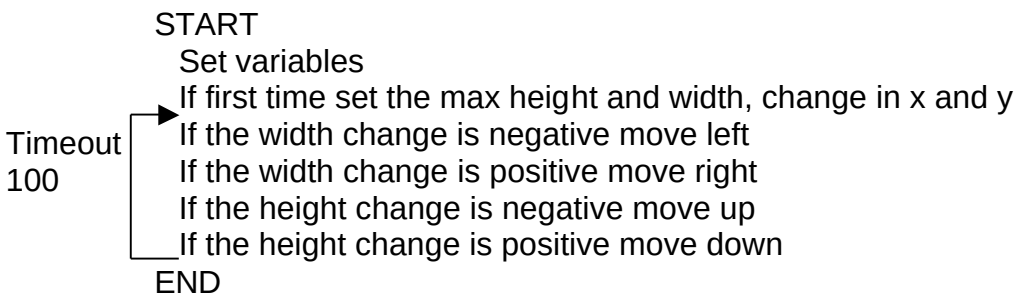
e.g.

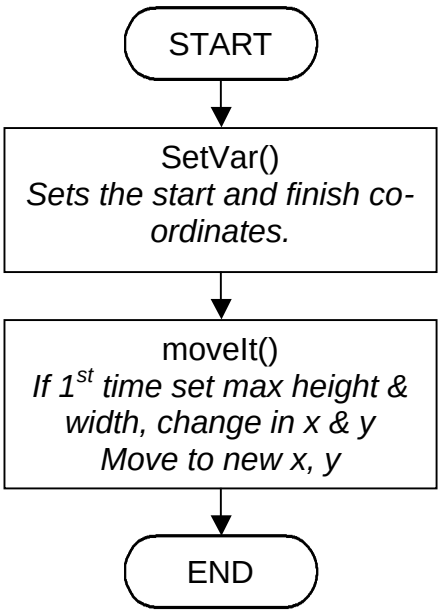


I wanted to make the font a relatively big size so that the children could see it clearly but also I did not want the GUI to be more than one screen size. The code that sets the font within the applet is *Font currentFont = new Font("Arial",*

Font.PLAIN,20); - (line 39). This line sets the font to Arial and its size to 20pt. It also sets no formatting on the text e.g. no italic, bold or underlined.

I wrote a piece of JavaScript code to move any animations around the screen (see movingObj.HTML).





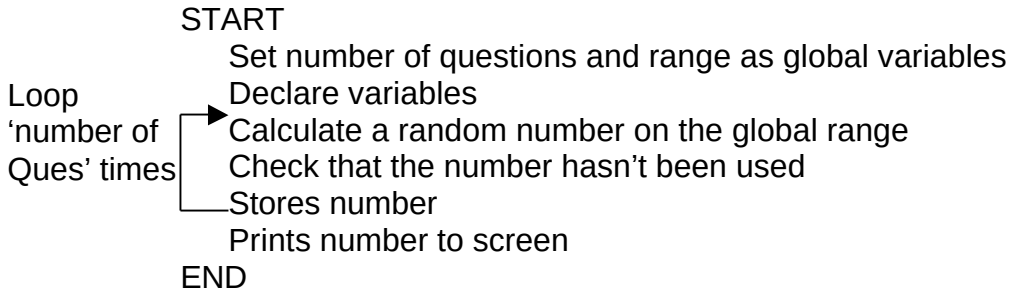
6 Development and Testing

This chapter discusses the steps leading up to the final solution, how they were developed and tested.

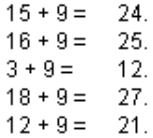
6.1 Prog2.Java

Prog2 implements the Abstract Windows Toolkit. This library contains all the components to build the GUI within an applet. It enables the use of graphics, event handling, user interface components and layout managers (see SUN MICROSYSTEMS INC., 2003. **Abstract Window Toolkit (AWT)** for more details). It also extends the JApplet class, which gives the basic of the applet (See Java_Files/Prog2.Java).

The aim of Prog2.Java is to calculate a set of 5 unique random integers.



Output



Looking at the code it is clear to see that the same code is replicated.

So instead of having two functions to ‘Calculate a random number on the global range’ and ‘Check that the number hasn’t been used’ only one is needed.

6.2 Prog2a.Java

This has taken the evaluation of prog2 and implemented it.

START
Set number of questions and range as global variables
Declare variables
Loop ‘number of Ques’ times → Calculate a random number on the global range &Check that the number hasn’t been used & then Stores number.
Prints number to screen
END

Output

```
9 + 9 =    18.
12 + 9 =   21.
3 + 9 =    12.
11 + 9 =   20.
1 + 9 =    10.
```

The function CalCheckR has solved calculating and storing the unique random integers. Next I will look at starting to design the GUI.

