Principles of Programming Languages
Small examination 2

Student ID: Name:

Problem 1 Show the type consistency of the following program fragment, which is written in the subset of C language presented in the lecture, according to (1) and (2).

```
int *p;
int x[3];
p = x;
```

(1) Rewrite the variable declarations `int *p;` and `int x[3];` in the postfix notation presented in the lecture.

(2) Show the type consistency of the assignment expression `p=x` by applying the inference rules to the declarations of `p` and `x` in the postfix notation obtained in (1).

Problem 2 A lambda expression $(\lambda x.\ \lambda y.\ x)\ ((\lambda z.\ z)\ w)$ can be transformed to $(\lambda y.\ w)$ by applying the β reductions. Write the each step of the β reductions. (Although there are more than one sequences of β reductions, write one of them.)

Problem 3 Write the output to the display when executing the following program in C++.

```
#include <stdio.h>
class B {
public:
  virtual char f() { return 'B';}
  char g() { return 'B'; }
  char testF(B *b) { return b->f();}
  char testG(B *b) { return b->g();}
};
class D : public B {
public:
  char f() { return 'D';}
  char g() { return 'D';}
};
int main(void) {
  D *d = new D;
  printf("%c%c\n", d->testF(d), d->testG(d));
  return 0;
}
```

Problem 4 Write the solution (the substitution to the variable `X`) to the query `a(X).` after defining `a`, `b`, `c`, `d`, and `e` in Prolog as follows.

```
a(1) :- b.
a(2) :- e.
b :- !, c.
b :- d.
c :- fail.
d.
e.
```

Problem 5

Show the meaning of the following programs (1) and (2) by using the rules presented in the lecture. Note that the programs are in the small subset of C presented in the lecture. Let the states before executing the programs both to be $\sigma = \{(\mathtt{X}, 3), (\mathtt{Y}, 1), (\mathtt{Z}, 0)\}$.

(1) `Z=(X+4);`

(2) `while(Y){Y=(Y-1);}`

Rules presented in the lecture Typing rules

- Rules for function calls, pointers, arrays

$$\frac{e : \tau[n]}{e[i] : \tau} \qquad \frac{e : \tau()}{e() : \tau} \qquad \frac{e : \tau*}{*\, e : \tau} \qquad \frac{e : \tau[n]}{e : \tau\&}$$

- Rule for assignment operator =, where e is an l-value expresssion and not a constant.

$$\frac{e : \tau \quad e' : \tau}{e = e' : \tau}$$

- Rule for the & operator where the outermost part of τ is not &.

$$\frac{e : \tau}{\&e : \tau\&} \qquad \frac{e : \tau\&}{*\, e : \tau} \qquad \frac{e : \tau * \quad e' : \tau\&}{e = e' : \tau\&}$$

Rules for lambda calculus

- β reductions

$$(\lambda x.M)\ N \xrightarrow[\beta]{} M[N/x]$$

$$\frac{M \xrightarrow[\beta]{} N}{\lambda x.M \xrightarrow[\beta]{} \lambda x.N} \qquad \frac{M \xrightarrow[\beta]{} N}{MP \xrightarrow[\beta]{} NP} \qquad \frac{M \xrightarrow[\beta]{} N}{PM \xrightarrow[\beta]{} PN}$$

- Substitutions

$$\begin{array}{rcl} c[N/x] & = & c \\ x[N/x] & = & N \\ x[N/y] & = & x \quad (x \neq y) \\ (\lambda y.M)[N/x] & = & \begin{cases} \lambda y.M & \textbf{if } x = y \\ \lambda y.(M[N/x]) & \textbf{if } x \neq y,\ y \notin FV(N) \\ \lambda z.((M[z/y])[N/x]) & \textbf{if } x \neq y,\ z \neq x,\ y \in FV(N), \\ & \quad z \notin FV(M),\ z \notin FV(N) \end{cases} \\ (M_1 M_2)[N/x] & = & (M_1[N/x])(M_2[N/x]) \end{array}$$

- Free variables

$$
\begin{array}{rcl}
FV(c) & = & \{\} \\
FV(x) & = & \{x\} \\
FV(\lambda x.M) & = & FV(M) \setminus \{x\} \\
FV(M_1 M_2) & = & FV(M_1) \cup FV(M_2)
\end{array}
$$

Operational semantics for the small subset of C

- Rules for arithmetic expressions
  - Sequences of numbers: $< n, \sigma > \to m$ where m is an integer represented by the sequence of numbers n in the decimal representation.
  - Variables: $< x, \sigma > \to \sigma(x)$
  - Addition:

$$\frac{< a_1, \sigma > \to m_1 \quad < a_2, \sigma > \to m_2}{< (a_1 + a_2), \sigma > \to m} \ (m \text{ is the sum of } m_1 \text{ and } m_2.)$$

  - Subtraction:

$$\frac{< a_1, \sigma > \to m_1 \quad < a_2, \sigma > \to m_2}{< (a_1 - a_2), \sigma > \to m} \ (m \text{ is the difference of } m_1 \text{ and } m_2.)$$

  - Multiplication:

$$\frac{< a_1, \sigma > \to m_1 \quad < a_2, \sigma > \to m_2}{< (a_1 * a_2), \sigma > \to m} \ (m \text{ is the product of } m_1 \text{ and } m_2.)$$

- Rules for statements
  - Assignments:

$$\frac{< a, \sigma > \to m}{< x = a;, \sigma > \to \sigma[m/x]}$$

    where $\sigma[m/x]$ is defined as follows.

$$(\sigma[m/x])(y) = \begin{cases} m & \textbf{if } y = x \\ \sigma(y) & \textbf{if } y \neq x \end{cases}$$

  - Sequences:

$$\frac{< c_1, \sigma > \to \sigma_1 \quad < c_2, \sigma_1 > \to \sigma_2}{< c_1 \ c_2, \sigma > \to \sigma_2}$$

  - **while** statements:

$$\frac{< a, \sigma > \to 0}{< \textbf{while } (a) \ \{c\}, \sigma > \to \sigma}$$

$$\frac{< a, \sigma > \to m \quad < c, \sigma > \to \sigma_1 \quad < \textbf{while } (a) \ \{c\}, \sigma_1 > \to \sigma_2}{< \textbf{while } (a) \ \{c\}, \sigma > \to \sigma_2} \ (\textbf{if } m \neq 0)$$