

Login name _____

Quiz 1

Name _____

CSE 131B

Signature _____

Spring 2004

Student ID _____

Compilation/Compiler Overview, Names/Scopes/Bindings

1. Give the order of the phases of compilation in a typical compiler as discussed in class

- | | |
|---|---|
| A – Machine-specific code improvement (optional) | B – Scanner (lexical analysis) |
| C – Parser (Semantic analysis/intermediate code gen.) | D – Parser (syntax analysis) |
| E – Machine-independent code improvement (optional) | F – Target code generation |
| G – Source language (for example, C) | H – Target language (for ex., assembly) |

_____ -> _____ -> _____ -> _____ -> _____ -> _____ -> _____ -> _____

2. Consider the following pseudocode:

```

x : integer;           -- global var declaration

procedure set_x ( n : integer )
  x := n;

procedure print_x()
  output( x );        -- print the value of x

procedure one()
  x : integer;        -- local var declaration
  set_x( 1 );
  print_x();

procedure two()
  set_x( 2 );
  print_x();

input( x );           -- reads user input from keyboard into x
if ( x > 5 )
  set_x( 0 );
  one();
  print_x();
  two();
  print_x();
else
  set_x( 3 );
  two();
  print_x();
  one();
  print_x();

```

What does the program output if the user input is the value 4 and the language uses static scoping?

What does the program output if the user input is the value 4 and the language uses dynamic scoping?

(over)

3. Given the following C program, answer the questions using 1 – 4 that best describes the variable in question. If the variable/name is a pointer, then in this context the object it is bound to is the object it refers/points to.

```
int * foo( int x );

int a = 420;
static int b;

int main( void ) {

    static int c = 404;
    int d = 5;
    int *e;

    e = foo( d );
    <<----- B
}
```

- 1 – Name in scope / the object name is bound to is not alive
- 2 – Name is not in scope / the object name is bound to is not alive
- 3 – Name is not in scope / the object name is bound to is alive
- 4 – Name in scope / the object name is bound to is alive

```
int * foo( int x ) {

    int f = 911;
    int *g;
    int *h;
    static int *i;

    g = (int *) malloc( sizeof( int ) );
    h = g;
    i = (int *) malloc( sizeof( int ) );
    free( i );
    <<----- A

    return( &x );
}
```

At the location marked B, the variable/name e _____

At the location marked B, the variable/name i _____

At the location marked B, the variable/name f _____

At the location marked A, the variable/name h _____

At the location marked A, the variable/name b _____

At the location marked A, the variable/name c _____

At the location marked B, which variable/name would be considered a dangling reference _____
(If none, then state NONE)

At the location marked A, which variable/name would be considered a dangling reference _____
(If none, then state NONE)

Where in the C Runtime Environment are the following variables/functions allocated:

x _____ c _____ a _____ where h is pointing _____
b _____ d _____ foo() _____

Is there a memory leak at the location marked B? _____ Why or why not?