[CPEN-411 2023] Assignment 1:
Instrumentation, Program Analysis, and Modeling

This assignment discusses program analysis using Pintool[1] and writing efficient searching algorithms. The assignment is divided into three parts. The first part looks to prepare you to start with this assignment. The second part looks at the assignment itself. The third part deals with the submission process. **Please read this document very carefully and in detail!**

Preparatory Steps: Setup

1. **[0 Marks]** Copy the **assignment1.tar.gz** to <username>**@ssh.ece.ubc.ca** server.

```
Assignment-1 — -bash — 130×36
dhcp-128-189-229-100:Assignment-1 prashantnair$ rsync -rtv assignment1.tar.gz prashantnair@ssh.ece.ubc.ca:~/.
```

   If you do not have **rsync** (command) installed. You could try installing it.
   Alternatively, you can try using the **scp command.**
   For instance, Mac users can use **Homebrew**[2] to install these tools. For those who use Windows and Mac can use GUI tools like **CyberDuck**[3] to transfer files into remote servers. Linux users can use their **package manager** and install *rsync* or *scp*.

2. **[0 Marks]** untar **assignment1.tar.gz** in the ece server.

```
Assignment-1 — ssh prashantnair@ssh.ece.ubc.ca — 130×36
dhcp-128-189-229-100:Assignment-1 prashantnair$ ssh prashantnair@ssh.ece.ubc.ca
Welcome to Ubuntu 20.04.3 LTS (GNU/Linux 5.4.0-100-generic x86_64)

139 updates can be applied immediately.
4 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

New release '22.04.1 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

*** System restart required ***

UBC Electrical and Computer Engineering <it@ece.ubc.ca>

This system is for use by authorized users only, and subject to UBC Policy SC14.

Individuals using this computer system without authority, or in excess of their
authority, are subject to having all their activities on this system monitored
and recorded by system personnel.

In the course of monitoring individuals improperly using this system, or in the
course of system maintenance, the activities of authorized users may also be
monitored.

Anyone using this system expressly consents to such monitoring and is advised
that, if such monitoring reveals possible evidence of criminal activity, system
administrators may provide the evidence of such monitoring to law enforcement
officials.
Last login: Tue Sep 13 16:22:52 2022 from 128.189.229.100
ssh-linux4:~> tar - xvzf1 as
assignment1/        assignment1.tar.gz
ssh-linux4:~> tar - xvzf assignment1.tar.gz
```

3. **[0 Marks]** Change the directory into the **assignment1** folder.

```
Assignment-1 — ssh prashantnair@ssh.ece.ubc.ca — 130×34
/Users/prashantnair/Work/CPEN411/2022/Assignment-1 — ssh prashantnair@ssh.ece.ubc.ca
ssh-linux4:~> cd assignment1
ssh-linux4:~/assignment1>
```

Preparatory Steps: Algorithm

1. **[0 Marks]** Change directory into **assignment1/algorithms** folder.

```
Assignment-1 — ssh prashantnair@ssh.ece.ubc.ca — 130×34
/Users/prashantnair/Work/CPEN411/2022/Assignment-1 — ssh prashantnair@ssh.ece.ubc.ca
ssh-linux4:~> cd assignment1
ssh-linux4:~/assignment1> cd algorithms
ssh-linux4:~/assignment1/algorithms>
```

