**Little Heath Sixth Form**

**Computing** Personal Learning Checklist

**Student Name: ……………………….…………………………………..………**

|  |  |
| --- | --- |
| **Unit Name Problem Solving, Programming, Data Representation and Practical Exercise** | **Unit Code: Comp 1** |
| *Minimum Target Grade:* | *Aspirational Target Grade:* |

*KEY:* ***Red =*** *with difficulty* ***Amber*** *= not sure* ***Green*** *= yes*

|  |  |  |  |
| --- | --- | --- | --- |
| **GCSE Re-Cap (Skills and Knowledge)** | **Red** | **Amber** | **Green** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Skills** | **Red** | **Amber** | **Green** | **To address this before the exam I will:-** |
| * **Devising algorithms** |  |  |  |  |
| * **Programming in a text based language** |  |  |  |  |
| **Knowledge/Specification** |  |  |  |  |
| **Fundamentals of Problem Solving**  **Stages of Problem Solving**   * + Understand the problem.   + Define the problem.   + Define boundaries.   + Plan solution.   + Check solution. |  |  |  |  |
| **Algorithm Design**  Express the solution to a simple problem as an algorithm using flowcharts, pseudo-code or structured English and the standard constructs:  • sequence  • assignment  • selection  • repetition  Hand trace simple algorithms.  Convert a simple algorithm from  • structured English into pseudo-code,  • pseudo-code into high level program code.  Understand the standard algorithms: Bubble Sort, Linear Search. |  |  |  |  |
| **Finite state machines with outputs**  Draw and interpret simple state transition diagrams, transition tables. |  |  |  |  |
| **Fundamentals of Programming**  **Data Types**  Use the following appropriately.  • Built-in Integer, byte, real, Boolean, character, string, date/time.  • User-defined Enumerated, sub range, sets, records, arrays.  **The Role of Variables**  Recognise the different roles a variable can take:  fixed value, stepper, most recent holder, most wanted holder,  gatherer, transformation, follower.  **Programming Statements**  Use these statement types.  • Type Definitions  • Variable Declarations  • Constant Definitions  • Procedure/Function Declarations  • Assignment  • Iteration  • Selection  • Procedure and Function calling.  Explain the advantages of procedure/functions.  **Arithmetic operators** +, –, /, x, DIV, MOD  **Relational operators** =, <, >, <>, <=, >=  **Boolean operators** NOT, AND, OR  **Logical bitwise** **operators** NOT, AND, OR, XOR  **Set operators** Union, difference, intersection, membership.  **Built-in functions** Arithmetic functions: round, truncation. String handling functions: length, position, substring, concatenation. String conversion functions to/from integer, real, date/time.  **Constants and Variables** Explain the advantages of named variables and constants.  **Procedure and Function Parameters.** Describe the use of parameters to pass data within programs. Understand the different mechanisms for parameter passing: by value and by reference.  **Fundamentals of Structured Programming** Understand the structured approach to program design and construction.  Construct and use structure tables, structure charts and hierarchy charts when designing programs.  Use meaningful identifier names.  Use procedures/functions with interfaces.  Use procedures that execute a single task.  Explain the advantages of the structured approach.  **Data Structures**  • One- and Two-Dimensional Arrays Use arrays in the design of solutions to simple problems.  • Fields, Records and Files Read/write records from/to a file: csv file or file of records.  **Validation** Understand the importance of validation of input data. Program simple validation. |  |  |  |  |
| **Fundamentals of Data representation**  **Bit Patterns in a Computer**  Explain the different interpretations that may be associated with a pattern of bits.  **Representation of Denary Integers**.   * Describe the representation of unsigned denary integers in binary. Perform conversion from denary to binary and vice-versa. * Binary Arithmetic Add two binary numbers and multiply two binary numbers. * Representation of signed integers by Two’s Complement Describe the use of Two’s Complement to perform subtraction. Convert a denary integer into Two’s Complement and vice versa. * The Concept of Number Bases: Denary, Binary and Hexadecimal. Describe the conversion of a denary integer to hexadecimal form and vice versa. Describe the use of hexadecimal as shorthand for binary. * Integers and Numbers with a Fractional Part Draw a distinction between integers and numbers with a fractional part in a computer context. * Describe how an unsigned denary number with a fractional part is represented in fixed-point form in binary.   **Information Coding Schemes Describe standard coding systems for coding character data.**  • ASCII Differentiate between the character code representation of a denary  • Unicode digit and its pure binary representation.  **Error checking and correction**  Parity bits, Hamming code.  Gray coding Describe Gray coding. Explain why and where it is used.  **Representing Images**  Describe how bit patterns may represent graphics  Bitmapped Graphics Bitmaps: resolution, colour depth and simple bitmap file calculations.  Vector Graphics Vector graphics: drawing list – objects and their properties.  Compare bitmaps to vector graphics; advantages, disadvantages.  **Representing Sound**  Describe how bit patterns may represent sound  The need for compression of sound files and basic techniques for compression.  Sampled Sound and Nyquist theorem  Sampling resolution, sampling rate.  Sound Synthesis Streaming audio.  Differentiate between analogue and digital data and analogue and Analogue and Digital Signals digital signals.  Describe the principles of operation of an analogue to digital  converter. |  |  |  |  |
| **Systems Development Life Cycle**  Describe the stages of development of a hardware/software system:-  Analysis  Design  Implementation - Program the solution as per design.  Testing Specify the method of testing the programmed solution (dry run testing, black box testing, white box testing).  Specify the selection of test data including normal (typical), boundary and erroneous data.  Test the solution using selected test data.  Evaluation |  |  |  |  |

|  |
| --- |
| **REVISION**  **Use the information on this checklist to make revision cards and notes** |

**Grade tracking:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| *Grade* | *Date* | *Grade* | *Date* | *Grade* | *Date* |
|  |  |  |  |  |  |
| *Grade* | *Date* | *Grade* | *Date* | *Grade* | *Date* |
|  |  |  |  |  |  |

*Note: You should discuss this checklist regularly with your subject teacher/mentor*