Exercise 2:

Write a Pintool (in JIT mode) that prints into a file called "**edge-profile.csv**", the profiling information about each executed basic block and each executed jump instruction (both: direct and indirect).

The pintool should be named "**ex2.so"**.

For each basic block (bbl) with a non-zero execution count, the tool should emit the following information, in this exact format:

<bbl_1 address>, <bbl_1 exec count>, <taken count, if bbl_1 terminates with a condition jump>, <fallthru count if bbl_1 terminates with a condition jump>, <exec count of each target jump up to 10 target addresses, sorted from hottest to coldest, if bbl_1 terminates with an indirect jump: <target $addr_1$, exec count>, <target $addr_2$, exec count>,…< target $addr_{10}$, exec count>>

<bbl_2 address>, <bbl_2 exec count>, <taken count, if bbl_2 terminates with a condition jump>, <fallthru count, if bbl_2 terminates with a condition jump>, <exec count of each target jump up to 10 target addresses, sorted from hottest to coldest, if bbl_2 terminates with an indirect jump: : <target $addr_1$, exec count>, <target $addr_2$, exec count>,…< target $addr_{10}$, exec count>>

…

<bbl_n address>, <bbl_n exec count>, <taken count, if bbl_n terminates with a condition jump>, <fallthru count, if bbl_n terminates with a condition jump>, <exec count of each target jump up to 10 target addresses, sorted from hottest to coldest, if bbl_n terminates with an indirect jump: : <target $addr_1$, exec count>, <target $addr_2$, exec count>,…< target $addr_{10}$, exec count>>

The basic blocks should be **sorted** from most frequently executed ("hottest") to the least frequently executed ("coldest") ones.

You can assume that total number of basic blocks is less than 10,000.

The pintool should not run longer than 5 seconds (elapsed time) on the bzip2 input.

Tips:

1. *See **jumpmix.cpp** pintool* on how to collect statistics on taken vs. non-taken conditional jumps and on indirect jumps.
2. Consider using the following instrumentation API obtain a target addresses of an indirect branch/call:
   INS_InsertCall(tail, IPOINT_BEFORE, AFUNPTR(do_branch_indirect), IARG_BRANCH_TARGET_ADDR, IARG_BRANCH_TAKEN, IARG_END);
   where **do_ranch_indirect()** has the following prototype:
   VOID do_ranch_indirect(ADDRINT target, BOOL taken) { .. }

Test your pintool:

In the moodle you'll find the input binary file called "**bzip2.gz**" along with an input file to give it called "**input.txt.gz.**
Ftp the files to your T2 Linux account and open them using the **gunzip** command.
To run it simply type: $ **./bzip2 –k –f input.txt**
This will compress the file **input.txt** and generate a new file **input.txt.bz2**

To test your pintool on the above **bzip2** binary file, simply type:
<pindir>/pin –t ex2.so -- ./bzip2 –k –f input.txt

Submission requirements:
The submission of this exercise is **in pairs only**.
Submit 1 compressed file called **"ex2.zip"** into the moodle exercise2 link containing the following files:

1. The binary of your pintool **ex2.so** (compiled, and tested by you that it runs and gives the result).
2. A directory called: '**src**' containing all the **source files** (.cpp and .h files) of your pintool along with the "**makefile**", "**makefile.rules",** and a **REDAME.txt** file that includes your **full name**, your **ID** and a description of the compilation command and how to run the tool.

Submission deadline: extended to midnight Thursday, May 15th, 2025.