2. **[0 Marks]** Understand the searching algorithm in **searchoriginal.c** and **searchoriginal.h**

```
jhwoo36@ssh-linux3: ~/assignment1/algorithms
jhwoo36@ssh-linux3:~/assignment1/algorithms$ ls
assignment1.o  Makefile  README  runall.sh  searchnew.c  searchnew.h  searchoriginal.c  searchoriginal.h
jhwoo36@ssh-linux3:~/assignment1/algorithms$ vim searchoriginal.c
```

```
jhwoo36@ssh-linux3: ~/assignment1/algorithms
#include "searchoriginal.h"

long long int searchorig(unsigned long long int *arr, long long int size, long long int value){
        return linearSearch(arr,size,value);//Return the index of the element in arr[] which has data that is equal to value
}

long long int linearSearch(unsigned long long int *arr, long long int size, long long int value){
        unsigned long long int counter1, counter2;
        for (counter1 = 0; counter1 < size-1; counter1++){
                if (arr[counter1] == value){
                        return counter1;
                }
        }
        return -1;
}
```

```
jhwoo36@ssh-linux3: ~/assignment1/algorithms
jhwoo36@ssh-linux3:~/assignment1/algorithms$ ls
assignment1.o  Makefile  README  runall.sh  searchnew.c  searchnew.h  searchoriginal.c  searchoriginal.h
jhwoo36@ssh-linux3:~/assignment1/algorithms$ vim searchoriginal.c
jhwoo36@ssh-linux3:~/assignment1/algorithms$ vim searchoriginal.h
```

```
jhwoo36@ssh-linux3: ~/assignment1/algorithms
long long int searchorig(unsigned long long int *arr, long long int size, long long int value);
long long int linearSearch(unsigned long long int *arr, long long int size, long long int value);
```

3. **[0 Marks]** Compile the source files in the **assignment1/algorithms** folder.

```
jhwoo36@ssh-linux3: ~/assignment1/algorithms
jhwoo36@ssh-linux3:~/assignment1/algorithms$ make
gcc -c searchnew.c
gcc -c searchoriginal.c
gcc -O3 assignment1.o searchnew.o searchoriginal.o -o assignment1.bin
jhwoo36@ssh-linux3:~/assignment1/algorithms$ ls
assignment1.bin  Makefile  runall.sh  searchnew.h  searchoriginal.c  searchoriginal.o
assignment1.o    README    searchnew.c  searchnew.o  searchoriginal.h
jhwoo36@ssh-linux3:~/assignment1/algorithms$
```

4. **[0 Marks]** Execute your compiled code.

```
prashantnair — ssh prashantnair@ssh.ece.ubc.ca — 130×34
~ — ssh prashantnair@ssh.ece.ubc.ca
ssh-linux4:~/assignment1/algorithms> bash runall.sh Your Student ID Number
```

5. **[0 Marks]** Change to the **assignment1/output**.

```
jhwoo36@ssh-linux3: ~/assignment1/output
jhwoo36@ssh-linux3:~/assignment1/algorithms$ make
gcc -c searchnew.c
gcc -c searchoriginal.c
gcc -O3 assignment1.o searchnew.o searchoriginal.o -o assignment1.bin
jhwoo36@ssh-linux3:~/assignment1/algorithms$ ls
assignment1.bin  Makefile  runall.sh    searchnew.h  searchoriginal.c  searchoriginal.o
assignment1.o    README    searchnew.c  searchnew.o  searchoriginal.h
jhwoo36@ssh-linux3:~/assignment1/algorithms$ bash runall.sh
assignment1.bin   Makefile          runall.sh         searchnew.h       searchoriginal.c  searchoriginal.o
assignment1.o     README            searchnew.c       searchnew.o       searchoriginal.h
jhwoo36@ssh-linux3:~/assignment1/algorithms$ bash runall.sh 100
jhwoo36@ssh-linux3:~/assignment1/algorithms$ ls
assignment1.bin  Makefile  runall.sh    searchnew.h     searchnew.o       searchoriginal.h     searchoriginal.o
assignment1.o    README    searchnew.c  searchnew.inst  searchoriginal.c  searchoriginal.inst
jhwoo36@ssh-linux3:~/assignment1/algorithms$ cd ../output/
jhwoo36@ssh-linux3:~/assignment1/output$ ls
searchnew.txt  searchorig.txt
```

6. **[0 Marks]** Make sure that **sortorig.txt** has PASSED.

```
jhwoo36@ssh-linux3: ~/assignment1/output
jhwoo36@ssh-linux3:~/assignment1/output$ vim searchorig.txt
jhwoo36@ssh-linux3: ~/assignment1/output
PASSED
```

