Degree: Artificial Intelligence
Subject: Fundamentals of Programming 2
Practical project 1

Anna Sikora
Manuel Montoto
Nischey Verma Kumar
Computer Architecture and Operating Systems
Universitat Autònoma de Barcelona
Cerdanyola, Spain
{anna.sikora, manuel.montoto}@uab.cat

***Abstract*—During this second semester of the 2022-2023 academic year you will have 2 practical projects in the Fundamentals of Programming 2 subject. This document presents the first one of them related to the concepts we've seen up to now. With the double aim of putting into practice the knowledge obtained in this subject and of acquiring independent work habits, the practical project 1 will focus on the development of a program in C which includes different aspects of programming in C, such as program arguments, static and dynamic memory or dynamic data structures.**

You will have 3 sessions in order to deliver a functional version of the program. However, this project will require more hours of dedication, not only the 3 in-present sessions. During the last sessions you must deliver the work done.

In this document we introduce a description of the problem you will have to solve. It will be related to a simulation of robots that help us in our household chores.

***Index Terms*—C programming, dynamic memory, program arguments, linked lists, stacks, queues, simulator based on events, Artificial Intelligence.**

I. Introduction

With the double objective of putting into practice the knowledge obtained in this subject and of acquiring independent work habits, the laboratory practices will focus on the development of a C program that will cover different aspects seen up to now. In this project we propose a set of possible milestones (implementation of different functionalities of the program) and a deadline (first deliverable) in which you will have to deliver the work done.

The planning necessary to achieve these milestones, and even deciding which milestones you want to achieve, is your responsibility.

In Section II we present the description of the proposed program. In Section IV we describe the milestones that the project may have (although they depend on you and your organization) and in Section V we give some recommendations and general orientations. Finally, in Section VI you will find the delivery description and how the project will be assessed.

II. Problem description

Currently, the technology arrives to the point that robots are available to help us at home, as it can be seen on Figure 1. In order to analyze different approaches, check different behaviours and assess their productivity, we have to develop a preliminary version of a program that will simulate robots' tasks.

For this preliminary version of the simulator, we will work with 3 kinds of robots that will help us in: sorting books, organizing plates and going for shopping.

Fig. 1. Example of a robot helping at home.

The simulator will be driven by events. The number of events must be passed to the program as an argument from the command line. We provide you with a template (files `.c` and `.h`) where we included a base for the program, in particular functions to generate a set of events in order to check the correct functionality of your program and a template.

The simulator generates events (one by one) using the function we provide you. Then, it starts consuming them (event by event). Depending on the generated event type (type of household chores), a corresponding robot proceeds with the event execution.

A. Household chores 1: Sorting books

Functionality For household chores 1, we will have one robot. In this case, the robot organizes our books. Imagine

that the robot organizes our library. It takes a given book and searches for the place to insert. The robot will look at the current list of books searching for a place to insert it. We want to have all our books sorted by the author (Figure 2).

The list of books will remain up to the end of the program. The program gives information about the final number of books in the list. Finally, all the books of the list will be released using `free()` only once, at the end of the program.

Fig. 2. Sorting books.

Data structure

Book – each book will contain the following fields: author (a string of characters), title (a string of characters), year (integer number).

You can use static arrays for author and title, for example:

```
char author[20];
```

Be carefull when using arrays of characters in C. Please, see Section V for further information about how to use strings in C.

B. Household chores 2: Managing plates

Functionality

For household chores 2, we will have one robot. The robot will organize our kitchen segregating coming plates (see Figure 3). There are 3 different types of plates. Each type of plate will be segregated by its type and put to the corresponding stack of the same plates. As there are 3 types of plates, there will be also 3 stacks (one stack for each type of plate).

Fig. 3. Managing plates.

The robot controls to which stack a given plate must be put and how many plates are in each stack. At the beginning, all stacks are empty. If a plate comes, it means that the robot must put it on the corresponding stack.

If there are MAX_CAPACITY (a constant value given in a template) of plates in a stack, it means that the robot can put it back to the wardrobe (it will remove all plates - MAX_CAPACITY plates - from the stack and the stack will be empty again).

As the number of plates depends on the number of events generated by our simulator, at the end of the program execution there still might be plates in stacks. In this situation, the program gives information about the final number of plates that were not put in a wardrobe (the total number of plates that are still in all stacks). After that, these plates will be released using `free()`.

Data structure

Plate - each plate has the corresponding type of plate (dinner plate, soup plate, dessert plate). Moreover, each plate has a color (white, green, yellow and beige).

Please, see Section V for further information about enumeration types in C.

C. Household chores 3: Shopping

Functionality

For household chores task 3, we will have a set of robots. In this case, each robot will take a list of purchases (i.e. the robot has a number of things to buy) and will go to the supermarket for shopping (see Figure 4. One robot will go shopping once.

Fig. 4. Shopping.

When an event of shopping appears, a new robot is created for this task and a list of purchases is assigned to it (i.e. the number of things to buy will be assigned to it). The robot goes to the queue (supermarket queue) and is added to the end of the queue. Each robot will wait in the queue to be served if there are more robots before it. When its turn comes (i.e. no other robot is buying), it will be removed from the queue (i.e. released using `free()`) and it will buy all things from the shopping list (i.e. a number of things to buy). After the shopping is finished, it can go back home.

