9608/42/PRE/O/N/20

Last update: Anuj Verma, 03:16 PM 06/10/2020

These are the files that constute the solution to the pre-release material for Computer Science component 9608/42 of the October/November 2020 examination series.

Filename	**Type**	**Purpose**
9608_w20_PM_42	`.pdf`	The pre-release material file released by CAIE.
Planning	`.md`	This is the markdown text file that this PDF was created from.
Planning	`.pdf`	You are currently reading this file. It describes the solution used in answering the pre-release material and houses all material apart from code (such as identifier tables and structured English).
Main Python notebook	`.ipynb`	The Jupyter Notebook in which the Python code was originally written.
Main Python notebook	`.pdf`	The PDF version of the Jupyter Notebook (for the viewer whose system doesn't have Jupyter).
Component Programs	`.py`	The Python 3.8 file that contains all executable code (for the viewer whose system doesn't have Jupyter).
Conponent Pseudocode	`.psu`	All pseudocode, grouped by the task numbers, written using an open-source custom-built extension.
Standalone Compiled Program	`.py`	The final Python program.
Standalone Compiled Pseudocode	`.psu`	The final pseudocode.

TASK 1 – Low-level programming

> Low-level programming is a type of programming language that uses op codes and operands to create instructions.
>
> The table (in the question paper) shows part of the instruction set for a processor that has one general purpose register, the Accumulator (ACC), and an Index Register (IX).

TASK 1.1

> Write assembly language program code that allows a user to input 5 characters. The characters are not stored.

The program is given in the table below, and is in the attached Word document.

TASK 1.3

> Write assembly language program code that adds the values stored in four consecutive memory locations starting at `NUMBER` using the Index Register.
>
> Store the final total value in memory location `TOTAL`.

Label	Op code	Operand	Comment
	LDR	#0	// initialise Index Register to 0
LOOP:	LDX	NUMBER	// load value from NUMBER + contents of Index Register
	ADD	TOTAL	// add value to TOTAL
	STO	TOTAL	
	INC	IX	// increment Index Register
	LDD	COUNT	// increment COUNT
	INC	ACC	
	STO	COUNT	
	CMP	MAX	// is COUNT = MAX ?
	JPN	LOOP	// jump to LOOP if FALSE
	END		// end program
MAX:	4		
COUNT:	0		
NUMBER:	23		
	17		
	38		
	13		
TOTAL:	0		