7. **[0 Marks]** Change to the **assignment1/algorithms**.

```
prashantnair — ssh prashantnair@ssh.ece.ubc.ca — 130×34
~ — ssh prashantnair@ssh.ece.ubc.ca
ssh-linux4:~/assignment1/output> cd ../algorithms/
ssh-linux4:~/assignment1/algorithms>
```

8. **[0 Marks]** Understand the x86-ISA instructions used in the **Linear-Search** algorithm.

```
jhwoo36@ssh-linux3: ~/assignment1/algorithms
jhwoo36@ssh-linux3:~/assignment1/algorithms$ vim searchoriginal.inst
```

```
jhwoo36

Disassembly of section .text:

0000000000000000 <searchorig>:
0:      f3 0f 1e fa             endbr64
4:      55                      push   rbp
5:      48 89 e5                mov    rbp,rsp
8:      48 83 ec 20             sub    rsp,0x20
c:      48 89 7d f8             mov    QWORD PTR [rbp-0x8],rdi
10:     48 89 75 f0             mov    QWORD PTR [rbp-0x10],rsi
14:     48 89 55 e8             mov    QWORD PTR [rbp-0x18],rdx
18:     48 8b 55 e8             mov    rdx,QWORD PTR [rbp-0x18]
1c:     48 8b 4d f0             mov    rcx,QWORD PTR [rbp-0x10]
20:     48 8b 45 f8             mov    rax,QWORD PTR [rbp-0x8]
24:     48 89 ce                mov    rsi,rcx
27:     48 89 c7                mov    rdi,rax
2a:     e8 00 00 00 00          call   2f <searchorig+0x2f>
2f:     c9                      leave
30:     c3                      ret

0000000000000031 <linearSearch>:
31:     f3 0f 1e fa             endbr64
35:     55                      push   rbp
36:     48 89 e5                mov    rbp,rsp
39:     48 89 7d e8             mov    QWORD PTR [rbp-0x18],rdi
3d:     48 89 75 e0             mov    QWORD PTR [rbp-0x20],rsi
41:     48 89 55 d8             mov    QWORD PTR [rbp-0x28],rdx
45:     48 c7 45 f8 00 00 00    mov    QWORD PTR [rbp-0x8],0x0
4c:     00
4d:     eb 2a                   jmp    79 <linearSearch+0x48>
4f:     48 8b 45 f8             mov    rax,QWORD PTR [rbp-0x8]
53:     48 8d 14 c5 00 00 00    lea    rdx,[rax*8+0x0]
5a:     00
5b:     48 8b 45 e8             mov    rax,QWORD PTR [rbp-0x18]
5f:     48 01 d0                add    rax,rdx
62:     48 8b 10                mov    rdx,QWORD PTR [rax]
65:     48 8b 45 d8             mov    rax,QWORD PTR [rbp-0x28]
69:     48 39 c2                cmp    rdx,rax
6c:     75 06                   jne    74 <linearSearch+0x43>
6e:     48 8b 45 f8             mov    rax,QWORD PTR [rbp-0x8]
```

Preparatory Steps: Pintool

1. **[0 Marks]** Change directory into **assignment1/tracerorig** folder.

```
prashantnair — ssh prashantnair@ssh.ece.ubc.ca — 130×34
~ — ssh prashantnair@ssh.ece.ubc.ca        ~/Downloads — -bash
ssh-linux4:~/assignment1/algorithms> cd ../tracerorig/
ssh-linux4:~/assignment1/tracerorig> ls
clean_tracer.sh  makefile  makefile.rules  make_tracer.sh  obj-intel64  quick_make.sh  README  tracer.cpp
```

2. **[0 Marks]** Understand the **assignment1/tracerorig** folder.

```
ssh-linux4:~/assignment1/algorithms> cd ../tracerorig/
ssh-linux4:~/assignment1/tracerorig> ls
clean_tracer.sh  makefile  makefile.rules  make_tracer.sh  obj-intel64  quick_make.sh  README  tracer.cpp
ssh-linux4:~/assignment1/tracerorig>
```

