PL midterm solution

1 Java specifies the sizes of its primitive types.
Java adopts hybrid implementation.

2 Correct wording: C++ is usually implemented by compilation.
Comment
Theoretically, any language may be compiled or interpreted.
"C++ is a compiled language" is purely due to common implementation practice.

3 Pro Help communication between subprograms
Con Hard to read
Must be dynamically type-checked
May take a longer time to search a variable
Variables names must be stored and compared.

4 Pro Flexible: Variables don't have fixed types
Con Type unsafe: Runtime type errors
Inefficiency: Type checking code at run time

5 a) First, string; then, number
b) The value of **`f(10)`** is the number 3628800.
The value of **`f("10")`** is the string "1".

6 The occurrence of **`static`** in File1.cpp.
Replace
```
static int f() { return x; }
```
in File 1.cpp by
```
namespace {  int f() { return x; } }
```

7 Line 6 **`fact(n-1)`** Error: Unexpected array reference
Detected by semantic analyzer
Line 7 **`end`** Error: **`END IF`** statement expected
Detected by syntax analyzer

8
```
1 3 5 7
2 4 6 8
```

9 a) **00**

b) **11**

c) **02**

d) **12**

10 a) **@_**

b) **$i<@_** or **$i<=$#**

11 a) **((a . c) (a . b) (c . b))**

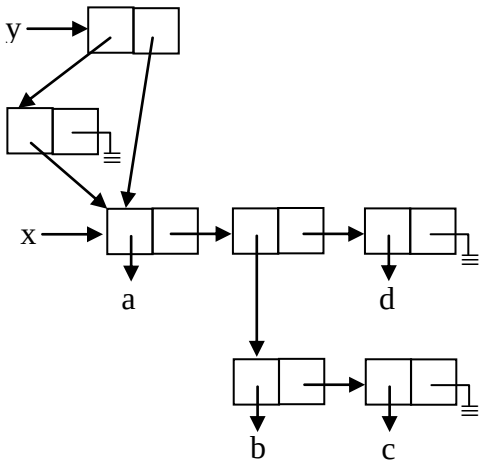
b) **(cons `(,a . ,b) (hanoiAPS (- n 1) c b a acc))**

Initial value of **acc** = **'()**

c)
```
(lambda (v)
  (hanoiCPS (- n 1) c b a
            (lambda (w) (con (append v (cons `(,a . ,b) w))))))
```

Initial value of **con** = **(lambda (v) v)**

12 a)



b) 1 **(((a (b c) d)) a (b c) d)**

2 **(b c)**

13 a) Both space and time complexities are $O(n)$.

To see why, note that the best-case insertion occurs when the element is going to be inserted into the front of the list. In that case, only one cons cell is allocated by the code **`(cons x ys)`**, which in effect copies the element **`x`**.

Therefore, the best-case input of the insertion sort is a list whose elements are in increasing order.

For example, $(1\ 2\ 3 \cdots n)$ is a best-case input.

For the best-case space complexity, let

$s(n)$ = # of cons cells allocated by isort in the best case on sorting a list of n elements

then

$s(0) = 0$

$s(n) = s(n-1) + 1$

Clearly, $s(n) = O(n)$

So is the best-case time complexity.

b) Space complexity $O(1)$

$\because$ No copy is necessary.

Time complexity $O(n)$

$\because$ Just like **`isort`**, each imperative-style insertion takes $O(1)$ time in the best case. Sorting n elements then takes $n \times O(1) = O(n)$ total time in the best case.

14 a)

$$\frac{\dfrac{\text{fix} : t1 \quad f : t2}{f : t2 \qquad \text{fix } f : t3}}{f\ (\text{fix } f) : t4}$$

$$\text{fix} = \lambda f.\ f\ (\text{fix } f) : t1$$

We have the following equations:

$t1 = t2 \to t3$ (1)

$t2 = t3 \to t4$

$t1 = t2 \to t4$ (2)

It follows from (1) and (2) that t3 = t4.

Hence,

$t1 = t2 \to t3 = (t3 \to t4) \to t3 = (t3 \to t3) \to t3$

b) Y contains the self application xx, which, as mentioned in class, is untypable in ML.

15 a) $\$\ f = \underbrace{£££££££££££££££££££££££££}_{26\ £\text{'s}}\ f$ (1)

$= f\ (\underbrace{£££££££££££££££££££££££££}_{26\ £\text{'s}}\ f)$ (2)

$= f\ (\$\ f)$ by (1)

b) From part a), we see in step (2) that the argument to f has to be delayed.
Thus, for eager evaluation, define

$£ = \underbrace{\lambda abcdefghijklmnopqstuvwxyzr}_{26\ \text{English letters (Letter r at the end)}}.r(\lambda z.\ \underbrace{\text{thisisafixedpointcombinator}}_{27\ \text{letters}}\ z)$

where the red-colored letter z may be replaced by any letter that does not appear in the expression thisisafixedpointcombinator.
The definition of $ remains unchanged.

c) $\$\ \lambda fix.\lambda f.f\ (fix\ f)$

16 a) compile ((λx.x 2 3) +)
⇒ compile (λx.x 2 3) (compile +)
⇒ abstract x (compile (x 2 3)) +
⇒ abstract x (compile (x 2) (compile 3)) +
⇒ abstract x (compile x (compile 2) (compile 3)) +
⇒ abstract x (x 2 3) +
⇒ S (abstract x (x 2)) (abstract x 3) +
⇒ S (S (abstract x x) (abstract x 2)) (abstract x x) +
⇒ S (S I (K 2)) (K 3) +

b) S (S I (K 2)) (K 3) +
= S I (K 2) + (K 3 +)
= I + (K 2 +) (K 3 +)
= + (K 2 +) (K 3 +)
= + 2 (K 3 +)
= + 2 3
= 5