Programming Languages: Mid-Term Exam

B. Mishra

October 30 1995

Problem 1.[5 pt] What are the values of `x`, `y` and `z` at the end of the following C program?

```
z = 0; x = 1; y = 2;
if (x = 1)
   if (y = 1) z = 1;
else z = 2;
```

☐ This is an erroneous C program.

☐ `x=1`, `y=2` and `z=0`.

■ `x=1`, `y=1` and `z=1`.

☐ `x=1`, `y=2` and `z=2`.

Problem 2.[5 pt] Consider the following FORTRAN program. What is the matrix `V` after the loop is executed?

```
    DO 14 I=1,N
    DO 14 J=1,N
    V(I,J) = (I/J) * (J/I)
 14 CONTINUE
```

■ An identity matrix.

☐ A matrix of all zeroes.

☐ A matrix of all ones.

☐ Complier flags an error.

Problem 3.[5 pt] Consider the following ADA program fragment.

```
type STRING_10_PTR is access STRING(1..10);

S: STRING_10_PTR := new STRING;
```

☐ Allocates a string to the pointer.

☐ Causes a CONSTRAINT-ERROR exception.

■ Compiler detects it as an erroneous program.

☐ Hard to tell what happens.

Problem 4.[5 pt] We are given the following syntax:

```
<S> ::= <A>
<A> ::= <B> + <A> | <D> <C>
<B> ::= <D> | <D> * <B>
<D> ::= x | (<C>)
<C> ::= +x | -x
```

Which of the following strings are valid sentences of the grammar?

(*i*) `<B> + <B> + <B> + x -x`

(*ii*) `x + x + x * x * x * (-x) + x + x -x`

(*iii*) `x * x * x * x * x * x * x`

☐ All three: (i), (ii) and (iii).

☐ Only (ii) and (iii)

☐ Only (iii).

■ Only (ii).

Problem 5.[5 pt] Consider the following program, where `<BINDING>` could be "call-by-value", "call-by-reference", "call-by-value/result" (copy-in-copy-out) or "call-by-name".

```
procedure SWAP( A, B: <BINDING> INTEGER) is
   T: INTEGER;
begin
   T := A;
   A := B;
   B := T;
end
```

The program works correctly for the following modes.

■ "call-by-reference" and "call-by-value/result" only.

☐ "call-by-value" and "call-by-value/result" only.

☐ "call-by-reference" only.

☐ "call-by-reference", "call-by-value/result" and "call-by-name" only.

Problem 6.[15 pt] Assume that you have written a compiler that uses a run-time *display*, separate from the activation stack. Give a quick sketch of the stack and the display when control reaches the positions of the two comments in the following program. Assume that parameters are called by value.

```
program main;
   procedure P(a: integer);
      procedure Q(b: integer);
         begin { a) What are the stack and display now? }
         R(a+2, b+3)
```

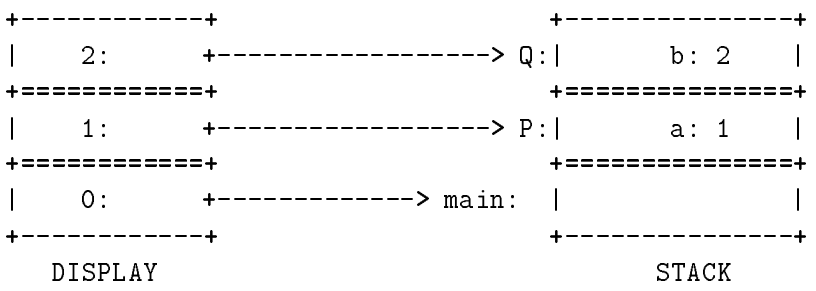
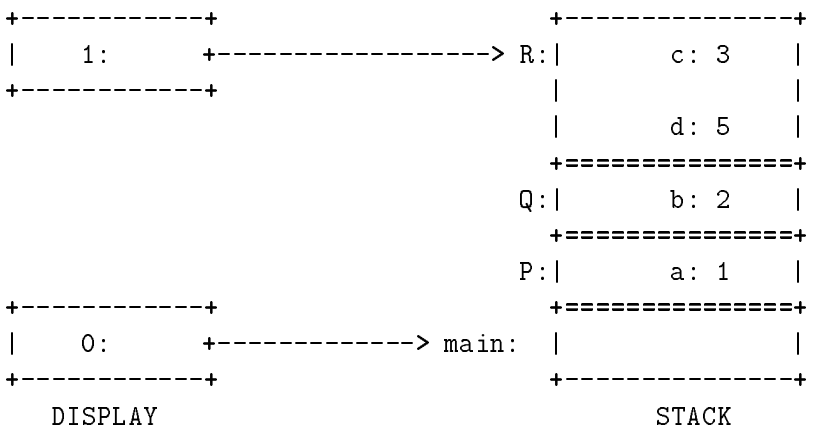
```
      end;
   begin Q(a+1);
   end;
procedure R(c,d: integer):
   begin { b) What are the stack and display now? }
   end;
begin P(1)
end.
```

The stack is a list of activation records (or frames). Each frame contains the parameter and local variables for one procedure invocation. It also contains the return address, and other control informations (not shown here).

The display is mapping from lexical level numbers to stack frames. It is really an auxiliary stack, but since the number of nested procedures is normally small, one can choose a preassigned small maximum and allocate the stack in a fixed location.

Knowing the level number of an identifier, and the offset of a variable in its stack frame, it is fast to access nonlocal variable, even with nested recursive procedures.

a)



b)