3. **[0 Marks]** Understand the **tracer.cpp** file and compile.

```
ssh-linux4:~/assignment1/tracerorig> vim tracer.cpp

/* ================================================================== */
// Global variables
/* ================================================================== */

FILE* out;
bool output_file_closed = false;

/* ================================================================== */
// Add your variables below this
/* ================================================================== */

/* ================================================================== */
// Add your variables above this
/* ================================================================== */
```

```
/* ================================================================== */
// Instrumentation callbacks
/* ================================================================== */

// Is called for every instruction
VOID Instruction(INS ins, VOID *v)
{
        // begin each instruction with this function
        UINT32 opcode = INS_Opcode(ins);
        INS_InsertCall(ins, IPOINT_BEFORE, (AFUNPTR)BeginInstruction, IARG_INST_PTR, IARG_UINT32, opcode, IARG_END);
        //+++++++++++++++++ ADD YOUR CODE BELOW THIS POINT ++++++++++++++++++++++++++//

        //+++++++++++++++++ ADD YOUR CODE ABOVE THIS POINT ++++++++++++++++++++++++++//
        // finalize each instruction with this function
        INS_InsertCall(ins, IPOINT_BEFORE, (AFUNPTR)EndInstruction, IARG_END);
}

VOID Fini(INT32 code, VOID *v)
{
        assert(instrCount == (nonmeminstCount+meminstCount));

        fprintf(out, "instrCount,%ld\n", instrCount);
        fprintf(out, "nonmeminstCount,%ld\n", nonmeminstCount);
        fprintf(out, "meminstCount,%ld\n", meminstCount);
```

```
prashantnair — ssh prashantnair@ssh.ece.ubc.ca — 130×34
~ — ssh prashantnair@ssh.ece.ubc.ca        ~/Downloads — -bash
}


/* ================================================================== */
// Analysis routines
/* ================================================================== */

void BeginInstruction(VOID *ip, UINT32 op_code, VOID *opstring){
        instrCount++;
        if(instrCount%1000000 == 0){
                std::cout << "." << std::flush;
                if(instrCount%10000000 == 0){
                        std::cout << " -- " << instrCount/1000000 << " Million Instructions" << std::endl << std::flush;
                }
        }
}

/* ++++++++++++++++ ADD YOUR FUNCTION (IF ANY) BELOW +++++++++++++++++++ */



/* ++++++++++++++++ ADD YOUR FUNCTION (IF ANY) ABOVE +++++++++++++++++++ */


void EndInstruction()
{

}
```

```
prashantnair — ssh prashantnair@ssh.ece.ubc.ca — 130×34
~ — ssh prashantnair@ssh.ece.ubc.ca        ~/Downloads — -bash
ssh-linux4:~/assignment1/tracerorig> bash quick_make.sh
g++ -Wall -Werror -Wno-unknown-pragmas -D__PIN__=1 -DPIN_CRT=1 -fno-stack-protector -fno-exceptions -funwind-tables -fasynchronous
-unwind-tables -fno-rtti -DTARGET_IA32E -DHOST_IA32E -fPIC -DTARGET_LINUX -fabi-version=2 -faligned-new  -I/ubc/ece/home/pnj/facul
ty/prashantnair/assignment1/pin-dir//source/include/pin -I/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//source/inclu
de/pin/gen -isystem /ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir/extras/stlport/include -isystem /ubc/ece/home/pnj/f
aculty/prashantnair/assignment1/pin-dir/extras/libstdc++/include -isystem /ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-d
ir/extras/crt/include -isystem /ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir/extras/crt/include/arch-x86_64 -isystem
/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir/extras/crt/include/kernel/uapi -isystem /ubc/ece/home/pnj/faculty/prash
antnair/assignment1/pin-dir/extras/crt/include/kernel/uapi/asm-x86 -I/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//e
xtras/components/include -I/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//extras/xed-intel64/include/xed -I/ubc/ece/h
ome/pnj/faculty/prashantnair/assignment1/pin-dir//source/tools/Utils -I/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir/
/source/tools/InstLib -O3 -fomit-frame-pointer -fno-strict-aliasing   -c -o obj-intel64/tracer.o tracer.cpp
g++ -shared -Wl,--hash-style=sysv /ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//intel64/runtime/pincrt/crtbeginS.o -
Wl,-Bsymbolic -Wl,--version-script=/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//source/include/pin/pintool.ver -fab
i-version=2    -o obj-intel64/tracer.so obj-intel64/tracer.o  -L/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//intel6
4/runtime/pincrt -L/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//intel64/lib -L/ubc/ece/home/pnj/faculty/prashantnai
r/assignment1/pin-dir//intel64/lib-ext -L/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//extras/xed-intel64/lib -lpin
-lxed /ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//intel64/runtime/pincrt/crtendS.o -lpin3dwarf  -ldl-dynamic -nost
dlib -lstlport-dynamic -lm-dynamic -lc-dynamic -lunwind-dynamic
ssh-linux4:~/assignment1/tracerorig>
```

