**COMPUTER ORGANISATION**

**Assignment 1**

**Name**: Aman Aslam Sayyad

**MIS:** 112103127

**Div**: 2

**Batch**: T2

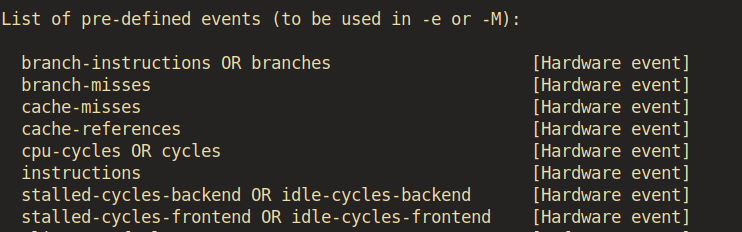
**1. Title: Study and experimentation using perf tool to observe different statistics of a program.**

**1. Installing perf**



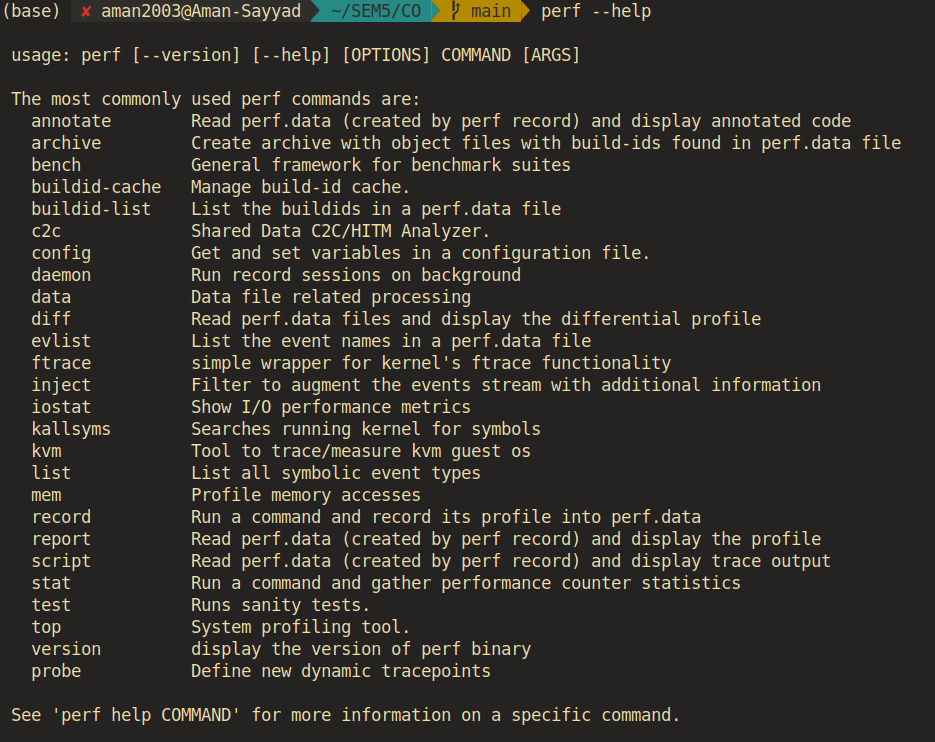
**2. Perf commands**:

1. perf list: Gives a list of supported events



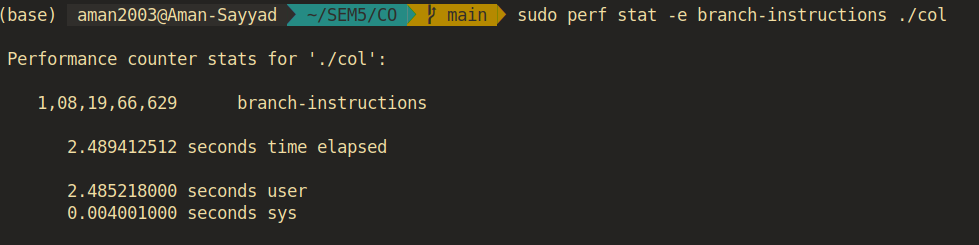


1. Perf --help: It provides a list of options and subcommands for the perf tool, allowing you to access help documentation for various perf functionalities.



