CS614 Spring25 End Sem Marking Scheme

Q1.

(A) 2 and 4 are false.

- 0.5 marks for each. Deduct 0.5 if mentioned any others (Total Marks = 1)

(B)
```
STACK[p] = {o1}
STACK[r] = {o1}
STACK[q] = {o3}
HEAP[o1,f] = {o2}
```

- 0.5 marks for each correct PTA relation. (Total Marks = 2)

(C) Define strong update and weak update with an example.

- 2 marks for correct definition with an example.

Tell whose approach is correct and why?

- 1 mark. (Total Marks = 3)

(D) Give a code where you show when can't you guarantee that formal parameters do not point to any local objects.

for eg:

```
main() {
    . . .
    o1 = o2.foo(o3);
    o4 = o1.foo(o1);
    . . .
}
A foo(p1) {
    . . .
    a = new A(); // o5
    return a;
}
```

In the second call `p1` → `o5` which was created in `foo` which by just analyzing the method `foo` can't be guaranteed.

- 1 mark each for correct code and explanation. (Total Marks = 2)

Q2.

(A) First one can be. Second one cannot. [2]
Second one is imprecise because the iteration indices do not overlap. [1]

(B) lines is a reference variable on stack, pointing to an array object on the heap. Each index of this array points to a Line object on the heap. Each Line object points to two Point objects on the heap: one through src and another through dst. Each of these Point objects contains integer fields x and y inside. 2 marks if all drawn correctly.

In the 0th iteration, the first miss would be for the lines object. Then second for the Line object pointed to by lines[i], then one for lines[i].src. This fills up the cache. Then at line 4 lines[i].dst has a miss, which would result in evicting one of the previous objects. In the next iteration, depending on what you evicted, you may again incur <4 misses. Hence the max is 4. Marks are 1.5 if all of the reasoning is correct; 1 if only the initial lines array object is missed (we know ideally it should have been 0.25 because that object might be fetched only initially, but we have been uniform and kept mark boundaries at 0.5.

The two fields x and y can be kept inside the Line object itself, avoiding indirections for src and dst. This will also mean we use up 8 bytes judiciously, thus reducing cache misses. Marks are 1.5 if fully correct.

Q3.

A. C::what, C::who, Loop 23.

B. Unreachable code elimination @ what, nothing @ who and devirtualization @ Loop 23 (any other opts which derive from these speculations such as inlining are also marked as correct, whereas non-speculative optimizations ignored).

C. C::what, C::who, Loop 23, Main::how, C/E::what. Devirtualization @ 29.

D. 1 mark for strategy/methodology for picking/creating versions. 1 mark for justification of usefulness and 1 mark for VM support to allow.