4. **[0 Marks]** Refer to Pintool[1] website, write some **example Pintools.** You have a folder called examples, implement Pintools there and compile them.

```
prashantnair — ssh prashantnair@ssh.ece.ubc.ca — 130×34
~ — ssh prashantnair@ssh.ece.ubc.ca        ~/Downloads — -bash
ssh-linux4:~/assignment1/tracerorig> cd ../example/
ssh-linux4:~/assignment1/example> ls
clean_example.sh  example.cpp  make_example.sh  makefile  makefile.rules  obj-intel64  pintool.log  quick_make.sh  README
ssh-linux4:~/assignment1/example>
```

5. **[0 Marks]** Execute your Pintool to "instrument" **assignment1/algorithms/assignment1.bin**. **Note that the first parameter after assignment1.bin is "0" → Indicating Linear-Search**

```
jhwoo36@gattaca:~/ta/cpen411/2023/assignment1_test/tracerorig$ ../pin-dir/pin -t obj-intel64/tracer.so -m 0 -- ../algorithms/assignment1.bin 0
Your Student #
......... -- 10 Million Instructions
......... -- 20 Million Instructions
......... -- 30 Million Instructions
......... -- 40 Million Instructions
......... -- 50 Million Instructions
......... -- 60 Million Instructions
......... -- 70 Million Instructions
......... -- 80 Million Instructions
......... -- 90 Million Instructions
......... -- 100 Million Instructions
......... -- 110 Million Instructions
```

```
prashantnair — ssh prashantnair@ssh.ece.ubc.ca — 130×34
~ — ssh prashantnair@ssh.ece.ubc.ca
ssh-linux4:~/assignment1/tracerorig> cd ../output/
ssh-linux4:~/assignment1/output> vim tracerorig.out
```

```
instrCount,115533212
nonmeminstCount,0
meminstCount,0
branchCount,0
MEMDELAY,2000
NONMEMDELAT,1000
CYCLEVAL-SERIAL,0
CYCLEVAL-HIDEMEM,0
CHANGE,0
OPT-MEMDELAY,2000
OPT-NONMEMDELAT,1000
OPT-CYCLEVAL-SERIAL,0
OPT-CYCLEVAL-HIDEMEM,0
SPEEDUP,nan
SPEEDUP-HIDEMEM,nan
```