**3. Perf Events:**

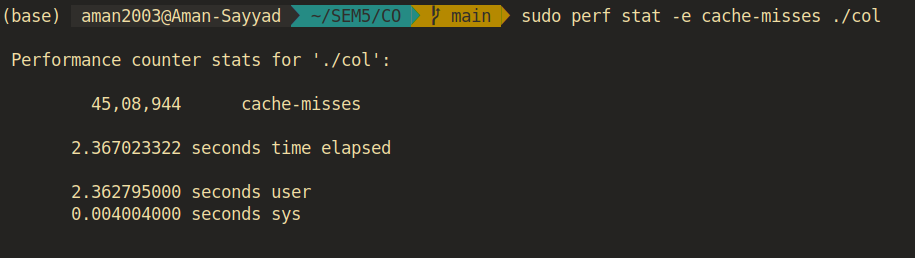
1. Branch instructions: This event counts the total number of branch instructions executed by the CPU. Branch instructions are instructions that can change the order of program execution, such as conditional branches and loops. Counting branch instructions can provide insights into the program's control flow and how often it makes decisions or jumps in its execution path.



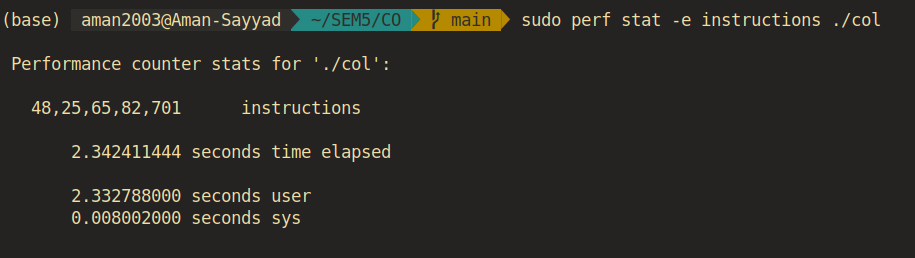
1. Branch misses:This event counts the number of times the CPU's branch predictor incorrectly predicts the outcome of branch instructions. Modern CPUs use branch prediction to speculate about which branch of a conditional statement will be taken. When the prediction is incorrect, it results in a "branch miss." High branch miss rates can indicate inefficiencies in the branch prediction mechanism and may lead to performance degradation.



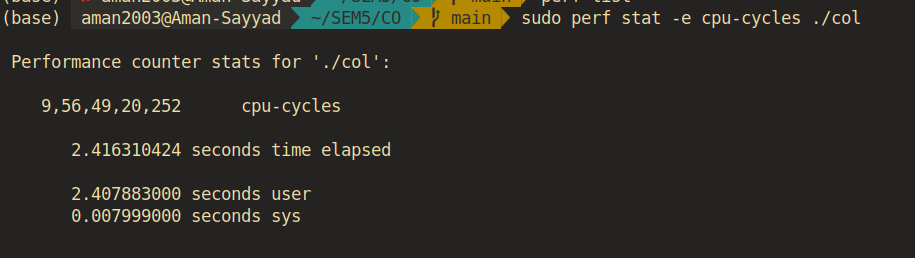
1. Cache misses:This event counts the number of times the CPU attempts to access data from its cache memory but fails to find the data there, requiring it to fetch the data from a slower memory location, such as RAM. Cache misses are significant because accessing data from cache is much faster than accessing it from main memory. A high cache miss rate suggests that the program's memory access patterns may not be cache-friendly, leading to slower execution.



