Programming-2 1st Midterm 2024

Question 1: Find the output of the following code (15 points):

```
#include <stdio.h>

void F(char * x, char * y){
      char temp = * x;
      * x = * y;
      * y = temp;
   }

void G(char * str, int m, int r){
   int i;
   if(m == r) printf("%s\n",str);
   else for(i= m; i<=r; i++){
      F((str+m),(str+i));
      G(str,m+1,r);
      F((str+m),(str+i));
   }
}

int main() {
   // Write C code here
 char str[]="ABC";
 int n =3;
 G(str,0,n-1);

   return 0;
}
```

Solution:

ABC
ACB
BAC
BCA
CBA
CAB

Question 2: According to the given info below:

Employee Information: firstName char (30), lastName char (30), employeeId (int), and startDate char (11).
Department Information: depName char (50), depId (int), managerName char (50).
Salary History: year (int), salary (double).

Write the following codes:

a) Write structs for Employee, Department and salaryHistory. (5 points)
b) For the number of n entered by the keyboard, make a dynamic memory allocation for Employee struct in main function. (5 points)
c) Write a function that adds an element to dynamic array employee as i'th array element. The prototype of function is "void addEmployee(Employee *employee, int k, char * firstName, char * lastName, int employeeId, char * startDate, Department dept)". (5 points)
d) ~~Write a function with prototype "void addSalary (Employee *employee, int employeeId, int year, double salary)" that adds year and salary entry belonging to the employeeId member to the dynamic array named employee. (10 points)~~ ***question canceled***

Solutions:

a:

```
typedef struct {
        char firstName[30];
        char lastName[30];
        int employeeId;
        char startDate[11];
} Employee;

typedef struct {
        char deptName[50];
        int deptId;
        char managerName[50];
} Department;

typedef struct {
        int year;
        double salary;
} SalaryHistory;
```

b:

```
 int main(){
        Employee employee;
        int n;
        printf("Enter n value");
        scanf("%d",&n);
        employee=(Employee*)malloc(sizeof(Employee)*n);

return 0;
}
```

c:

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

void addEmployee(Employee *employee, int k, char *firstName, char *lastName,
int employeeId, char *startDate, Department dept){
        employee[k].employeeId = employeeId;
        strcpy(employee[k].firstName, firstName);
        strcpy(employee[k].lastName, lastName);
                strcpy(employee[k].startDate, startDate);
}
```

Question 3: Re-write the following function without using recursion (25 points);

```
void F(char * str){
        char c;
        if(*str == '\0') return;
        c= * str;
        if(c >= 'a' && c<= 'z') *str = *str+1;
        else if(c == 'z') *str = 'a';
        F(str+1);
        return;
}
```

Solution:

```
void F1(char * str){
        char c;
        while (*str !='\0'){
                c = *str;
                if(c >= 'a' && c<= 'z') *str = *str+1;
                else if(c == 'z') *str = 'a';
                str++;
        }
        return;
}
```

Question 4: Write a function with prototype int *F(int matris[N][M]) that return the address of minimum element of matrix of size NxM. Assume that the N and M values derived as global. ()

Solution:

```
#define N 3
#define M 2

int *F(matrix[][]){
	int i,j,*p,min;

	min = matrix[0][0];
	p = &matrix[0][0];

	for(i=0;i<N;i++){
		for(j=0;j<M;j++){
			if(matrix[i][j] < min){
				min = matrix[i][j];
				p = &matrix[i][j];
			}
		}
	}
	return p;
}
```