6. **[0 Marks]** Add a constant delay = NONMEMDELAY to the CYCLEVALS counter after each instruction. In this example, we do not know which instructions are memory instructions and we assume every instruction to be a non-memory instruction.

```
prashantnair — ssh prashantnair@ssh.ece.ubc.ca — 130×34
~ — ssh prashantnair@ssh.ece.ubc.ca        ~/Downloads — -bash
INT32 Usage()
{
        cerr << "This tool creates helps instrument instructions" << endl << "Percentage Change in Non-Memory Instruction Latency:
 From -90 to +90 with -o" << endl;
        cerr << KNOB_BASE::StringKnobSummary() << endl;
        return -1;
}


/* ===================================================================== */
// Analysis routines
/* ===================================================================== */

void BeginInstruction(VOID *ip, UINT32 op_code, VOID *opstring){
        instrCount++;
        //This is an example, and this is an incorrect assumption
        CYCLEVALS = CYCLEVALS + NONMEMDELAY;

        if(instrCount%1000000 == 0){
                std::cout << "." << std::flush;
                if(instrCount%10000000 == 0){
                        std::cout << " -- " << instrCount/1000000 << " Million Instructions" << std::endl << std::flush;
                }
        }
}

/* ++++++++++++++++ ADD YOUR FUNCTION (IF ANY) BELOW +++++++++++++++++++++ */
```

```
prashantnair — ssh prashantnair@ssh.ece.ubc.ca — 130×34
~ — ssh prashantnair@ssh.ece.ubc.ca        ~/Downloads — -bash
ssh-linux4:~/assignment1/tracerorig> bash quick_make.sh
g++ -Wall -Werror -Wno-unknown-pragmas -D__PIN__=1 -DPIN_CRT=1 -fno-stack-protector -fno-exceptions -funwind-tables -fasynchronous
-unwind-tables -fno-rtti -DTARGET_IA32E -DHOST_IA32E -fPIC -DTARGET_LINUX -fabi-version=2 -faligned-new  -I/ubc/ece/home/pnj/facul
ty/prashantnair/assignment1/pin-dir//source/include/pin -I/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//source/inclu
de/pin/gen -isystem /ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir/extras/stlport/include -isystem /ubc/ece/home/pnj/f
aculty/prashantnair/assignment1/pin-dir/extras/libstdc++/include -isystem /ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-d
ir/extras/crt/include -isystem /ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir/extras/crt/include/arch-x86_64 -isystem
/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir/extras/crt/include/kernel/uapi -isystem /ubc/ece/home/pnj/faculty/prash
antnair/assignment1/pin-dir/extras/crt/include/kernel/uapi/asm-x86 -I/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//e
xtras/components/include -I/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//extras/xed-intel64/include/xed -I/ubc/ece/h
ome/pnj/faculty/prashantnair/assignment1/pin-dir//source/tools/Utils -I/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir/
/source/tools/InstLib -O3 -fomit-frame-pointer -fno-strict-aliasing   -c -o obj-intel64/tracer.o tracer.cpp
g++ -shared -Wl,--hash-style=sysv /ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//intel64/runtime/pincrt/crtbeginS.o -
Wl,-Bsymbolic -Wl,--version-script=/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//source/include/pin/pintool.ver -fab
i-version=2    -o obj-intel64/tracer.so obj-intel64/tracer.o  -L/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//intel6
4/runtime/pincrt -L/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//intel64/lib -L/ubc/ece/home/pnj/faculty/prashantnai
r/assignment1/pin-dir//intel64/lib-ext -L/ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//extras/xed-intel64/lib -lpin
-lxed /ubc/ece/home/pnj/faculty/prashantnair/assignment1/pin-dir//intel64/runtime/pincrt/crtendS.o -lpin3dwarf  -ldl-dynamic -nost
dlib -lstlport-dynamic -lm-dynamic -lc-dynamic -lunwind-dynamic
ssh-linux4:~/assignment1/tracerorig>
```

```
jhwoo36@gattaca:~/ta/cpen411/2023/assignment1_test/tracerorig$ ../pin-dir/pin -t obj-intel64/tracer.so -m 0 -- ../algorithms/assignment1.bin 0
Your Student #
......... -- 10 Million Instructions
......... -- 20 Million Instructions
......... -- 30 Million Instructions
......... -- 40 Million Instructions
......... -- 50 Million Instructions
......... -- 60 Million Instructions
......... -- 70 Million Instructions
......... -- 80 Million Instructions
......... -- 90 Million Instructions
......... -- 100 Million Instructions
......... -- 110 Million Instructions
```

```
prashantnair — ssh prashantnair@ssh.ece.ubc.ca — 130×34
~ — ssh prashantnair@ssh.ece.ubc.ca
ssh-linux4:~/assignment1/tracerorig> cd ../output/
ssh-linux4:~/assignment1/output> vim tracerorig.out
```

```
Terminal
instrCount,115533212
nonmeminstCount,0
meminstCount,0
branchCount,0
MEMDELAY,2000
NONMEMDELAT,1000
CYCLEVAL-SERIAL.115533212000
CYCLEVAL-HIDEMEM,0
CHANGE,0
OPT-MEMDELAY,2000
OPT-NONMEMDELAT,1000
OPT-CYCLEVAL-SERIAL,0
OPT-CYCLEVAL-HIDEMEM,0
SPEEDUP,inf
SPEEDUP-HIDEMEM,nan
```

