Lebanese University
Faculty of Sciences 1

I2204 E – INFO216 E

Imperative Programming

Partial 2018-2019
Wed. Nov. 14th pm
Duration: 1 hour

Exercise I: Mysterious string (6 points)

Draw the memory state tracing the following program. Deduce the output.

```
#include <stdio.h>
char* f2(char *s){
    if (*s=='\0'|| *(s+1)=='\0' || *(s+2)=='\0')
        return s;
    *s+=11;
    return f2(++s);
}
char* f1(int n, char* str){
    str+=n;
    f2(str);
    *(str+1)-=3;
    return str;
}
void test(){
    int n=3;
    char str[]="difficult or easy";
    printf("%s",f1(n++,str+10));
}
void main() {
    test();
}
```

Exercise II: Donating Blood by Compatible Type (10 points)

Blood types are very important when a blood transfusion is necessary. In a blood transfusion, a patient must receive a blood type compatible with his or her own blood type. If the blood types are not compatible, that can cause death. Therefore, it is important that blood types be matched before blood transfusions take place. For simplicity, the following chart shows 3 different blood types along with their matching donor and receiver blood types.

		Receiver		
		A	B	C
Donor	A	x		
	B	x	x	
	C	x	x	x

Modified version of a blood type chart

We want to keep track of available donors along with their blood types in a linked list of donors. Each donor is identified by : his name, his phone number and his blood type.

In case of emergency, a matched donor will be contacted for blood transfusion.

1. Define the types **Donor** and **Node**. The type Node represents a node in a simply linked list of donors.
2. Write the function `find1Donor` which given a receiver's blood type and the donors' list, returns a pointer to the first occurrence of a donor with a matching blood type if it exists.

The lab technician was not satisfied with the proposed data structure. He complained about the fact that it takes too much time to search for a compatible donner. Thus, we restructured the above data in 3 linked lists to separate the available donners' blood types. We used the following declaration to store the 3 linked lists:

```
Node* donors[3];
```

3. Rewrite the function `find1Donor` taking this time, as parameters, the receiver's blood type and the donors array.

Exercise III: Loop it (6 points)

Write a **recursive** function `loopIt` that modifies a given simply linked list into a circular one. In a circular linked list, the last node points to the first node of the list.

Good luck!