6.3 Prog3.Java

The aim of prog3.Java is to create a single sum question and answer GUI. It uses panels with a background colour of black as a border. This program doesn’t quite work because the sum that it displays on the screen is out of sink with the answer it is holding, although the processes work.

This program uses the action command method. This method is called when a keyboard event happens. It has been depreciated since version 1.4 of Java SDK. This means it still works at the moment but in newer versions of the language it might not be included. I have explored many different possible solutions, but I found none to be satisfactory. This will be discussed in greater detail further on in this project.

START

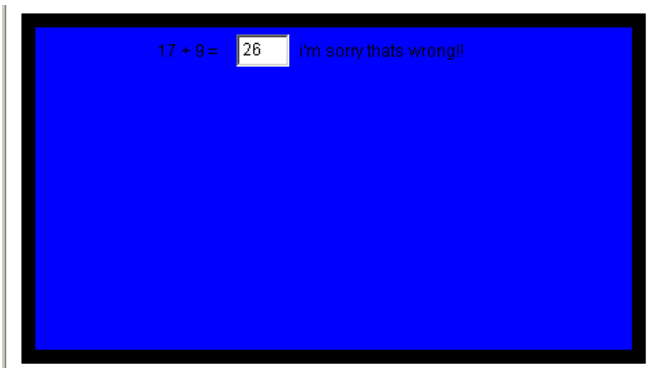
Initialises and adds the screen objects
Set number of questions and range as global variables
Declare variables
Calculate some unique random integers & then Stores number.

* When enter is pressed in textbox
  If not all quest are completed
    If answer right
      Say well done
      Move to next question
    Else
      Say sorry its wrong try again
  Else all questions are completed

END

(Note: * dictates an event is triggered.)

Output



In this case when the answer 26 is put in the program concludes that 26 is the wrong answer. On closer examination 26 was the number that 'got it right' the question before. By changing line 125 from `greeting = ""+no[count]+" + 9 = ";` to `greeting = ""+no[count-1]+" + 9 = ";` would work.

6.4 Prog5.Java

Prog5 is a working one-question program. It adds ten on to a unique random integer.

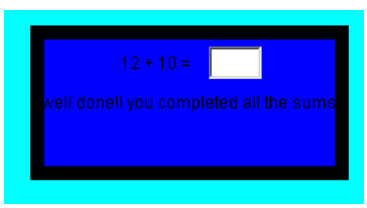
START

Initialises and adds the screen objects
Set number of questions and range as global variables
Declare variables
[Calculate some unique random integers & then Stores number.]

* When enter is pressed in textbox
    If not all quest are completed & it is textBox1
        If last quest
            Well done you've completed all the questions
        Else
            if answer is right
                Say well done
                Move to next question
            Else
                Say sorry its wrong try again

END

Output



If the user get the last question wrong the program still well done you have completed all the questions. This is because the logic is not quite right. This program checks to see if it the last question and acts on it before it checks to see if the user has got the question correct.

To rewrite the previous psuedo code

```
…
    When enter is pressed in textbox
        If not all quest are completed & it is textBox1
            if answer is right
                If last quest
                        Well done you've completed all the questions
                Else
                    Say well done
                    Move to next question
            Else
                Say sorry its wrong try again
…
```

would solve this logic problem.

6.5 Prog5a.java

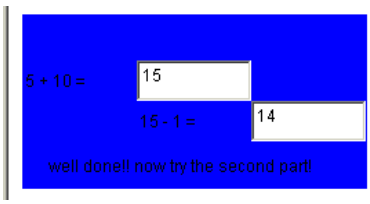
The aim of prog5a is to have two questions on the GUI. The first is a number add ten and the second, is that result subtract one. The GUI no longer will have a border made from panels but spaced out to meet the GUI design requirements (see 5.8).

```
START
        Initialises and adds the screen objects
        Set number of questions and range as global variables
        Declare variables
        [Calculate some unique random integers & then Stores number.]
```

When enter is pressed in textbox1
If not all quest are completed & it is textBox1
if answer is right
Say well done
Move to next 2nd part of the question
Else
Say sorry its wrong try again

When enter is pressed in textBox2
If not all quest are completed & it is textBox2
if answer is right
If last quest
Well done you've completed all the questions
Else
Say well done
Move to next question
Else
Say sorry its wrong try again
END

Output



The layout of the GUI is visibly like the design. The textboxes are too big and the font is too small. Both parts of the question work.

It was at this stage, where I had a working prototype I chose to go in Westglade school, Nottingham to do some initial testing. The programs I decided to take into school were Level2.HTML, Level3.HTML and level4.HTML. These three pages were combined into a website that the children could access. For the testing purposes the website was run from the local disk and not from the web.

Prior to going to the school, I took my website in the university to see if it would run the their machines. I found that the applet would not load. The university computing network has a very security measures. I attempted to research the problem before going into the school. I found the problem was due to the university computers not having the most up to date plugin for Java. This plugin is free to download from http://java.sun.com/getjava/download1.html although through security restriction within the university I did not have the privileges to download and install this update.

