Name: ____________________
U#: ____________________

Exam 1
COP 3514 Program Design

06/03/2013
2:00-3:15 pm

- **Closed books, notes, laptop, phone, neighbors**
- **Good luck!**

1. Write a program that has two integer variables **xBigger** and **equal** defined in the main function. The program should prompt the user to enter two numbers: **x** and **y**. After the user enters the numbers an **Evaluate** function should be called. After the call to the **Evaluate** function the value of **xBigger** should be 1 if **x** is bigger than **y** otherwise it should be 0 and the value of **equal** should be 1 if **x** is equal to **y** otherwise it should be 0. The return type of **Evaluate** must be **void**. You may only use integer and integer pointers as function parameters.

   ***NOTE: No input validation required**

(25 points)

```
void Evaluate(int x, int y, int *p1, int *p2)
{
        *p1=0;
        *p2=0;
        if(x>y)
                *p1=1;
        if(x==y)
                *p2=1;
}

int main()
{
        int x, y, xBigger, equal;
        printf("Enter 2 numbers:\n");
        scanf("%d %d",&x,&y);
        Evaluate(x,y,&xBigger,&equal);

        return 0;
}
```

2. **a)** Define a variable type called **Person** that has a character array parameter of size 20 called **name**, a double parameter called **levelOfAwesomeness,** an integer parameter called **awesome** and an integer parameter called **theMostAwesome**.

(5 points)

```
typedef struct
{
	char name[20];
	double levelOfAwesomeness;
	int awesome;
	int theMostAwesome;
}Person;
```

b) Declare three variables **person1**, **person2** and **person3** of type **Person**:

Barney with **levelOfAwesomeness** 1000
Robin with **levelOfAwesomeness** 500
Ted with **levelOfAwesomeness** 400
Set the **awesome** parameter to 0 for all 3 variables.
Set the **theMostAwesome** parameter to 0 for all 3 variables.

(5 points)

```
Person person1={“Barney”,1000,0,0};
Person person2={“Robin”,500,0,0}
Person person3={“Ted”,400,0,0}
```

c) Complete the function bellow that takes 3 pointers to a **Person** variables as parameters and sets the **awesome** parameter of a **Person** to 1 if and only if the **levelOfAwesomeness** of that **Person** is above 300 and it sets the **theMostAwesome** parameter to 1 only if the **levelOfAwesomeness** parameter of that **Person** variable is higher than the **levelOfAwesomeness** parameters of the other two **Person** variables.

(15 points)

```
void isAwesome(Person *p1, Person *p2, Person *p3)
{

        if(p1->levelOfAwesomeness>300)
                p1->awesome=1;
        if(p2->levelOfAwesomeness>300)
                p2->awesome=1;
        if(p3->levelOfAwesomeness>300)
                p3->awesome=1;

        if((p1->levelOfAwesomeness>p2->levelOfAwesomeness)
                &&(p1->levelOfAwesomeness>p3->levelOfAwesomeness))
                p1->theMostAwesome=1;

        if((p2->levelOfAwesomeness>p1->levelOfAwesomeness)
                &&(p2->levelOfAwesomeness>p3->levelOfAwesomeness))
                p2->theMostAwesome=1;

        if((p3->levelOfAwesomeness>p2->levelOfAwesomeness)
                &&(p3->levelOfAwesomeness>p1->levelOfAwesomeness))
                p3->theMostAwesome=1;
```

3. Complete the **say_hello()** function such that the output of the program is:

(25 points)

```
Hello
Hello again
I already said hello
REALY?!
```

```
#include <stdio.h>

void say_hello();

int main()
{
	say_hello();
	say_hello();
	say_hello();
	say_hello();
	return 0;
}
```

void say_hello()

```
void say_hello()
{
	static int x=0;
	if(x==0)
		printf(“Hello\n”);
	if(x==1)
		printf(“Hello again\n”);
	if(x==2)
		printf(“I already said hello\n”);
	if(x==3)
		printf(“REALLY?!\n”);
	++x;
}
```

4. Complete the definition of the **shiftLeft()** function. The parameters are **startAddr** which is a pointer to the first element of an array and **endAddr** which is a pointer to the last element of an array. The function should shift all of the element values by one position to the left and set the last element value to 0. **You may only use pointers inside of the function**.

   An example of a function call to **shiftLeft()** is presented in the main function below and the output of it is:

   (25 points)

```
2 3 4 5 6 7 8 9 10 0
```

```
int main()
{
        int array[10]={1,2,3,4,5,6,7,8,9,10};
        int i;
        shiftLeft(&array[0],&array[9]);

        for(i=0;i<10;i++)
                printf("%d ",array[i]);
        printf("\n");
        return 0;
}


//You may use only pointers
void shiftLeft(int *startAddr,int *endAddr)
{

        while(startAddr<endAddr)
        {
                *startAddr=startAddr[1];
                startAddr++;
        }
        *startAddr=0;
}
```

Bonus question:
You will receive partial credit on the bonus question only if it is more than 50% accurate.

(10 points)

You are given:

```
typedef struct Dream
{
        int number;
        struct Dream* dream;
}Dream;

Dream first_dream;
```

Assume that the **number** parameter in the **first_dream** is 10, the **dream** parameter in the **first_dream** is the second **dream** and the **number** inside it is 9. The **dream** inside of the second **dream** is the third **dream** and the number is 8. Write a ***recursive*** function **Inception()** that prints the value of the **number** parameter in each **dream** starting with **first_dream**. The function ends when 8 is printed out. Assume that the function is called as **Inception(&first_dream)**

```
//Must use recursion
void Inception(Dream *d)
{

        if(d->number==8)
                printf("%d\n",d->number);
        else
        {
                printf("%d\n",d->number);
                Inception(d->dream);
        }
}
```