COMS W4115
Programming Languages and Translators
Homework Assignment 3

Prof. Stephen A. Edwards
Columbia University

Due Monday, December 5th, 2016
at 4:10 PM

Submit solution on paper unless you're a CVN student. Include your name and Columbia ID (e.g., se2007).

Do this assignment alone. You may consult the instructor and the TAs, but not other students.

1. (20 pts.) For the following C array,

```
short a[3][4];
```

assume you are working with a 32-bit little-endian processor with the usual alignment rules (e.g., a Pentium).

   (a) Show how its elements are laid out in memory.

   (b) Write an expression for the byte address of `a[i][j]` in terms of a, i, and j.

   (c) Verify parts a) and b) by writing a small C program that allows you to **test your hypothesis**. Examine the assembly language output with the C compiler's -S flag (e.g., `gcc -O -S array.c`). Such a program should be simple and contain and access such an array, but not be so simple that the compiler can optimize most of it away. **Turn in an annotated assembly listing** that explains how it verifies your hypothesis. Make sure the assembly listing is no more than about 40 lines, either by simplifying your program or trimming the output.

2. (20 pts.) For a 32-bit little-endian processor with the usual alignment rules, show the **memory layout** and **size in bytes** of the following three C variables.

```
union {
  struct {
    char a; /* 8-bit */
    int b;  /* 32-bit */
  } s;
  struct {
    char c;  /* 8-bit */
    short d; /* 16-bit */
  } t;
} u1;
```

```
struct {
  char a;
  short b;
  int c;
  char d;
} s1;
```

```
struct {
  char a;
  short b;
  char c;
  short d, e, f;
  char g;
} s2;
```

3. (20 pts.) Draw the layout of the stack just before *bar* is called in *foo*. Indicate storage for function arguments, local variables, return addresses, and stored frame pointers. Indicate where the stack and frame pointers point.

```
void bar(int x, int y, int z);
void foo(int a, int b, int c)
{
  int d, e;
  bar(4, 2, 7);
}
```

4. (20 pts.) Draw the layouts of s1 and s2 the virtual tables for the Ellipse and Square classes. Indicate how the runtime decides to call the appropriate *area* function for *s1* in *main*.

```
public class Shape {
   double x, y;
   public double area() { ... }
}

class Ellipse extends Shape {
   private double height, width;
   public double area() { ... }
}

class Square extends Shape {
   private double width;
   public double area() { ... }
}

public class Main {
  public static void main() {
    Shape s1 = new Square(10, 3, 15);
    Shape s2 = new Ellipse(3, 8, 5, 7);
    System.out.println( s1.area() );
  }
}
```

5. (20 pts.) For each lambda term, show the steps in deriving its normal form or explain why it has none.

   (a) $(\lambda x\,.+x\,2)\;3$

   (b) $(\lambda y\,.\,y)$

   (c) $(\lambda z\,.\,z\,z)\;(\lambda z\,.\,z\,z)$

   (d) $(\lambda x\,.\,x)\;(\lambda x\,.\,x)\;(\lambda x\,.\,x)$

   (e) $(\lambda x\,.\,\lambda y\,.+y\,3)\;((\lambda z\,.\,z\,z)\;(\lambda z\,.\,z\,z))\;((\lambda x\,.+x\,3)\;36)$