On arrival at the school I found that the school computers also didn't have the last plugin installed, but I was allowed to download and install the update. This set me behind schedule as the update was quite large and the school only had a standard Internet connection.

I had four different children of varying level of abilities test my website. All the children found it easy to navigate around my website although some struggled with the numeracy.

For one child in particular that was having quite a lot of difficulties with the numeracy, I used a number line to help explain the concept. She found it easier to have something that she could along and see the whole process.

Two of the children, whom had a fairly high level of ability, raced through the website and finished in minutes. Where the other tow children found it harder, one took about eight to complete all level and the other took about fifteen minutes to complete the website. This just goes to show how the website must be able to cope with a big difference of abilities.

I asked the children some questions relating to both the website and the project in general (See 11.6).

I found that three out of the four children had computers at home, this included games consoles e.g. a Playstation.

I observed that all the children liked to listen to music and like to have animations on screen.

Fifty percent of the children said that they found a problem with what the program asked them to do. When asked why they said they didn't know.

When asked about what would they do if they got stuck when working on a computer, the answers were asking a teacher or click a help button.

All three children ranged over how long they spent on a computer at home. One said he was on it every night and another said she never used it.

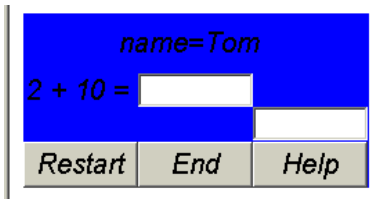
6.6 Prog6.Java, BlanksApplet.Java

In my design I spoke about interfacing Java and JavaScript. This is done to have more control over the applet.

The code is the same as Prog5a except for the line `userName=getParameter("nameOfUser");`. This line looks to where the applet is called in the HTML code and looks for a parameter called “nameOfUser”.

Within the object tag there is `<Param name="nameOfUser" value="Tom">` (See Prog6.HTML). This means that the Java applet looks into the HTML code to obtain the name of the user, in this case “Tom”. A good example of this is in (BlanksApplet.Java).

Output



A few other changes have appen in the GUI the font is bigger, the GUI now has three buttons on it. The code to make the questions work is still unchanged.

6.7 Prog6a.Java

Program Prog6a was coded in order to look a possible way around the depreciation warning. The problem is that up until this point I have been using the 'action' method. This method is called when a keyboard event happens. The action method has been superseded by the actionPerformed method. The ActionPerformed method does not work in the same way as the action method. They differ by the way of identifying the object from which the event was triggered.

Therefore the depreciated method is `public boolean action(Event e, Object o)` where the object is identified by `e.Target`. In actionPeformed, the object is an ActionEvent instead of an Event. It is not possible to use the Target option with ActionEvent so getSource is use instead. GetSource does not recognise textboxes in the same way the Target does so the logic will not work.

6.8 Prog7Work.Java

One major point that came out of my analysis is that I would like to save a log to the local disk. Prog7Work attempts to do this by writing a byte at a time.

If a question is correct then the score held would be 48, which is the byte representation of 0. Every time a question is got wrong the score number will increase. These bytes are then written to a file.

START

Initialises and adds the screen objects
Set number of questions and range as global variables
Declare variables
[Calculate some unique random integers & then Stores number.]
When enter is pressed in textbox1
If not all quest are completed & it is textBox1
if answer is right
Say well done
Move to next 2nd part of the question
Stores the score
Else
Say sorry its wrong try again

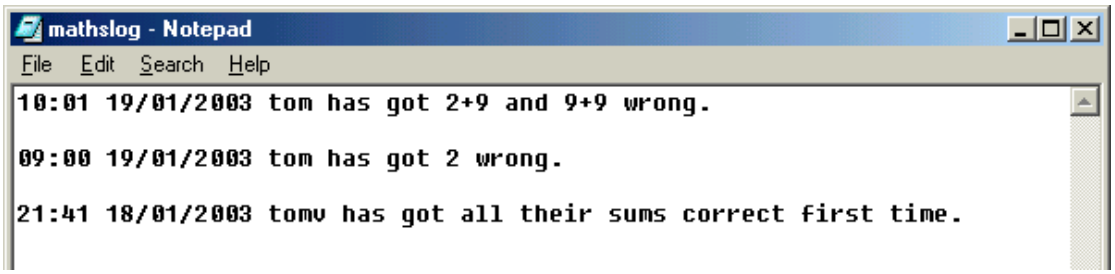
When enter is pressed in textBox2
If not all quest are completed & it is textBox2
if answer is right
If last quest
Well done you've completed all the questions
Else
Say well done
Move to next question
Else
Say sorry its wrong try again
Writes the score to a file
END

The program is also highly functionated. This is because Java script can be use to called these functions from outside of the applet (See Prog11.html). this method is also demonstrated by WriteApplet.Java.

