UNIVERSITY OF CENTRAL FLORIDA
Department of Computer Science

COP 4020 - PROGRAMMING LANGUAGES
MID TERM EXAM (Fall 2001)
OPEN NOTES AND BOOKS
DURATION: 75 mins

NAME:____

QUESTION 1 (20 points):
Determine the environment at each numbered point in the following C program. The environment at a specific point in the program can be defined as a set of identifiers that can be used at that point. For each identifier in that environment you should also specify the entity that it represents. Please use the syntax *identifier:entity* in order to show the bindings of the identifiers in each environment.

```
#include <stdio.h>
int *pp;
int a=1;

void p1 (int x, int y)
{
    int b=x+y;
    // [1]
    printf("p1:%d %d\n",x,y);
    pp=(int *) malloc(sizeof(int));
    *pp = x;
    *pp = *pp - a;
    if (*pp>0)
       p1(a,b);
}
void p2 (int x)
{
    static int b=2,a;
    // [2]
    printf("p2:%d\n",x);
    a=x-1;
    if (a<b)
      { b+=2; p1(a+2,a+1);free(pp);}
}

main()
{
    int b=0;
    // [3]
    p2(b+1);
    printf("====\n");
    p1(a+1,b+1);
}
```

QUESTION 2 (20 points):
Show the lifetimes of the variables in the following C program. In order to do this, you need to trace the execution of the program until its termination. You can use a time-chart to show when the variables are created and destroyed related to the functions' executions (i.e., call and return).

```
#include <stdio.h>
int *pp;
int a=1;

void p1 (int x, int y)
{
    int b=x+y;
    printf("p1:%d %d\n",x,y);
    pp=(int *) malloc(sizeof(int));
    *pp = x;
    *pp = *pp - a;
    if (*pp>0)
       p1(a,b);
}
void p2 (int x)
{
    static int b=2,a;
    printf("p2:%d\n",x);
    a=x-1;
    if (a<b)
      { b+=2; p1(a+2,a+1);free(pp);}
}

main()
{
    int b=0;
    p2(b+1);
    printf("====\n");
    p1(a+1,b+1);
}
```

You can use the following output as a hint:

```
p2:1
p1:2 1
p1:1 3
=====
p1:2 1
p1:1 3
```

QUESTION 3 (20 points):
Determine the output of the following program for the following parameter passing mechanisms:

- call by reference
- copy mechanism call by value
- call by name

Also rewrite this program in C to simulate the call by reference technique.

```
#include <stdio.h>
int i, a[4];
f(int pa, int pb, int pc)
{
  a[i] = 2;
  i = pa;
  pc++;
  pa = a[i] + pc + pb;
}
main()
{
  a[0]=0; a[1]=0; a[2]=0; a[3]=0;
  i = 1;
  f(a[i],a[i],i);
  printf("%d %d %d %d\n",i,a[0],a[1],a[2],a[3]);
}
```

QUESTION 4 (20 points):

Given the grammar where capital letters are non-terminals and small letters are terminals:

```
C -> B | B c C
B -> A | A a B
A -> T e T | g A | i C i
T -> b | d | f | h
```

- Draw the parse tree for the string "ifehcgfehiabed". Note that the grammar is un-ambigious, therefore there is only a single parse tree corresonding to this string.
- Show the leftmost derivation of this string, starting with non-terminal C.
- Show that if the first "i" is removed from the third production rule, the grammer becomes ambigious (i.e., the last production rules becomes Ci rather than iCi).

QUESTION 5 (20 points):
Find the weakest precondition of the following program phrase:

```
{???}
if (B < A) then B = B + A
           else A = B - A
{B>A}
```

QUESTION 6 (BONUS 10 points):
Write an ML function that returns the summation of the values of the **non-leaf nodes with two children** in a given binary tree using the following datatype definition.

Remember that in ML "|" represents a disjoint union in type decleration and "*" represents the cartesian product.

```
datatype btree = EMPTY | NODE of int*btree*btree;
```