Remove any modification to tracer.c and proceed to the next step.
Essentially remove the line CYCLEVALS = CYCLEVALS + NONMEMDELAY.
Note: Your Pintool should not execute longer than 5 mins on the server. If it takes longer to execute, we will give you 0 marks. Please write efficient programs and Pintools.

Evaluated Steps [5 marks]: Algorithm and Pintool

1. **[0.5 Marks]** Write a Pintool in **assignment1/traceorig/** to count the number of instructions in **assignment1/algorithms/assignment1.bin** for Linear Search.

- This should already be implemented in the **preparatory example.**

2. **[1 Marks]** Add a function in this Pintool to count the number of memory and non-memory instructions in **assignment1/algorithms/assignment1.bin** for Linear Search.
   - "**meminstCount" and "nonmeminstCount"** variables must be updated in this example.
   - **Note:** There is an assertion and if (**meminstCount + nonmeminstCount) != instrCount**, your program will fail. It is non-trivial to count memory instructions. You will need to understand the functions INS_IsMemoryRead() and INS_IsMemoryWrite() for this. These functions are described here: Memory Read and Write Functions
   - **Additionally:** Please refer to an example here: Memory Traces
3. **[0.5 Marks]** Add a function in this Pintool to count the number of branch instructions in **assignment1/algorithms/assignment1.bin** for Linear Search.
   - The "**branchCount"** variable needs to be updated in this example.
   - **Note:** It is non-trivial to branch instructions. You will need to understand the functions INS_IsBranch() for this. These functions are described here: Branch Count Function

4. **[0.5 Marks]** Assuming all instructions are executed one after another with their own delays, compute the **total execution time** required to execute **assignment1/algorithms/assignment1.bin** for Linear Search. For this exercise, the Pintool is pre-filled with delays (CPI) for memory and nonmemory instructions.
   - Your **CYCLEVALS** variable needs to be *repeatedly* updated in this example. This needs to be done by using the values of **MEMDELAY** for memory instructions and **NONMEMDELAY** for non-memory instructions.
   - The output file, tracerorig.out will reflect this number.

5. **[0.5 Marks]** Assuming all memory instructions do not show any latency or block any other instructions (*perhaps due to some computer architecture optimization)*, compute the **total execution time** required to execute **assignment1/algorithms/assignment1.bin** for Linear Search. For this exercise, the Pintool is pre-filled with delays (CPI) for memory and nonmemory instructions.
   - Your **CYCLEVALP** variable needs to be repeatedly updated in this example.

6. **[0.5 Marks]** You can change the latency (CPI) of non-memory instructions by +90% to -90%. **Repeat steps 3 and 4**. On canvas, enter this value (*such as 10%, 20%, -50%, etc.*) wherein the **speedup as compared to step 3 just crosses 1.2**. This has trade-offs, a 50% increase in non-memory instruction CPI causes a 50% decrease in memory instruction CPI.

   - You **OPTCYCLEVALP** and **OPTCYCLEVALS** variables need to be *repeatedly* updated in this example. This needs to be done by using the values of **MEMDELAY** for memory instructions and **NONMEMDELAY** for non-memory instructions.
   - The input parameter to your pintool "-m" is used to pass the change in latency.

```
ssh-linux4:~/assignment1/tracerorig> ../pin-dir/pin -t obj-intel64/tracer.so -m -20 -- ../algorithms/assignment1.bin 0 Your Student #
```

   - Remember, you need to report the value used when your SPEEDUP in your output file **just surpasses 1.2**.