6.9 Prog10.java

In Prog10 a main function has been added in order to give the program a dual functionality. This means it can be ran as an applet but also it can be ran as an application. I have done this for the purposes of testing. It is possible to run an application and save to local disk where as it is not possible to so when using an applet because of the security restrictions of IE.

Output



The way to overcome the security restrictions of IE is to get a certificate for the applet that allows the user to make the choice whether it should be allowed to access the user's local disk. This would be beyond the scope of this project, which is why I have tested it as an application. As an application the program writes to file successfully.

6.10 AdditionOfNine.Java

AdditionOfNine is the final program version of the project.

START
- Initialises and adds the screen objects
- Set number of questions
- Declare variables
- [Calculate some unique random integers & then Stores number.]
- *When enter is pressed in textbox1
  - If not all quest are completed & it is textBox1
    - if answer is right
      - Say well done
      - Move to next 2nd part of the question
      - Stores the score
    - Else
      - Say sorry its wrong try again

- *When enter is pressed in textBox2
  - If not all quest are completed & it is textBox2
  - if answer is right
    - If last quest
      - Well done you've completed all the questions
    - Else

```
                    Say well done
                    Move to next question
            Else
                Say sorry its wrong try again
        When setLevel function is called externally level is set, corresponding
          range values, object on screen are set
        When BgColor is called externally the background colour is set
        When setFileName is called externally the filename is set
        Writes the score to a file as a string
END
```

There is a full description of this program using the office Java documentation tool (See javadoc of AdditionOfNine.HTML\AdditionOfNine.HTML)

I tested this program again at Westglade School, Nottingham. I had six children working individually, all of varying levels of ability. I then asked then to fill in a questionnaire about the program.

I found that:

- Every child felt that they had enjoyed working on the website I had designed.
- All of the children like the animations.
- The majority of the children felt that adding nine in this was not difficult but they all thought that the website I had designed had helped them
- They all thought that the website was easy to use.

Some of their suggestions on how to improve it were:

- To have more pictures,
- To have different background colours
- To have music

- For it to be harder
- For there to a game of some sort.

The teacher feed back was also helpful:

- There was no reward after each level.
- If a child got a question wrong they weren't told the right answer after say 3 goes.
- No help function.
- Doesn't save a log file.
- It would have been nice to have an introducing animation to start with.
- It would be better to say add and subtract because the children quite often mistake the two when reading the signs.
- Press enter to move to the next level as oppose to clicking a button.
- The program didn't give any feedback on how the child had performed.
- Bigger symbols.
- To have a number line visible.
- That the program's simplicity is good.

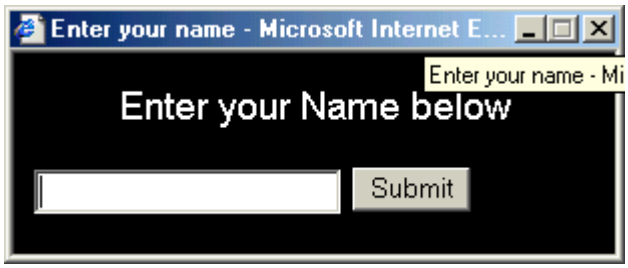
6.11 Prog2.HTML and Prog2a.HTML

I had two options in which to get the name of the user in to the program. The first was for the name to be entered in through a HTML interface, and then passed to the applet, and the second was to be entered straight into the applet.

The advantage of using HTML is that it would make the applet GUI much more simple. The disadvantage of using HTML is that it is hard to restrict exactly what is entered, where as with an applet it would be possible to format the name as much was necessary.

Prog2.HTML and Prog2a.HTML explore different ways of achieving the first possibility. When prog2.HTML loads it pops up Prog2a.HTML. If the entry into the text box is not satisfactory then an alert box is triggered. The variable is then passed back to the parent page and then through the setUserName function is set within the applet.

Prog2.HTML



If the entry into the text box is not satisfactory then an alert box is triggered.

The second possibility of achieving this, is the applet obtaining the username from inside the applet. AdditionOfNine.Java does this by

```
if(e.target == txtName)
{
setuserName(txtName.getText());
txtName.setVisible(false);
lblHelp.setText("Answer the sum below and press enter.");
txtAns1.requestFocus();
}
```

7 Discussion

This chapter discusses how well the final solution solves the problem.

7.1 The success of the program in solving the problem

The website does help teach addition of nine to Key Stage One, Year Two. It is a simple design and it is clear what do. The children that tested it all thought that it was interesting and they enjoyed working on it.

The applet has firm foundation that could easily be extended upon.

The website enables children have more interdisiplinary learning, by using ICT skills as well as numeracy skills.

