Name:
SSN :

CSE 428

Fall 1997

Midterm #1

8 October 1997

The exam consists of 5 problems on 5 pages, totaling 100 points. Read each question carefully and use your time judiciously.

Write your name/number on every page.

1. Consider the following grammar: (20 pts)

$$\begin{aligned} E &::= E \text{“}*\text{”} E \mid F \\ F &::= F \text{“}+\text{”} U \mid T \text{“}-\text{”} T \mid T \\ T &::= \text{“}-\text{”} U \mid \text{“}(\text{”} E \text{“})\text{”} \mid U \\ U &::= N \mid Id \end{aligned}$$

in which N and Id are nonterminals which generate integers and identifiers, respectively.

(a) Which of the following strings *cannot* be generated by this grammar:

i. 3 – 4 – X
ii. X ∗ (– (Y + Z))
iii. X + (– Y)
iv. X + Y – Z

(b) Which of the following strings can be generated unambiguously (i.e., they have only one possible parse tree.

i. X ∗ Y + Z
ii. X ∗ 2 ∗ Z
iii. (X – – Y)
iv. X + Y + Z

2. The following program searches a sorted array for the **first** number which occurs twice. (20 pts)
Insert the following assertions:

   (a) the precondition stating that array A[1..n] is sorted (by $\leq$) **and** that some number occurs twice in the array;

   (b) the loop invariant; and

   (c) the postcondition stating that such an element has been found (and at which indices).

   All assertions should be given as formulas in first-order logic. The loop invariant and the negation of the loop condition should imply the postcondition, but you do not need to show this, nor do you need to show that your loop invariant actually is an invariant.

```
{                                                    }

j := 1;
while A[j] ≠ A[j+1] do

{                                                    }

    j := j+1;
end do

{                                                    }
```

3. Consider the following program: (20 pts)

```
program main
  i,j : integer;

  procedure tybalt(a : integer)
  begin
    a := a + i;
    j := a + j;
    write(a,j);
  end tybalt;

  procedure mercutio(i : integer)
  j : integer;
  begin
     j := i + 1;
     tybalt(j);
    write(i,j);
  end mercutio;

begin main
  i := 1;
  j := 3;
  mercutio(j);
    write(i,j);
end main;
```

What is output by this program under call-by-value parameter passing and

(a) static scoping

(b) dynamic scoping

4. Consider the following program: (20 pts)

```
program main
  i,j : integer;

  procedure Benvolio(a,b,c :integer)
  begin
    a := b + c;
    b := i + a;
    write(a,b);
  end Benvolio;

begin main
  i := 1;
  j := 3;
  Benvolio(i,i,j);
    write(i,j);
end main;
```

What is output by this program if **all** parameters are passed using the following (state any assumptions you need to make).

(a) call-by-value

(b) call-by-value-result

(c) call-by-reference

5. Consider the following outline of a program. (20 pts)

```
program main
   procedure stop

        procedure right
        begin right
            left();
        end right;

        procedure left
        begin
            stop();
        end left;

   begin stop
        right();
   end stop;

   procedure start();
   begin start
       stop();
   end start;

begin main
start();
end main;
```

Assume during the execution of this program, the following sequence of procedures calls occur: `main` calls `start`, `start` calls `stop`, `stop` calls `right`, `right` calls `left`, and `left` calls `stop`. At this point there are five activation records on the stack. Name them AR_1 through AR_5, in the order in which they were pushed. Call the area containing the global variables GL.

For each activation record, state which other activation records its dynamic link and static link point to. (The case for AR_1 is given for you.)

AR_1:	SL $= GL$	AR_3:	SL $=$	AR_5:	SL $=$
	DL $= GL$		DL $=$		DL $=$
AR_2:	SL $=$	AR_4:	SL $=$		
	DL $=$		DL $=$		