- **This value needs to reflect in the output file ../output/traceorig.out. The entry "CHANGE" should show this value. Make sure you do not change this value after you converge on the right value. If you change "-m" and run your experiment again, your "CHANGE" value in the output folder would be updated and overwritten.**

7. **[0.5 Marks]** Change the directory into **assignment1/algorithms** folder and implement your searching algorithm – in searchnew.c and searchnew.h.

   - You can only edit these files, and you can add any number of additional functions in searchnew.h and searchnew.c
   - This algorithm needs to **PASS**. After you execute bash runall.sh in this folder, check the ../output/sortnew.txt

8. **[0.5 Marks]** Change directory into **assignment1/tracernew** folder. Write a Pintool in its **trace.cpp** to count the number of instructions in **assignment1/algorithms/assignment1.bin** for <your sorting algorithm>.

```
ssh-linux4:~/assignment1> cd tracernew/
ssh-linux4:~/assignment1/tracernew> vim tracer.cpp
ssh-linux4:~/assignment1/tracernew> ../pin-dir/pin -t obj-intel64/tracer.so -m 0 -- ../algorithms/assignment1.bin 1
```

Your Student #

9. **[0.5 Marks]** Ensure that <your sorting algorithm> executes at least 20x lower total instructions as compared to your original Linear-Search.

   - You can check the output folder/tracernew.out file for the total instructions executed.

Submission

To submit, please execute the following command within **assignment1 folder.**

```
jhwoo36@ssh-linux3:~/assignment1$ ls
algorithms  create_submitarchive.sh  example  output  pin-dir  tracernew  traceorig
jhwoo36@ssh-linux3:~/assignment1$ ./create_submitarchive.sh
algorithms/
algorithms/assignment1.o
algorithms/searchnew.h
algorithms/searchoriginal.h
algorithms/searchoriginal.c
algorithms/Makefile
algorithms/README
algorithms/searchnew.c
algorithms/runall.sh
output/
tracernew/
tracernew/tracer.cpp
tracernew/clean_tracer.sh
tracernew/quick_make.sh
tracernew/obj-intel64/
tracernew/obj-intel64/tracer.so
tracernew/obj-intel64/tracer.o
tracernew/makefile
tracernew/README
tracernew/make_tracer.sh
tracernew/makefile.rules
traceorig/
traceorig/tracer.cpp
traceorig/clean_tracer.sh
traceorig/quick_make.sh
traceorig/obj-intel64/
traceorig/obj-intel64/tracer.so
traceorig/obj-intel64/tracer.o
traceorig/makefile
traceorig/README
traceorig/make_tracer.sh
traceorig/makefile.rules
jhwoo36@ssh-linux3:~/assignment1$ ls
algorithms  create_submitarchive.sh  example  output  pin-dir  submission.tar.gz  tracernew  traceorig
jhwoo36@ssh-linux3:~/assignment1$
```

You can then run **rsync** from your local computer and download the submission file.

```
Assignment-1 — -bash — 130×34
~ — ssh prashantnair@ssh.ece.ubc.ca        ~/Work/CPEN411/2022/Assignment-1 — -bash
MacBook-Pro-10:Assignment-1 prashantnair$ rsync -rtv prashantnair@ssh.ece.ubc.ca:~/assignment1/submission.tar.gz .
receiving incremental file list
submission.tar.gz

sent 43 bytes  received 1,848,394 bytes  1,232,291.33 bytes/sec
total size is 1,847,849  speedup is 1.00
```

Please upload **ONLY** the **submission.tar.gz on Canvas**. Make sure it has the updated algorithms, tracer.cpp, and output files. If we find stale files and execute those, you can get "0" marks.

Thus, be extremely careful and double check if your outputs, source files, etc. are the right ones! We have ~100 students in this course, and we will not re-evaluate your Assignment if you submit incorrect or stale work.

References:

[1] https://software.intel.com/sites/landingpage/pintool/docs/98484/Pin/html/index.html
[2] https://brew.sh/
[3] https://cyberduck.io/