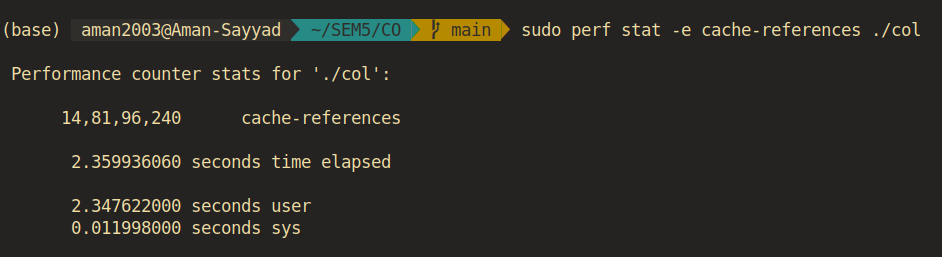
1. Cache references: This event counts the total number of memory access operations made by the CPU to its cache memory. It includes both successful accesses (cache hits) and unsuccessful accesses (cache misses). Cache references provide an overall measure of how frequently the CPU interacts with its cache. A high cache-reference count may indicate a high degree of memory access in the program.



1. CPU cycle:This event measures the total number of clock cycles the CPU takes to execute a program or a specific portion of code. CPU cycles represent the basic unit of time in CPU operations. This metric is useful for understanding the raw computational effort required for a task, as the number of cycles is proportional to the execution time on a CPU.



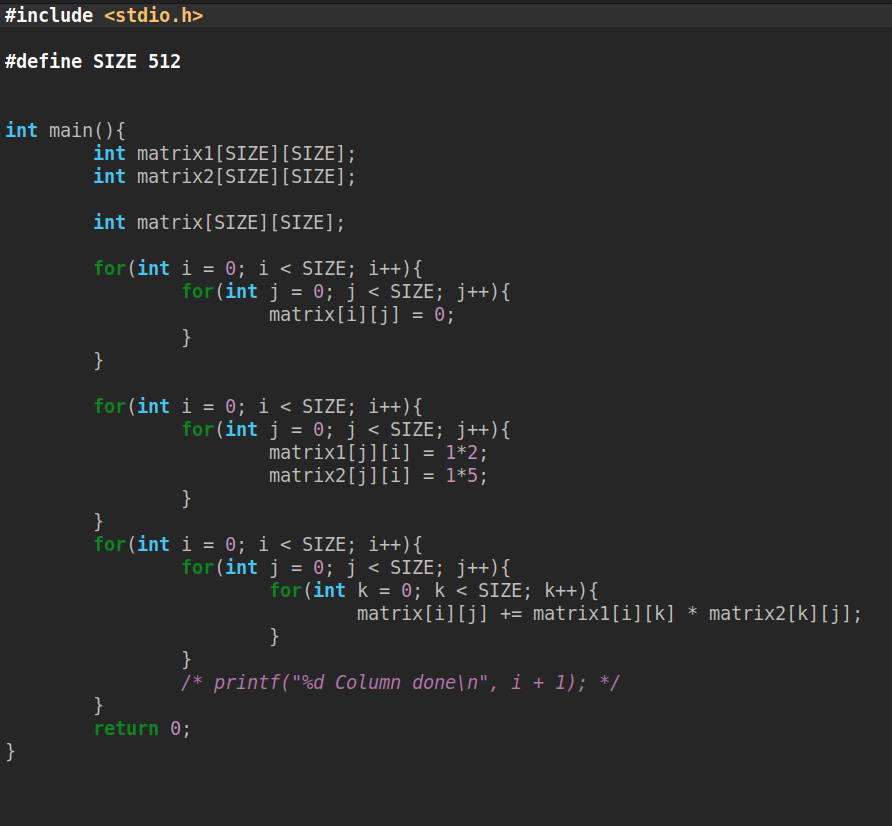
1. Instructions:This event counts the total number of machine-level instructions executed by the CPU. Machine instructions are the fundamental operations that the CPU performs to execute a program. Counting instructions helps assess the complexity and computational load of a program. It is often used as a baseline metric for program performance analysis.



