Computer Science Marking Scheme

Answe1: Any two from:
- Kilobyte is a SI (physics/science) / International System of Units measurement, whereas Kibibyte is an IEC (computing) / International Electrotechnical Commission / International System of Quantities / ISO measurement (1)
- Kilobyte is equivalent to 1000 bytes, whereas a kibibyte is equivalent to 1024 bytes (1)
- Kilobyte is equivalent to 103 bytes, whereas a kibibyte is equivalent to 210 bytes (1)
- Kilobyte is a base 10 measurement, whereas a kibibyte is a base 2 measurement (1)

Answer2: Any two from:
- To save disc/storage space (1)
- Reduce transmission time (1)
- Meet email attachment size restrictions (1)

Answer3: Any two from:
- Compressing / decompressing the file takes execution time (1)
- Both compressing and decompressing have to be done with compatible software / both ends of transmission have to use compatible software (1)
- (If the compression utility uses a lossy algorithm then) some of the data will be permanently removed / it won't be possible to reconstruct the original file. (1)
- Can affect the quality of the

Answer4:
- Features such as power consumption / processor / memory / number and type of components / physical size (1)
- can be optimized / chosen to fit just one specific task. (1)

Answer5: Any two from:
- Printer (1)
- Bank card (magnetic strip) reader/scanner (1)
- Bank card contactless/NFC reader/scanner (1)
- Cash input and giving change device (1)

Answer6: To prevent unauthorized people from reading / understanding the bank card information (1)

Answer7: Award one mark for each of (maximum of 4):
- Readability: high-level language is easily readable by a human because it is designed to look like English whereas a low-level language is binary patterns / mnemonics / abbreviations. (1)
- Portability: code written in a high-level language is easily transportable across machines whereas a low-level language is designed for a specific microprocessor / piece of hardware. (1)
- Uses: low-level languages are commonly used to write programs for hardware or devices, like printers, whereas high-level languages are commonly used for to write higher-level applications, such as word processors. (1)
- Optimisation: low-level languages are used to improve performance, especially of hardware, whereas high-level languages have to go through multiple layers of software to get to the hardware, e.g. the need for HLL to be compiled/interpreted makes programmes run slower than LLL, HLL has a higher level of abstraction removing unnecessary detail (one to many), LLL keeps more of a detail (one to one) (1)
- Ease of use: high-level languages usually have tools, functions, libraries and development environments which are accessible, even by those with little knowledge, whereas, low-level languages are often challenging to work with and require expertise and depth of knowledge. (1)