CS 60
MID-TERM EXAM
July 16, 2009
WRITE ALL YOUR ANSWERS ON SPACE PROVIDED.
answer ALL questions. Total points are 175.
ONLY STUDENTS THAT ARE REGISTERED FOR THIS COURSE
MAY TAKE THIS EXAM.

NAME:______________________________

1 { Circle for each part True or False depending whether or not the statement is true or false. Each question is worth 2 Points}

- { **True** or False } By big-endian we mean that the most significant byte has the lowest address.
- { True or **False** } The function `malloc` 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 another structure (`struct`), but not another structure (`struct`) with a structure (`struc`) inside it.
- { **True** or False } The command `mkdir` is normally used to create a new directory.
- { True or **False** } The command `cd` in UNIX is used to play CDs.
- { True or **False** } Linux is the predecessor of Unix.
- { True or **False** } The command `copy` in Linux is used to copy files.
- { **True** or False } A pointer to a function `XX` in C can be used to call the function `XX` in C.
- { True or **False** } The type `bool` in C does not exist in C++.
- { 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 } A program in C can make a call to `main` in any of its functions, provided `main` is defined before all its functions.

2 {Simple Questions}

a.- [4 points] Suppose that the statement `int x[] = {6,7,8,9}` is executed in a C++ program. Is `x[4]` initialized? What is printed when we execute the command `cout << *x` just after the above statement is executed?

```
   No it is not.  Prints 6.
```

b.- [4 points] Suppose that just after the statement `int *ptr = new int[10]` we execute the statements `ptr[0] = 3; ptr[1] = 4; ptr[2] = 1;`. What is the value of `ptr[0]` just after the above statements are executed? Now suppose that the statement `ptr++;` is executed now. Just after this statement is executed, what is the value of `ptr[1]`?

```
   Value of 3 is printed.  the value of ptr[1] is 1.
```

c.- [4 points] For the C++ code given below clearly indicate what `cout` prints.

```
#include <iostream>
using namespace std;
int main(void)
{  int i,s;
   s = 0;
   i = 2;
   for ( int i = 1; i<= 10; i++, s++)
       { if (s < 4) continue;
              else break;
       }
   cout << i << " " << s << endl;
}

ANS:
2 4
```

d.- [6 points] For the C++ code given below clearly indicate what `cout` prints.

```
#include <iostream>
using namespace std;
class XXX {
public:
  int X(int n=3, int m = 5) {if ( n <= 0 ) return 0;
                             return X(n-1,m)+m;
                            };
};

int main(void)
{
XXX y;
XXX z;
cout << y.X() << " " << z.X(4) << " " << z.X(7,y.X(y.X(),z.X(3))) << endl;
cout << z.X(y.X(5),y.X(10,2)) << endl;
}

Ans:
15 20 1575
500
```

e.- [8 points] In the directory entry

```
-rwxr-xr-- 1 std student 58159 2009-06-28 10:29 temp.txt
```

- Who is the Owner of the file? `std`
- What is the Group of the file? `student`
- What is the name of the file? `temp.txt`
- What is the size of the file in bytes? `58159`
- What permissions does the Owner have on the file? `rwx`
- What permissions does the Group have on the file? `r-x`
- What permissions does every one else have on the file? `r--`
- When was the file last modified? `6/128/2009`
- What would be different if it was a directory instead of a file? `a d in the first -`

f.- [3 points] If the above file temp.txt is the only file in the current directory how do you display the above information about the file but displaying the size in kilobytes?

```
ls -lh
```

g.- [3 points] Briefly explain what `ls *.c | less` does if run in the command line.

```
Pipes to the less command the names of all the *.c files
```

h.- [3 points] Name four commands that will allow you to edit and/or view the contents of a file.

```
   emacs vi cat lpr
```

i.- [3 points] In Linux suppose we have a program called loop that is running and won't quit even with Ctrl-C. How would you find the process id of that file, and how would you force it to quit?

```
   Look for the PID ivia the ps command and then execute the kill -9 <PID> command
```

3 {Makefile}

[8 points] Given the following make file and directory contents, and assuming that printhello.o is up to date, give the exact commands that are executed when make is run.

```
Directory Contents:

-rw-r--r-- 1 pjara grad   61 2009-07-08 01:02 hello.c
-rw-r--r-- 1 pjara grad   23 2009-07-08 01:02 hello.h
-rw-r--r-- 1 pjara grad  101 2009-07-08 01:03 main.c
-rw-r--r-- 1 pjara grad  237 2009-07-08 01:07 Makefile
-rw-r--r-- 1 pjara grad  141 2009-07-08 01:08 printhello.c
-rw-r--r-- 1 pjara grad 2620 2009-07-13 22:08 printhello.o
-rw-r--r-- 1 pjara grad   61 2009-07-08 01:07 world.c
-rw-r--r-- 1 pjara grad   23 2009-07-08 01:03 world.h

Makefile:

CC = gcc
CFLAGS = -g
OBJECTS = hello.o world.o printhello.o main.o
APP = hello

all : $(APP)

$(APP) : $(OBJECTS)
        $(CC) $(CFLAGS) $^ -o $(APP)

%.o : %.c
        rm -f *.o *~ \#* core*

fresh: clean all
```