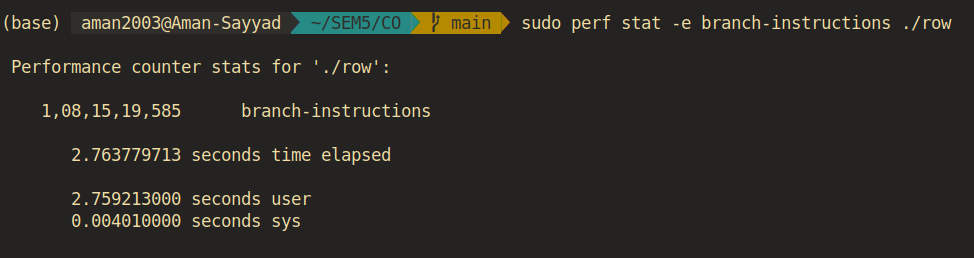
**2. Title: Write a program to multiply two different matrices of size 1024 x 1024.**

1. **Column-wise Access:**

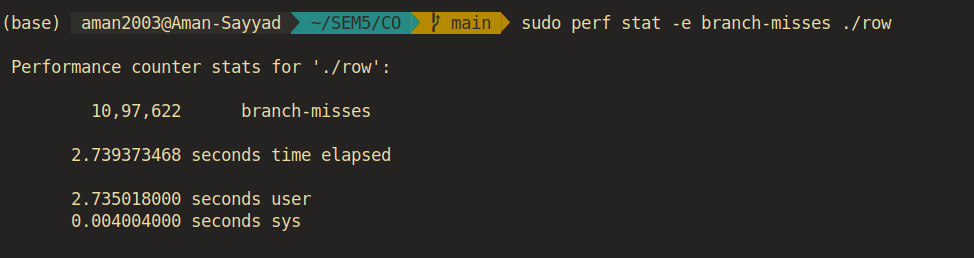
**code of column-wise access:**



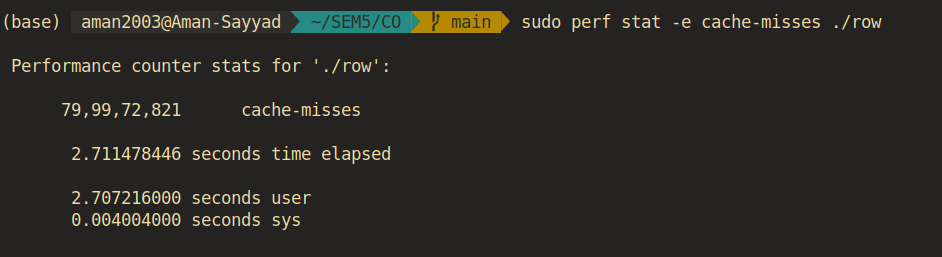
a. Branch-instructions:



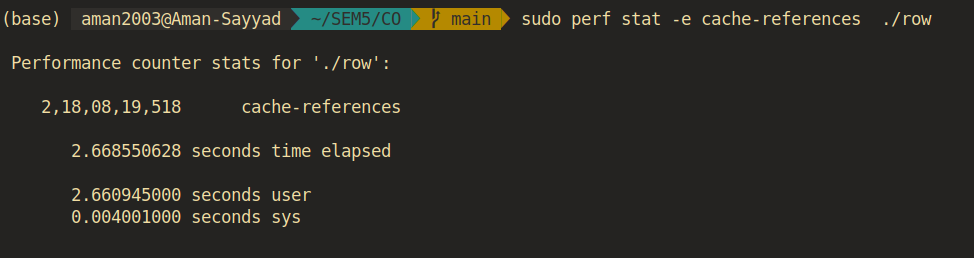
b. Branch-misses:



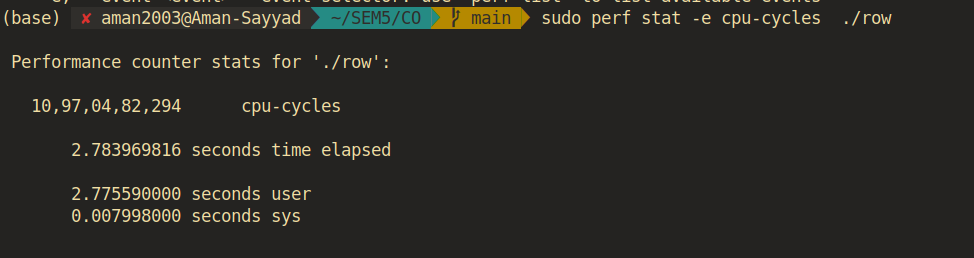
c. cache-misses:



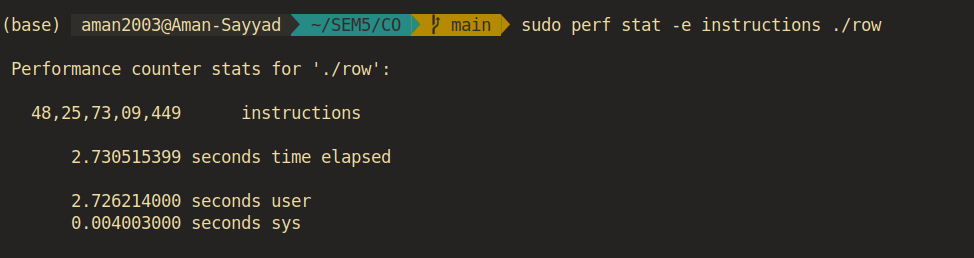
d. Cache-references:



e. Cpu-cycle:

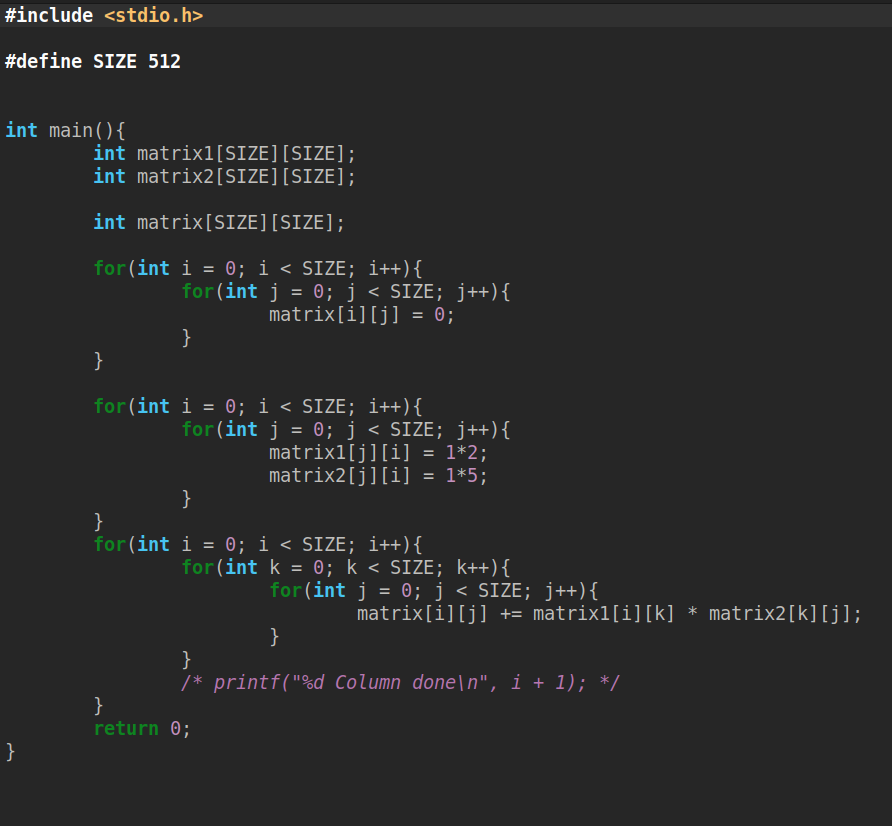


f. Instructions:

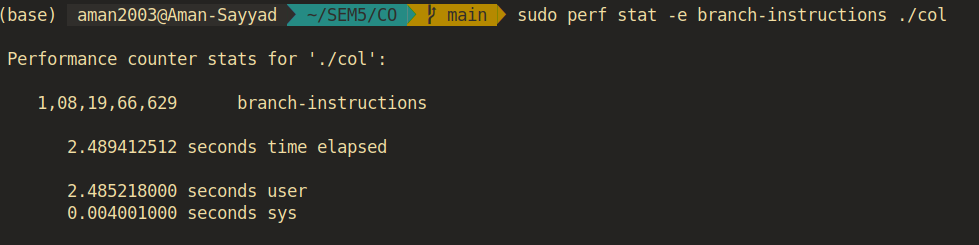


1. **Row-wise access:**

**code of row-wise access:**



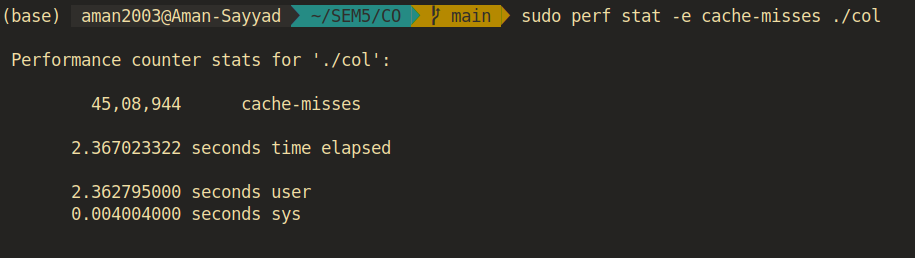
a. Branch-instructions:



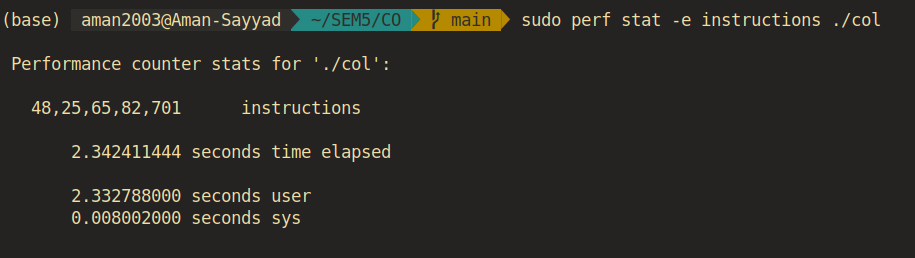
b. Branch-misses:



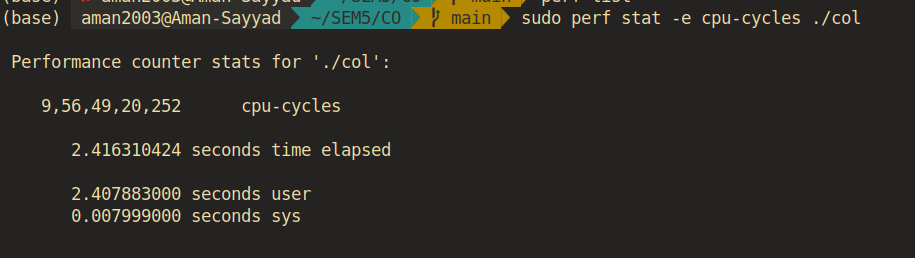
c. cache-misses:



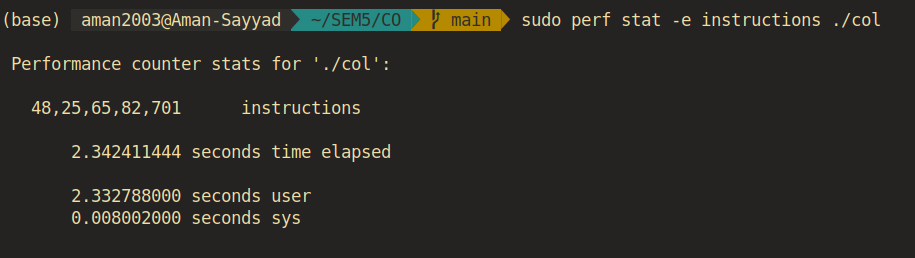
d. Cache-references:



e. Cpu-cycles:



f. Instructions:



**Differences:**

* While comparing I came to know the output of cache misses of column-wise is more than row-wise access in matrix multiplication
* cpu cycles of column-wise were more than row-wise because cache misses of column-wise access were more than row-wise access.
* Other events will not have any significant differences because the number of instructions will be the same for both programs, so branch instructions of both programs are the same.

**Conclusion:**

From above experiments with perf I came to know what software and hardware events are. More specifically I came to know different types of hardware events like branch instructions, branch misses, cache misses, cache references, cpu cycles etc.