7.2 The weaknesses of the program

There has been discussion throughout this project as to how much other material should accompany what is needed for the website to be useful. There has always been this balance between keeping children interested and putting to much material on the screen so that are distracted. Until near the final solution the GUI had moving animations and buttons. The final solution does not have these because the processes weren't satisfactory when the final testing was undertaken. Although it could be classed as an advantage because of the lack of distractions, it could also be classed as uninteresting.

The program doesn't provide any help to a child. Even an animated number line would provide a useful tool.

If a certificate could obtained for the applet then it would be classed as an advantage, but as the applet does not have a certificate it does not write to a file.

The GUI could more bold with special significance paid to the addition and subtraction signs.

The program could give more feedback to the child to show what and how they got a question wrong.

The website could have more breadth, with perhaps many extensions exercises to meet the needs of each individual child.

It could have more rewards, because the more the child wants to use it the more they will learn.

7.3 Suggestions to overcome the program's weaknesses

The certificate may be overcome if the information could pass from the applet to JavaScript and then saved from JavaScript. IE has less security restrictions on JavaScript than on Java.

With a little more time spent on enhancing GUI the website would become striking more interesting. I have observed in my research that music can simulate, so I closer study of the benefits of music could help.

8 Conclusions

Sell the program on it advantages to the teacher/school and the benefits to the child.

8.1 The Program

The website meets the aims set out in the introduction of this project and could a useful teaching aid within the classroom. With a little further study the website could easily compete with other software available on the market.

8.2 Possible further study

As stated previously this method of addition of nine could easily be applied to addition of eleven. It is also similar to methods of subtracting both nine and eleven. The website could be extended to include all of these variations of this method.

It could also be extended to other parts of the National Numeracy Strategy and with the possibility of including literacy and other essential parts of the national curriculum

It has the potential to become a site on the Internet that could be access by children, parents and teachers worldwide.

9 References

Software: -

2000. **Number Box 2**. Version 2.0.2. Black Cat educational software.

Monty. The Association of Teachers of Mathematics (ATM).

2001. **Maths Toolbox 1** [CD_ROM]. Rigby

2001. **Maths Toolbox 2** [CD_ROM]. Rigby

2001. **Abacus teacher's software** [CD_ROM]. Ginn.

HENLEY, J., WOODMAN, A. (consultant), 1997. **Maths Factory** [CD-ROM]. London: Harper Collins Publishers Ltd.

ECCLES, P., M^{C}KEOWN, A., 1999. **All about Number, level one** [CD-ROM]. Granada Learning LTD (SEMERC).

Books: -

NEGINO. T., SMITH. D., 2001. **JavaScript for the World Wide Web**. 4th ed. Berkeley: Peachpit Press

CASTRO, E., 1997. **HTML for the World Wide Web**. 5th ed. Berkeley: Peachpit Press

CADENHEAD, R., 2002. **Teach yourself Java 2 in 24 hours**. USA: Sams Publishing.

s.n. 1999**. Talking about Information Communications Technology, a guide for Teachers**. s.I.: Canterbury Christ Church University College.

FLANAGAN, C., 1997. **A Level Psychology**. New Edition. London: Letts.

HEATHCOT, P., BOND, K., 1997. **A Level Computing**. London: Letts Educational.

FOX, B., MONTAGUE-SMITH, A., WILKES, S., 2000. Using **ICT in Primary Mathematics – Practice and Possibilities**. London: David Fulton Publishers Ltd.

AINLEY, J., 1996. **Enriching Primary Mathematics with IT**. Clarke, S. London: Hodder & Stoughton Educational.

CICCO, E., FARMER, M., HARGRAVE, C., 1999. **Activities For The Using The Internet In Primary Schools**. Bridgend: WBC Book Manufacturers Ltd.

ASKEW, M., 1998. **Primary Mathematics – A Guide For Newly Qualified and Student Teachers**. Bristol: Hodder & Stoughton Educational.

ZUKOWSKI, J. (compilation editor), 2002. **Java 2, J2SE 1.4 Complete**. Alameda: SYBEX.

NIX, D., 2002. **Java**. [Unpublished module documents].

Government Documents or schemes: -

GREAT BRITAIN. **The National Curriculum**, 2001. DFEE.

GREAT BRITAIN. **The National Numeracy Strategy**, 2001. DFEE.

Schools: -

Victoria Primary Infants School,
Finedon Rd,
Wellingborough,
Northamptonshire.
NN8 4NT

The Holy Family Primary School,
Amberley Road
Patchway
Bristol
BS34 6BY

Millbrook Junior School,
Churchill Way
Kettering,
Northamptonshire.
NN15 8NS

Westglade Infant School
Syke Road,
Top Valley,
Nottingham,
Nottinghamshire.
NG5 9BG

Websites: -

ADOBE SYSTEMS INCORPORATED, 2003. **Adobe LiveMotion 2.0** [online]. Available at: <URL: http://www.adobe.com/products/livemotion/main.html> [Accessed 21 March 2003].

