1) Calculate “speed up” in a 5 stage pipeline CPU which every stage has different time. Stage durations: 20, 15, 25, 20, 20.

2) Consider two different machines, with two different instruction sets, both of which have a clock rate of 200 MHz. The following measurements are recorded on the two machines running a given set of benchmark programs:

Instruction Type	Inst Count (millions)	CPI
Machine A		
Arithmetic and Logic	8	1
Load and Store	4	3
Branch	2	4
Others	4	3
Machine B		
Arithmetic and Logic	10	1
Load and Store	8	2
Branch	2	4
Others	4	3

Determine the effective CPI, MIPS rate, and execution time for each machine. (Million instructions per second = MIPS: an approximate measure of a computer's raw processing power)

3) Assume a program with 820,000,000 instructions is needed for spell checking of a very large file. There are 4 types of instructions in this program and each type needs N clock cycle for execution.

Instruction Class	Clock cycles per Instructions	Number of Instructions
Branch	3	150,000,000
Store	4	185,000,000
Load	5	260,000,000
ALU/R-type	4	225,000,000

Duration of complete run of the program is 1.57 seconds. Find out clock cycles time of execution in this computer.

4) Compute the Clocks Per Instruction (CPI) of a machine which has an average CPI for ALU operations of 1.1, a CPI for branches/jumps of 3.0, and a hit rate of 60% in the cache. A hit in the cache takes 1 cycle pipelined and a cache miss takes 120 cycles. Assume 22% of instructions are loads, 12% are stores, 20% are branches/jumps and the balance are ALU operations.

5) For the following code snippet, identify all of the RAW, WAW, and WAR hazards. Provide a list for each hazard. (Hint, remember that you have to check more than neighbor instructions.)

```
ADD R1, R2, R3
SUB R3, R4, R6
MUL R5, R4, R7
ADDIU R5, R5, 1
SUB R6, R3, R9
ANDI R2, R1, R9
```

Bonus point) Most computers pass parameters using a run-time stack stored in memory.

This means any function call and return requires potentially several memory accesses. An alternate architecture, Berkeley RISC, uses register windows. With register windows, local variables that are passed as parameters are placed in a set of registers that overlap registers between the registers of the calling function and the called function.

This “overlap” is a window. It allows parameter passing without the use of memory. See the figure below.

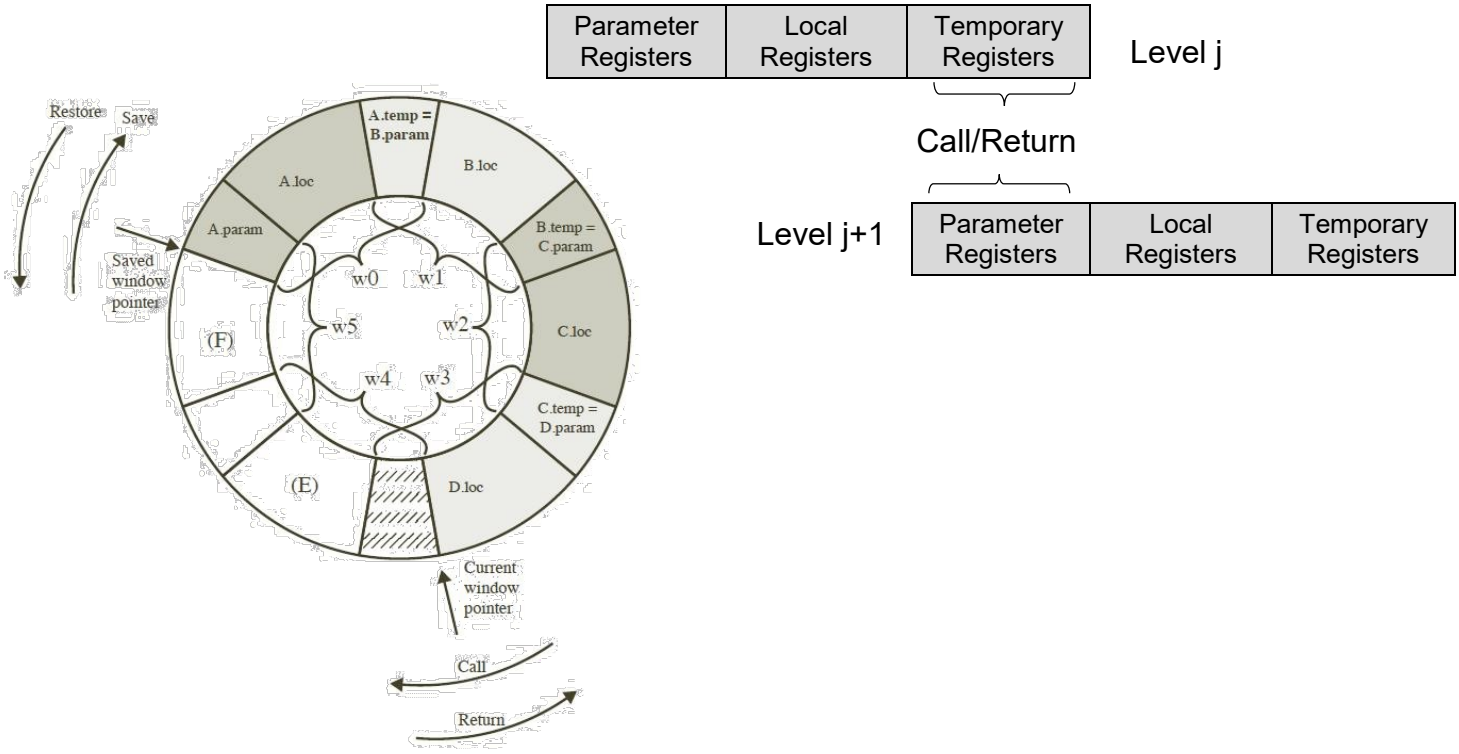
Register windows replace memory access operations so IC remains the same but CPI changes as register operations require fewer clock cycles than memory operations.

Use for CPI: Loads/stores: 4, ALU and unconditional branches: 2, conditional branches: 3, procedure calls and returns: 15.

Architects are trying to decide whether to use additional registers in the CPU for register windows or a larger register file.

Enlarging the register file reduces the number of loads and stores by 40% and 30% respectively. Register windows reduces procedure call CPI to 4.5 and returns to 3.

Given a benchmark of 40% load, 13% store, 31% ALU, 8% conditional branches, 2% unconditional branches, 3% procedure call and 3% return, should we use the added registers for a register window or a larger register file?



If you have any questions regarding this assignment, feel free to contact us.

Please submit your homework, simulations and projects in the following format:

Name_StudentNumber_HW3 (BillGates_12345678_HW3)

Good Luck!