

Programming Project - Unit 3

[NEA Title]

Lewis Miller | Candidate Number: [2086]

Computer Science | H446-03 Programming Project

Bournemouth School | Centre Number: 55119

Monday, 30 April 2018

# Mark scheme

1. Analysis
   1. Problem identification
      1. Description of the problem
      2. Describing and justifying the features that make the problem solvable by computational methods
      3. Explaining why the problem is amenable to a computational approach
   2. Stakeholders
      1. Identifying parties that may have an interest in the solution
      2. Explaining how the solution is suited to the stakeholder’s needs
   3. Researching the problem
      1. Finding instances of the problem
      2. Researching pre-existing solutions
      3. Identifying and explaining essential features of a solution
      4. Identifying and explaining the limitations of the proposed solution
   4. Specification of a solution
      1. Specify and justify the solution requirements including hardware and software configurations where appropriate
      2. Specify and justify measurable success criteria for the proposed solution.
2. Design
   1. Breaking down the problem into its constituent components
   2. Describing a solution
      1. Describing the structure of a solution
      2. Describing and justifying the individual algorithms used in the complete solution
      3. Describing usability features used in the solution
      4. Identifying key variables / data structure / classes and justifying / validating my choices
      5. Identifying and justifying useable test data during development and post development
3. Development
   1. Iterative development
      1. Commented code listings for each stage
      2. Evidence of prototype solutions for each stage
   2. Testing to inform development
      1. Evidence of testing at each stage, justifying the reason for the test
      2. Evidence of remedial actions in later iterations
4. Evaluation
   1. Testing to inform evaluation
      1. Evidence of testing robustness of final solution
      2. Evidence of usability testing (User feedback)
   2. Success of solution
      1. Comparison of process and solution with the original success criteria
   3. Describing the final product
      1. Commenting on the effectiveness of the design and usability
   4. Maintenance and further development
      1. Discussion of maintainability of solution
      2. Discussion of potential further development

# Contents

Mark scheme 1

Contents 3

Analysis 4

Identifying the problem 4

Stakeholders 4

Similar products and existing solutions 4

Python (IDLE) 4

Scratch 5

Small Basic 5

BBC Micro:Bit 5

# Analysis

## Identifying the problem

In a society that is rapidly advancing through the technological era, it is becoming more and more imperative for younger students to gain an understanding of not only how computers work, but also how they can be used and programmed. As such I will be developing a program designed to help younger students not only to become engaged in programming but also to develop thinking skills that can be helpful for completing tasks later in life.

Because the problem is primarily focused on how school students in lower years interact with and understand computers and computational methods, a suitable approach could prioritise a “hands-on” approach to the subject, aiming to encourage pupils to gain an understanding through trial and error. This would allow them to gain a deeper insight into how real computer programs run in real life. In order to do this, students could be provided with their own IDE or similar program, which itself would have to be programmed. These aspects make the program amenable to a computational approach as providing a rich learning environment integrated with the ability to produce a working example of software is often difficult to do with plain pen and paper.

## Stakeholders

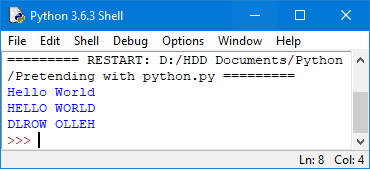
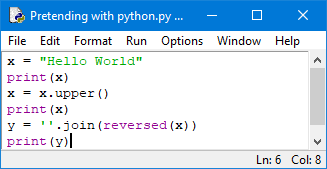
As the program is an educational assistant its primary stakeholders will be teachers and students.

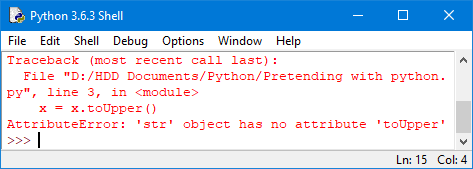
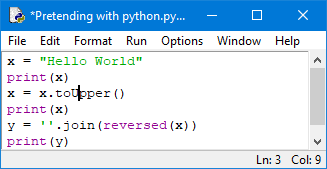
* Teachers
  + Effective teaching aid
  + Easy to show a class to use
* Students
  + Easy to learn, understand and use
  + Allows for developmental experimentation

## Similar products and existing solutions

### Python (IDLE)

Python is a high level programming language with simple whitespace / colon based syntax that uses an English-like command set. Version 0.9.0 was first released in 1991 and since then it has grown into one of the most popular programming languages for people new to programming. The Python IDLE is a very minimal IDE, highlighting keywords and automatically indenting your lines. It has no error checking or advanced features that IDEs like Visual Studio and WebStorm possess. The Shell will show the error when it occurs, but there is very little detail given, making it harder for novice programmers to understand what went wrong.

The Python IDLE (left) and Shell (right)

When an error occurs, the Shell tells you where it happened and what went wrong, but it is not a detailed description

Python is good for new programmers as it provides an easily understood programming language with a lot of extensibility and compatibility options. The primary drawback of Python isn’t the language, but the integrated IDLE. It lacks most forms of assistance provided that many other IDEs provide, requiring novices to refer to documentation repeatedly while starting out so they can understand what went wrong and why.

### Scratch

### Small Basic

### BBC Micro:Bit