SUN MICROSYSTEMS INC., 2003. **New to Java Programming** [online], Available at: <URL: http://developer.java.sun.com/developer/onlineTraining/new2java> [Accessed 26 March 2003]

MUELLER, M., 2003. **Java World - Tutorial** [online], Available at: <URL: http://www.javaworld.com/javaworld/jw-12-2000/security/writeFile.Java> [Accessed 27 March 2003]

GRISCOM, D., 2003. **Code Signing for Java Applets** [online], Available at: <URL: http://www.suitable.com/docs/signing.html> [Accessed 27 March 2003]

IISC. 2003. **What Is JavaScript?** [online], Available at: <URL: http://sunsite.iisc.ernet.in/virlib/js/jscript/ch1.htm> [Accessed 26 March 2003]

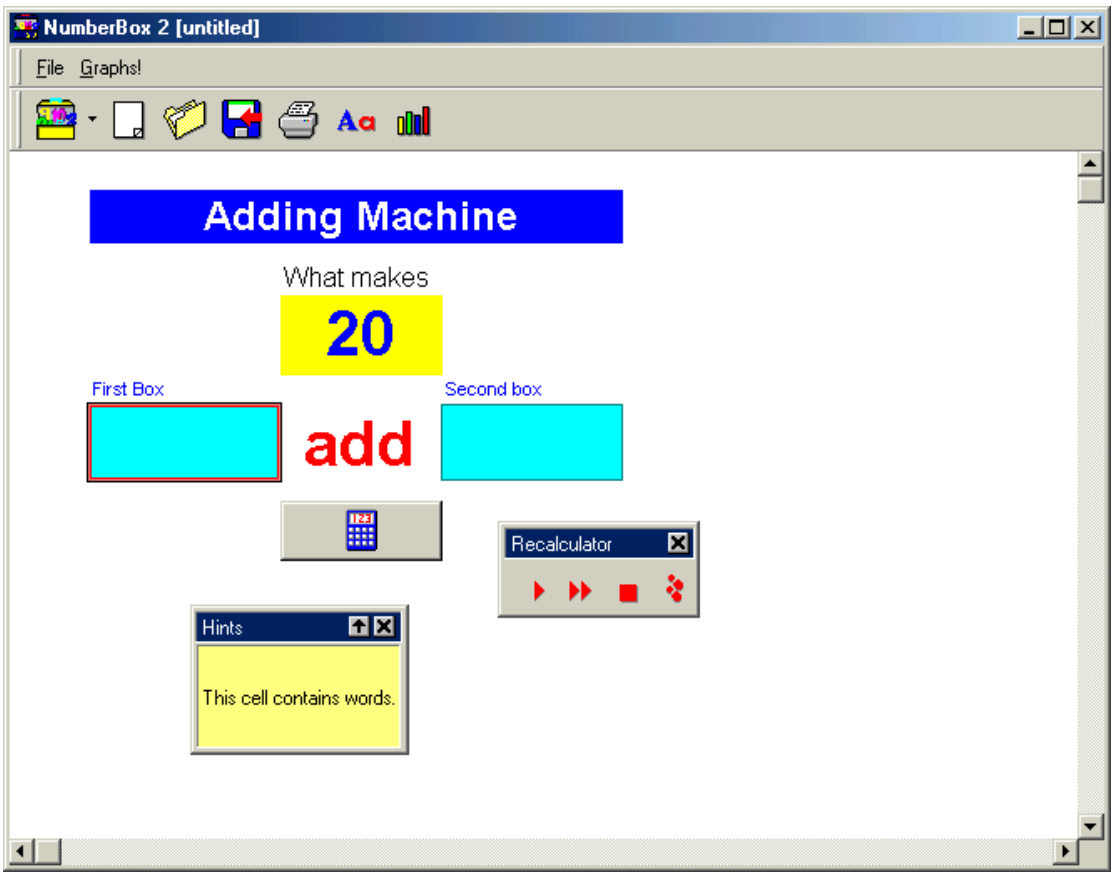
TALKAR, J., 2003. **Java to JavaScript Communication** [online], Available at: <URL: http://www.codeproject.com/jscript/javatojs.asp> [Accessed 27 March 2003]

SRINIVAS, R. 2003. **Java Web Start to the Rescue** [online], Sun Microsystems Inc., Available at: <URL: http://developer.java.sun.com/developer/technicalArticles/JavaLP/javawebstart/#code2> [Accessed 27 March 2003]

