REPORT

- **Exercise2**

In this exercise, it is requested to write a C program that creates a new file in which are listed all the files present in the directory generate in the previous exercise using the following instructions:

```
sprintf(command, "ls ./%s > list.txt", dir);
system(command);
```

Then, the program must read that file and for each line read, using the system call *fork()* creates a new child, which sorts the content of the file by executing the system call *execlp()*.

```
while( fgets(line, sizeof(line), fp) != NULL){
      n = strlen(line);
      line[n-1] = '\0';

      sprintf(path, "./%s/%s", dir, line);

      child = fork();
      if(child < 0){
            fprintf(stderr, "Errore fork\n");
            break;
      }
      else if(child == 0){ // I'm the child
            printf("I'm sorting %s\n", path);
            execlp("sort" ,"mysort", "-n", "-o", path, path, (char *) NULL);

      }else{ // I'm the father
            count++;

             if(count >= c){
                 wait(&status);
                 count--;
              }

      }
}
while(wait(&status) > 0);
```

In the main loop, basically, the father reads one line at a time and creates a child which is in charge to sort the file, then since the number of child cannot be greater than the value ***C***, that is passed through the command line, there is an *if statement* in which the father will perform a *wait()* waiting the end of one of its children before creating more but, since the number of files could not be a multiple of ***C***, it is necessary to add another while loop at end in which is performed a *wait()* in order to wait the end of all the children. Finally,

it is used again the system call *system()* in order to print in one file all the numbers in an ascending order.

```
system("cat ./data/f* > all_sorted.txt");
system("sort -n -o all_sorted.txt all_sorted.txt");
```