4 {Pointers}

(a) [6 points] For the C++ code given below clearly indicate what `cout` prints.

```
#include <iostream>
using namespace std;
int main(void)
{
   int x = 8;
   int y = 12;
   int *pi = &y;
   int *qi = pi;
   int *ri = &x;
   (*pi) = y;
   (*qi) = x;
   (*ri) = (*pi);
   pi = &x;
   cout << (*pi) << " " << (*qi) << " " << (*ri) << endl;
}
 ANS
8 8 8
```

(b) [6 points] For the C++ code given below clearly indicate what `cout` prints.

```
#include <iostream>
using namespace std;
int main(void)
{
   int y=2;
   int x=5;
   int *pi=&y;
   int **ppi=&pi;
   pi = &x;
   cout << *pi << " " << **ppi << endl;
   **ppi=&pi;
   pi = &y;
   cout << *pi << " " << **ppi << endl;
}

Ans:
```

```
5 5
2 2
```

(c) [4 points] For the C++ code given below clearly indicate what `cout` prints.

```
#include <iostream>
using namespace std;
int main(void)
{
   int y=6;
   int x=3;
   int& p=y;
   int& q=x;
   p = 5;
   q = 6;
   cout << y << " " << x << endl;
}
ANS
5 6
```

5 {Other Simple Questions}

a.- [4 points] For the C code given below clearly indicate what `printf` prints.

```
#include <stdio.h>
int main(void)
{
        int i=1;
        {
                int i;
                i = 2;
                i++;
                if (i == 2) printf(" NO\n");
                else printf("%d\n",i);
        }
        if (i == 3) printf(" NO\n");
                    else printf("%d\n",i);
}

ANS:
3
1
```

b.- [6 points] For the C code given below clearly indicate what `printf` prints.

```
#include <stdio.h>

void sap(int *x, int *y, int z)
{
  int temp = *y;
  *y = z;
  z = *x;
  *x = temp;
}
int main(void)
{
  int x = 12;
  int y = 24;
  int z = 64;
  sap(&x, &y, z);
  printf("The value of x is %d, y is %d and z is %d\n",x,y,z);
}
```

```
ANS:
The value of x is 24, y is 64 and z is 64
```

c.- [4 points] For the C++ code given below clearly indicate what `cout` prints.

```
#include <iostream>
using namespace std;
void func(int& x)
{
  x = 0; // Initialize
  /********* Function *************
  //       for Wed Quiz          */
  ////////////////////////////////
  x = 2005; /* Change value */
  cout << "x is " << x << endl;
}
int main(void)
{
int z = 123;
func(z);
cout << "z is " << z << endl;
}
```

```
ANS:
x is 2005
z is 2005
```

d.- [6 points] For the C++ code given below clearly indicate what `cout` prints.

```
#include <iostream>

void sap(int& x, int *y, int z)
{
  int temp = *y;
  *y = z;
  z = x;
  x = temp;
}
int main(void)
{
  int x = 64;
  int y = 24;
  int z = 12;
  sap(x, &y, z);
  cout << "The value of x is " << x << " y is "<< y << " and z is " << z << endl;
}

ANS:
The value of x is 24, y is 12 and z is 12
```

6 {More Questions}

a.- [6 points] For the C++ code given below clearly indicate what `cout` prints.

```
#include <iostream>
using namespace std;
int main(void)
 {int x = 9;
  int b = 4;
  cout << b << " " << x << endl;
  { int x = 5;
    int b;
    b = ++x;
    cout << b << " " << x << endl;
    x++;
  }
  cout << b << " " << x << endl;
}

ANS:
4 9
6 6
4 9
```

b.- [4 points] For the C++ code given below clearly indicate what `cout` prints.

```
#include <iostream>
using namespace std;
int main(void)
 {int *p = new int[3];
  p[0] = 20; p[1] = 31; p[2] = 55;
  cout << p[0] << " " << p[1] << endl;
  p++;
  cout << p[0] << " " << p[1] << endl;
  p--;
  delete [] p;
 }

ANS:
20 31
31 55
```