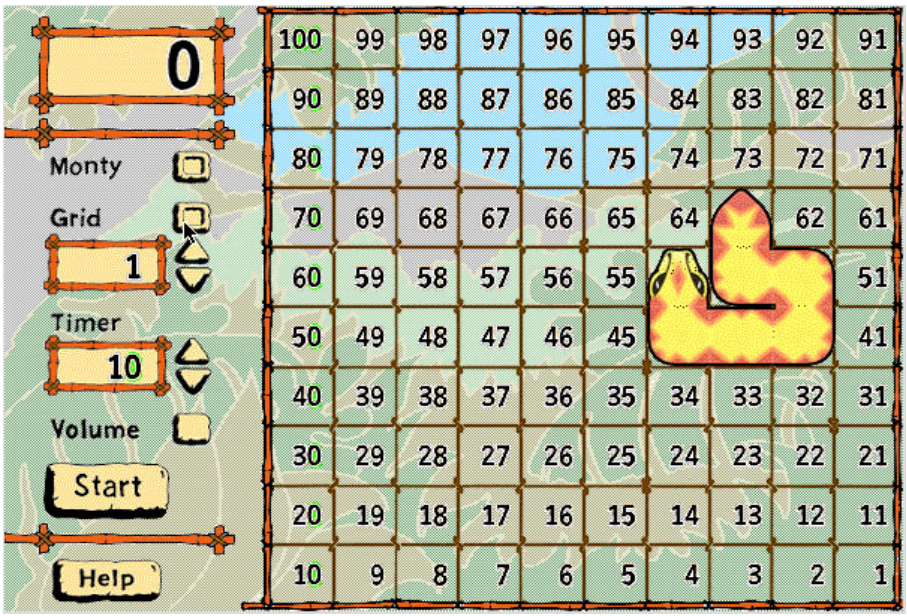
SUN MICROSYSTEMS INC., 2003. **Abstract Window Toolkit (AWT)** [online], Available at: <URL: http://java.sun.com/j2se/1.4.1/docs/guide/awt/> [Accessed 27 March 2003]

10 Appendix

10.1 (Black Cat – Number Box 2)

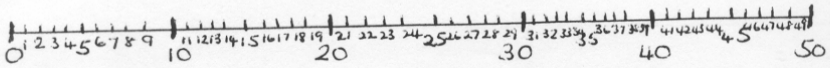


10.2 (Monty)



