CS 60
Third QUIZ
April 20, 2010
WRITE ALL YOUR ANSWERS ON SPACE PROVIDED.
ANSWER ALL PARTS. TOTAL POINTS ARE 30.

NAME:_____

1 { Circle for each part True or False depending whether or not the statement is true or false. Each question is worth 1 Point}

- { True or **False** } The body of the for statement `for(i = 0; i >= 0; i++)` is executed zero times.
- { **True** or False } The Boolean type in C does not exist.
- { **True** or False } When `x` is declared as an integer, then the value of `x` is 1 after executing the statement `x = (5 - 3) && 1;`
- { **True** or False } The command `mkdir` is used to create a new directory.
- { True or **False** } The command `cd` in UNIX is used to play CDs.
- { True or **False** } The command `gcc o xxx xxx.C` compiles the C++ program stored in file xxx.C and leaves the executable in `xxx`.
- { True or **False** } In C (`printf`) one uses `%d` to print the basic type `double`.
- { **True** or False } By big-endian we mean that the most significant byte has the smallest address byte.
- { **True** or False } The dynamically allocated variables in C are located in an area of memory which in C is called the Heap.
- { **True** or False } The local variable in C are located in an area of memory which in C is called the Stack.
- { True or **False** } The Global and static variables in C are located in an area of memory which in C is called the Stack.
- { True or **False** } A structure (`struct`) in C can only consist of two or more named members of identical type.
- { True or **False** } When the four piles have the following exposed cards 3H 4S 5D 6C then your program for homework three should discard 3H because the four cards form a run.

- { **True** or False } The function `calloc` returns a pointer to a block of memory all of which has been initialized to zero.
- { True or **False** } A structure (`struct`) in C may contain inside it a a structure (`struct`), but not a union (`union`).

2 {Code}

(a) [3 points] What does the following (portion of) code print?

```
int xxx = 12;
int yyy = 30;
int *pi = &xxx;              30
*pi = 31;                    32   30
pi = &yyy;
printf("%d\n",*pi);
pi = &xxx;
*pi=32;
printf("%d %d\n",xxx,yyy);
```

(b) [3 points] What does the following (portion of) code print?

```
int xxx = 13;
int yyy = 15;
int *pi = &yyy;
*pi = yyy;
printf("%d\n",*pi);           15
*pi = 52;                     13    52
printf("%d %d\n",xxx,yyy);
```

c.- [2 points] For the (portion of) code given below clearly indicate what the `printf` command prints.

```
int x,y,a,b;
a = 4;
x = a++;
b = 8;
y = ++b;
printf("%d %d %d %d\n", a, b, x, y);     5  9  4  9
```

(d) [3 points] What does the following code print?

```
#include <stdio.h>

  int main()
  { void xxx(int);
    int yyy(void);
    xxx(2);
    yyy();
    xxx(yyy());
    return 0;
  }

  void xxx(int n)
  {
    printf("Value to be displayed is %d\n", n);
  }

  int yyy(void)                  Value to be displayed is 2
  { static int a = 5;            Value to be displayed is 7
    a++;
    return a;
  }
```

e.- [4 Points] Below you will find two procedures that are stored in different files which are compiled with the command `gcc proc.c func.c`. Clearly indicate the value(s) they print when we execute the a.out executable generated by the above `gcc` command.

proc.c

```
#include <stdio.h>
int func(int, int);
int globX = 3;
extern int globCount;

int main(void)
{
  int x=5, y=7, z;
  z = func(y,x);
  printf("%d %d %d\n",z,globX,globCount);
  z = func(globX,globCount);
  printf("%d %d %d\n",z,globX,globCount);
}
```

func.c

```
int globCount = 4;
extern int globX;

int func(int a, int b)
{
  globCount++;
  globX--;
  return(a+b*globX);
}
```