CS 316 Exam 2

Spring, 2003

Name:

Problem 1: (10 pts.) What do the following three loops compute?

```
        int i = 0;
        while (a[i++] != 0);
```

and

```
        int i;
        for (i = 0; a[i] != 0; i++);
```

and

```
        for (int i = 0; a[i] != 0; i++);
```

Are the three loops equivalent? That is, can you always replace the first loop with either of the other two?

Problem 2: (10 pts.) Recall that C has two kinds of conditionals, a conditional statement and a conditional expression. But C has only one kind of for-loop, a for-loop statement. Suppose that we want to make C more "orthogonal" and so we propose adding a for-loop expression to the language. Propose two different (seemingly reasonable) definitions for the return value of a for-loop expression. What kinds of problems might there be with implementing your proposals? Be sure to consider cases like a for-loop exited by a `break` (like) statement and for-loops with possibly empty expressions in the for-loop (like `for ( ; exp0; );`). (Hint: For one of the proposals, think about how functions work).

Problem 3: (10 pts.) Explain in detail exactly what the C expression

```
i + j / k
```

means and what its type is when the variables have the following data types. Be sure to specify any implicit type conversions.

- (a) `int i, j, k;`

- (b) `int i, j; double k;`

- (c) `double i; int j, k;`

Problem 4: (10 pts.) Suppose that we have two C arrays:

```
int x[10];
int y[10];
```

Why won't the assignment `x=y` compile in C? Can the declarations be fixed so that the assignment will work? If so, then what exactly does the assignment do?

Problem 5: (10 pts.) **(a)** Given the C declarations

```
struct
{  int i;
   double j;
} x, y;

struct
{  int i;
   double j;
} z;
```

the assignment `x=z` generates a compilation error, but the assignment `x=y` does not. Why?

(b) Give two ways to fix the code in part (a) so that `x=z` works. Which way is better and why?

Problem 6: (10 pts.) In an if-statement

```
if (exp1) exp2; else exp3;
```

the types of the expressions `exp2` and `exp3` do not matter at all. On the other hand, in

```
exp4 ? exp5: exp6
```

the types of `exp5` and `exp6` do matter. In what way do the types matter in the conditional expression? Explain the reason for this difference in the two kinds of conditionals.

Problem 7: (10 pts.) Suppose that in C we try to write a short-circuit version of an `and` function in the following way.

```
int and (int a, int b)
{  return a ? b : 0;
}
```

Why won't this work?

Problem 8: (10 pts.) The following C program prints two integer values; the first value typically is garbage (explain why) but the second value might be 2. Give a detailed explanation of why the second value could be 2. The second value could also be garbage (but it is much less likely). Give an explanation of why the second value could be garbage.

```
#include <stdio.h>

void p()
{  int y;
   printf("%d\n", y);
   y = 2;
}

int main()
{  p();
   p();
   return 0;
}
```

Problem 9: (10 pts.) Explain in detail the differences between the following three C functions.

```
int functionA()
{  static int i;
   i++;
   return i;
}

int functionB()
{  static int i = 0;
   i++;
   return i;
}

int functionC()
{  static int i;
   i = 0;
   i++;
   return i;
}
```

Problem 10: (10 pts.) Write a function, using C syntax, that will have three different effects, depending on whether the arguments are passed by value, by reference, or by value/result. Give an example of a specific function call that will has three different effects depending on the parameter passing method. Explain the details of how your function works with each kind of parameter passing method.