Task 1: Comparing RISC with CISC

```
x=3
y=5

First program

LOAD y (from memory into register1)
LOAD x (from memory into register2)
PROD register1, register 2
STORE (answer back into memory)

Second program

MULT x,y
```

Program 1

The program stores the value of x (3) into memory address 1 and the value of y (5) into memory address 2. PROD then multiplies the value of addresses 1 and (3 * 5 = 8).
This is then stored in the memory, a given address.

Program 2

This multiplies the values x (3) and y (5) to give 15.

Both are in assembly code but appear different. Program 1 is using RISC but program 2 is using CISC

What is RISC?

RISC = A reduced instruction set computer, is a type of computer architecture that uses a small but highly-optimised set of instructions. The instructions are less versatile and are very simple but each instruction (NOT TASK) only requires one clock cycle.
The same multiplying task would require the following assembly code:

```
LDA R1 ,      x
LDA R2,       y
   MULT x,y
STO R1,       x
```

For example, only the load and store instructions access memory. All other instructions operate on the registers. Each instruction is so simple that it can be executed very quickly in one cycle.

All instructions in a RISC system should execute in roughly the same, small, number of clock cycles (ideally 1) Therefore,
RISC supports pipelining = allowing different stages of instructions to be executed simultaneously.
RISC processors tend to have fewer transistors, produce less heat and are more energy efficient and cost less than non-RISC counterparts.
A greater number of registers are needed in order to provide faster access to data when programs are running.
More RAM is required to store the machine code instructions.
RISC has largely replaced the older CISC design. However, CISC is still used in microcontrollers and for embedded systems.

On the other hand, a RISC compiler has a harder job (is more complex) as it has to determine how the functionality specified by the higher level code can be built from a more limited set of instructions when converting to machine code.

What is CISC?

CISC = A complex instruction set computer, is a type of computer architecture that can operate with single instructions which execute several low-level operations (e.g. load, arithmetic, store).

A single instruction can carry out multiple operations. There is a diverse range of instructions (versatile instruction set) but tasks can be completed with fewer lines of more complicated assembly. The main idea is

to accomplish a task in a few lines of assembly as possible by using a large instruction set. The processor and hardware is capable of understanding a series of sub-tasks that make up a single instruction.

The distinguishing feature is that is carried out the load/store function with the same instruction that carries out the actual calculation.

To multiply 2 values stored in separate memory locations x and y and store the result of the calculation in location x. The assembly language CISC instruction would be MULT x, y . The instruction is equivalent to a high-level instruction such as a = a * b .

The hardware is more complicated and requires more silicon transistors.
However, the compiler has less work to do to translate the high-level language statement into machine code.
Since the code is very short, the registers and RAM is smaller as less is required to store instructions.

A CISC processor comes with a specific instruction (MULT). This specific instruction is capable of loading two values into separate registers, multiplying them and storing the product in the appropriate register. The entire task is done in one instruction.

A disadvantage of a CISC processor is that many specialised instructions had to be built into the hardware even though only ~ 20% is actually used in the average program.

Speed comparison
To handle complex tasks, simple instructions are combined from the reduced instruction set, CISC could carry out the same task with fewer instructions.
CISC = faster for more intensive tasks (can use complex instruction set).
RISC = faster for simpler operations as CISC uses a complex instruction set (so it can adopt pipelining).

RISC	CISC
Simpler hardware (burden on software/compiler)	Has more complex hardware (simpler compilers)
Used in laptop and desktop computers, made by Intel or AMD.	Used in smartphones and tablets, based around ARM processors.
Lower energy requirements, and can go into “sleep mode” when not actively processing.	Higher energy consumption
Physically smaller in size as less complex circuitry needed, less silicon needed thus making it cheaper.	Physically larger in size and require more silicon to make thus more expensive.
Can support pipelining	Can’t support pipelining
Instructions are simple so more needed (but instruction set is streamlined not very diverse)	Instructions are more complex so less needed (but instruction set is very diverse)
Run at lower clock speed (single cycles a second)	Runs at higher clock speed (more cycles a second)
Single machine cycle per instruction	Multiple machine cycles per instruction
Uses more registers, uses more of main memory	Uses more registers, uses less of main memory
Load and store register-register are independent instructions	Load and store memory-to-memory is induced in instructions
Higher performance with simpler, less intensive tasks.	Higher performance when handling more complex, intensive tasks.