10.3 Worksheet example 1.

Subtract 9 or 11



Here is an example,

40 - 11 = ☐

40 - 10 = 30

30 - 1 = 29

20 - 9 = ☐

30 - 11 = ☐

50 - 11 = ☐

40 - 9 = ☐

25 - 9 = ☐

15 - 11 = ☐

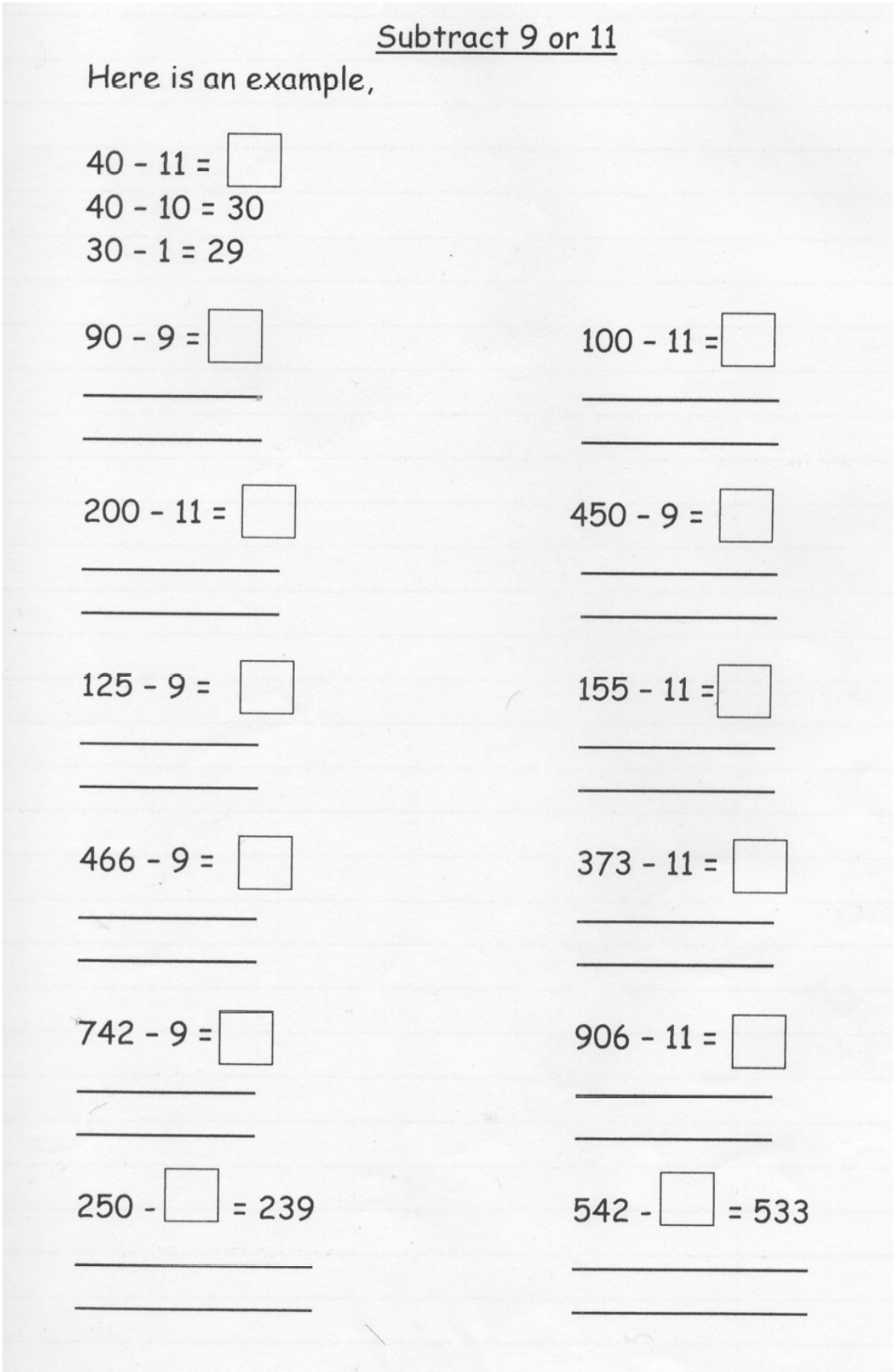
46 - 9 = ☐

37 - 11 = ☐

14 - 9 = ☐

23 - 11 = ☐

10.4 Worksheet example 2



Subtract 9 or 11

Here is an example,

40 - 11 = ☐
40 - 10 = 30
30 - 1 = 29

90 - 9 = ☐

100 - 11 = ☐

200 - 11 = ☐

450 - 9 = ☐

125 - 9 = ☐

155 - 11 = ☐

466 - 9 = ☐

373 - 11 = ☐

742 - 9 = ☐

906 - 11 = ☐

250 - ☐ = 239

542 - ☐ = 533

10.5 Questionnaire 1

Questionnaire for Key Stage One, Year Two class teachers

"I am planning to design an interactive website to help the teaching of addition of nine, by the method of addition of ten and then subtraction of one, to Key Stage 1, Year 2"

I would be very gratefully if you could answer a few questions in order to help with this project. I'm trying to get an accurate picture of what methods and opinions are around. I'm not assessing you or your school in any way so please answer the questions as accurately as possible.

How long does the average child, in your class, use a computer within numeracy lessons?

Every lesson	Once a week	Once a month	Once a term	Once a year	Never

How long does the average child, in your class, use a computer throughout the whole of their lessons?

Every lesson	Once a week	Once a month	Once a term	Once a year	Never

Do you think a program should incorporate animations to explain instructions and provide help with answering them?

Yes No

Do you think music would help keep a child's attention or distract them?

Help keep attention Distract attention

Would an interactive 'help' incorporated within the program be useful?

Yes No

In an average class, how many children will thoroughly understand the concept of addition of nine, as described at the top of the page?

None 10% 20% 30% 40% 50% 60% 70% 80% 90% All

How would you describe the level of ICT resources within your School?

1 Computer/Child 3-4 Computer/Class 1 Computer/Class 1 Computer/School

How many adult supervisors, on average, do you have within your classroom for numeracy?

1 or less 2 3 4 5+

What level of adult supervision do you think would be needed for children taking part in a program similar to the one described at the top of the page?

V. High High Medium Low None

What level of adult supervision do you think would be ideal?

V. High High Medium Low None

Do you think 'Drill and Practise' programs have a place within the classroom (if you do not know what 'drill and practise' is please leave blank)?

Yes No

Does your school have a computer club?

Yes No

How many children in your class have a computer available to them out of school?

None 10% 20% 30% 40% 50% 60% 70% 80% 90% All

Do you think a program, like this, has a place within your classroom?

Yes No

Please list any mathematical software that you use on a regular basis?

……
……
……

If you use any teaching software, e.g. Maths Toolbox 1 or Abacus teacher's software, please could you list them?

……
……
……

Thank you very much for your time.

Please feel free to give any comments and advice below, or if you would prefer please send email to project@tomvian.co.uk.

Thank you again for your time.

Thomas Vian

10.6 Questionnaire 2

Questionnaire of key stage one, year 2

Do you have a computer at home, inc play station etc.?

How do you think could be improved?

Do you like the animations?

Do you listen to the sounds?

Do you think we should use capital letters and full stops when we work on the computer?

If you had a problem on the computer what would you do?

Do you understand everything being asked in the program?

How often do you use the computer in school?

How often do you use a computer at home?

10.7 Questionnaire 3

Oscar's Maths Factory

Have you enjoyed working on Oscar's Maths Factory today?

Do you like Oscar the dog?

Do you think adding on 9 is easy?

Did Oscar help you today?

Was Oscar Maths Factory easy to use?

Thank you