(c) [4 points] For the C code given below clearly indicate what `printf` prints.

```
#include <stdio.h>
int main(void)
```

```
{
  int num = 3;
  int arr[] = {7, 6, 5, 4, 3, 2, 1};
  int *parr = arr;
  parr += 2;
  printf("%d %d\n",parr[2],arr[2]);
  num *= *parr;
  printf("%d\n",num);
}

ANS:
3 5
15
```

(d) [4 points] For the C code given below clearly indicate what `printf` prints.

```
#include <stdio.h>
#include <stdlib.h>
int main(void)
{
  struct stuff {
    char *a;
    int *b;
    double c[2];
    long double *d;
    } var;
  int num = sizeof(var);
  printf("%d\n",num);
  var.b = (int *) malloc(5*sizeof(int));
  num = sizeof(var);
  printf("%d\n",num);
  num = sizeof(int);
  printf("%d\n",num);
  num = sizeof(var.c);
  printf("%d\n",num);
}
```

```
ANS:
28
28
4
16
```

(e) [3 points] For the C code given below clearly indicate what `printf` prints.

```
#include <stdio.h>
int main(void)
{
int num = 1;
int val = 0;
switch (num++) {
  case 0 : val += 9;break;
  case 1 : val += 3;
  case 2 : val += 2; num++;
};
printf("%d %d\n",num,val);
}
```

ANS:

3 5

(f) [4 points] This C code is intended to initialize an array of values. Correct any errors (or state that there are none):

```
int i, j, arr[5][8];
for (i=0; i<=5; i++);
  for (j=0, j<=8, j++);
    arr[i,j] = 0;
```

(g) [4 points] Name the four member functions that are automatically created for you when you define a C++ class (if you don't define them yourself).

(h) [6 points] Clearly indicate what the following C++ program stored in files main.C and func.C print when they are compiled together and executed.

```
file main.C
------
#include "stdio.h"
int func(int, int);
int globX = 20;
extern int globCount;

int main(void)
{
  int x=8, y=2, z;
  z = func(x,y);
  cout << z << " " << globX << " " << globCount << endl;
  return(globCount);
}

file func.C
------
int globCount = 35;
extern int globX;


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

7 {HW4 Code}

[20 points] Write a simple C function that will take in a set of 20 cards, and find and print the highest and the lowest rank card among the 20 cards. The allowable ranks (in order) are : 2, 3, 4, 5, 6, 7, 8, 9, T, J, Q, K, A. The lowest rank is “2”, then “3”, and so till “A”. Print out the number of cards (among the 20 cards) that are not the highest nor the lowest ranked cards (among the 20 cards). You may modify the map function given below and use it in your program.

```
int Map(struct Card card)
{ int temp;

  switch(card.rank) {
    case(’T’): temp=10; break;
    case(’J’): temp=11; break;
    case(’Q’): temp=12; break;
    case(’K’): temp=13; break;
    case(’A’): temp=1; break;
    default:   temp=card.rank-’0’;
   }
  return temp;
}


The struct Card is defined as

struct Card{
    char rank;
    char suit;
   };
```

8 {Code}

[30 points] In class we talked about the following code whose main program creates a sorted linked list from its input. The input is a positive integer n followed by n lines of input specifying the elements in the list. Note that in the list we do not keep repeated elements. Now modify the code so that after reading the value of n followed by n lines each with an integer value, your program reads in another value which we call y. Then your program will check to see if y is in the list. If it is in the linked list, then it should be deleted, otherwise it should do nothing. After this operation your code should count an print the number of elements in the list. The code below should remain unchanged, so you only need to add code to delete an element from the list and then count the number of elements in the list.

```
#include <stdio.h>
#include <stdlib.h>

  typedef struct node{
      int val;
      struct node* next;
    } Node;

  void addit(Node *header, int w)
   { Node *ptr,*temp;
     ptr=header;
     while(ptr->next != NULL)
      {if (ptr->next->val > w) break;
       if (ptr->next->val == w) return;
       ptr= ptr->next;
      }
     temp= (Node *) calloc(1,sizeof(Node));
     temp->next=ptr->next;
     temp->val=w;
     ptr->next=temp;
     // header=NULL;  Adding this will not change the outcome
   }

  void printit(Node *header)
   { Node *ptr;
     ptr = header;
     printf("New Printing\n");
     while (ptr->next != NULL)
      { printf("%d\n",ptr->next->val);
        ptr= ptr->next;
      }
     printf("\n");
   }
```

```
int main(void)
{ int n,i,w;
  // Creates a sorted list of positive numbers.
  // Repeated elements are eliminated.
  Node *header;
  header = (Node*) calloc(1,sizeof(Node));
  header->next=NULL;
  header->val=-1;
  scanf("%d",&n);
  for (i=0; i<n; i++)
   {scanf("%d",&w);
    addit(header,w);
    printit(header);
   }
}
```