Examen SINF1252 août 2019

1) Ecrivez le corps de la fonction ndiff.

```
/* @pre string!=NULL, chaîne terminée par '\0'
 * @post retourne une chaîne de caractères contenant
 *       une seule fois chaque caractère présent dans string.
 *       le pointeur retourné doit être alloué par malloc
 *       et la taille de la zone mémoire retournée doit correspondre
 *       au nombre de caractères différents dans le string
 *       retourne NULL en cas d'erreur
 *       Exemple
 *       ndiff("AABABBC") -> "ABC" (zone mémoire de 4 bytes)
 *       ndiff("ZZZZ12AAZZ21") -> "Z12A" (zone mémoire de 5 bytes)
 */

char * ndiff(char *string) {

if(string == NULL)
    return NULL;

char *diff = (char*)malloc(1);
if(diff == NULL)
    return NULL;

int i = 0, k = 0;
while(string[i] != 0)
{
    int found = 0;
    int j = 0;
    while(j < i)
    {
        if(string[j] == string[i])
            found = 1;

        if(found == 1)
            break;

        j++;
    }
    if(found == 0)
```

```
    {
        diff[k] = string[i];
        diff = (char*)realloc(diff, k+1);
        if(diff == NULL)
            return NULL;

        k++;
    }
    i++;
}

diff[k] = 0;
return diff;
```

2) Ecrivez ici le corps de la fonction myfunc

```
#include <stdio.h>
#include <stdlib.h>
#include <errno.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>
#include <string.h>
/*
 * @pre file_name != NULL, nom du fichier source
 *      new_file_name != NULL, nom du fichier destination
 *
 * @post copie le contenu de {file_name} dans {new_file_name} en préservant les permissions.
 *       return le nombre de bytes copiées si la fonction se termine correctement, -1 en cas d'erreur
 */
int myfunc(char *file_name, char *new_file_name) {

if(file_name == NULL || new_file_name == NULL)
    return -1;

int fd1 = open(file_name, O_RDONLY);
if(fd1 == -1 )
    return -1;
```

```
struct stat *buf = (struct stat*)malloc(sizeof(struct stat));
if(buf == NULL)
{
   close(fd1);
   return -1;
}
if(stat(file_name, buf) == -1)
{
   free(buf);
   close(fd1);
   return -1;
}
int fd2 = open(new_file_name, O_WRONLY|O_CREAT, buf->st_mode);
free(buf);
if(fd2 == -1)
{
   close(fd1);
   return -1;
}
int count = 0;
char *cpy;
int err = read(fd1, &cpy, sizeof(char));
while(err == sizeof(char))
{
   count++;
   err = write(fd2, &cpy, sizeof(char));
   err = read(fd1, &cpy, sizeof(char));
}
if(err == -1)
{
   close(fd1);
   close(fd2);
   return -1;
}
if(close(fd1) == -1)
{
   close(fd2);
   return -1;
}
if(close(fd2) == -1)
   return -1;

return count;
```

3) La fonction suivante a été écrite en assembleur. Traduisez la en une version équivalente en C. Votre fonction doit nécessairement avoir comme nom mp.

mp:

```
    subl $8, %esp
    movl 16(%esp), %edx
    movl 12(%esp), %ecx
    movl %ecx,%eax
    addl %ecx,%ecx
    addl %eax,%ecx
    cmpl %edx,%ecx
    jle m1
    movl %edx, %eax
    addl $8, %esp
    ret
  m1:
    movl %ecx, %eax
    addl $8, %esp
    ret
```

```
int mp(int c, int d)
{
   int a = c;
   c += c;
   c += a;
   if(c <= d)
   {
      a = c;
      return a;
   }
   a = d;
   return a;
}
```