**Isaac A-level Booster – Assembly Language**

**Handout 2 – Assembling and Disassembling**

There is a one-to-one relationship between assembly code mnemonics and machine code instructions. Therefore, compared with compiling high-level source code, assembling is a relatively simple translation process.

Let’s do the job of an assembler! Translate this set of LMC mnemonics into their machine language instruction counterparts. Use the LMC Instruction Set here to help you.

|  |  |  |  |
| --- | --- | --- | --- |
| **Instruction** | **Description** | **Mnemonic** | **Numeric Code** |
| Load | Loads value into the ACC | LDA | 5xx |
| Store | Stores a value from ACC to RAM | STA | 3xx |
| Add | Adds a value to the ACC | ADD | 1xx |
| Subtract | Subtracts a value from the ACC | SUB | 2xx |
| Input | Takes input from user and stores in ACC | INP | 901 |
| Output | Outputs value from ACC | OUT | 902 |
| End | Halts the program | HLT | 000 |
| Branch if zero | Branches if the ACC is zero | BRZ | 7xx |
| Branch if zero or positive | Branches if the ACC is zero or positive | BRP | 8xx |
| Branch always | Branches always | BRA | 6xx |
| Data storage | Used to store one piece of data | DAT |  |

**a) Assemble this program into denary codes**

|  |  |
| --- | --- |
| INP  STA 99  OUT  HLT |  |

We can also work backwards from machine code instructions to assembly language mnemonics, this is called disassembling.

**b) Disassemble this program, converting machine code instructions back to mnemonics**

|  |  |
| --- | --- |
| 901  388  284  587  902  000 |  |