In this case, a simulation of period of time must be developed. We will do it in a very simple way, based on the number of events consumed and the number of things to buy.

The robot is added to the queue. If there are more robots already in the queue, it will be added at the end of the queue and must wait for its turn. The time will be passing in correspondence to the events consumed (events from the main program loop: plates, books, shopping). One unit time passes when one event (from the main loop of events) is consumed (for example, one book was sorted). Two unit times pass when

two events are consumed (for example, one book was sorted, one plate was managed), etc.

To give a clearer example: when a robot with ID=4 is generated to buy 3 things, it is added to the queue. If there are no other robot in the queue, it will be directly removed from this queue (i.e. released using `free()`) and served (it will buy its 3 things during 3 events consumed from the main loop (e.g. 1 sorting books, 1 managing plates, 1 sorting books).

If there are other robots in the queue before robot with ID=4 (e.g. robot ID=2 with 2 things to buy, robot ID=3 with 1 thing to buy), robot ID=4 will wait 3 events to be served.

As the number of robots depends on the number of events generated by our simulator, at the end of the program execution there still might be robots in the shopping queue. In this situation, the program gives information about the final number of robots that are still in the queue (the total number of robots that are still in the shopping queue). After that, these robots will be released using `free()`.

Data structure

Purchase – number of things to buy (integer number) and identifier of robot that goes for shopping (integer number).

III. Program arguments and template

The program must take as input argument `Number of events` that must be generated by the program.

Remember that, instead of choosing the "Run Code" option from Visual Studio Code, you can compile and execute the program from the terminal. Just input the following command to compile:

```
gcc -o executable source_file.c
```

and then execute the executable file with the corresponding arguments, for example:

```
program 100
```

If a user introduces a different number of arguments, the program must show an error message and indicate which is the correct format of the program execution.

We provide you with a template for starting the simulator implementation. Please, review it well, analyze and implement different functionalities step by step. You can find 2 source code files:

- `project1.h` - header file that contains all necessary structures and variables, some of them with certain initial values.
- `template.c` - a template for a source code of the simulator. It includes `project1.h`, and hence you can use all the structures and variables declared there. This file is divided in 5 parts:
  1) general functions for an event generation and argument checking
  2) a set of functions for the book sorting functionalities, in particular the implementation of a function for the random book generation
  3) a set of functions for the plate managing functionalities, in particular the implementation of a function for the random plate generation
  4) a set of functions for the shopping functionalities, in particular the implementation of a function for the random shopping generation.
  5) a `main()` function that will call a simulator loop.

The example outputs and statistics that we can obtain after execution of the program are presented in Appendix.

IV. Milestones

The goal of the project is to provide all the functionality of the simulator. You can set the following milestones:

1) passing required arguments to the program from the command line and managing errors
2) generating events - using the functions we've provided you in the template, you can generate event by event in your code
3) consuming events - a loop that consumes events (event by event); depending on the type of event, it calls a corresponding robot
4) functionality of the robot described in Section II-A Household chores 1 - Sorting books
5) functionality of the robot described in Section II-B Household chores 2 - Managing plates
6) functionality of the robot described in Section II-C Household chores 3 - Shopping

V. Recommendations

Here are some recommendations on how to develop the project:

- Properly organize the project with the classmate of your group.
- Comment your code with helpful comments (mostly for yourself). To provide a useful comment, it is very important that it is accompanied by the date and the author.
- To define enumerations in C, you can use a special enumeration type called `enum`. It is mainly used to assign names to integral constants, the names make a program easy to read and maintain. For example:

```
enum Plates
{
    dinner_plate=0,
    soup_plate=1,
    dessert_plate=3
};
```

- Working with strings in C you can use arrays of characters, for example: `char array[20];` or using dynamic arrays and allocating/releasing memory with `malloc()/free()`, respectively. Moreover, while doing operations on strings in C, you will have to use specific functions, for example:
  - `strlen()` - calculates the length of a string

TABLE I
WEIGHT OF EACH PRACTICAL PROJECT INTO THE SUBJECT'S FINAL GRADE.

Practical project 1	**20%**
Practical project 2	**20%**

  - `strcpy()` - copies a string to another
  - `strcmp()` - compares two strings
  - `strcat()` - concatenates two strings
- You can use version control tools (there are a few with free options):
  - **Bitbucket** https://bitbucket.org
  - **GitHub** https://github.com/
  - **GitLab** https://about.gitlab.com/

  In this way we will avoid panic attacks and problems due to loss of information, catastrophic decisions, etc.

VI. DELIVERY AND ASSESSMENT

The project will be done in pairs and you will have 3 'laboratory' sessions. Take into consideration, that this project will require more hours of dedication, not only the 3 in-present sessions. In the last session you will have to explain orally the solution and implementations carried out. After the last session you will have a short period (3 days) to hand in a report or fixing small errors. This means that by the last session you should already have the whole implementation and report almost finished and only check/fix/add any small details or observations you may have and comment them with the professors.

A. Delivery via Virtual Campus

The delivery will be done through the Virtual Campus until the indicated date. It's enough if only one of the members of the group deliver the project. You have to attach: the report in pdf format and the source code (`*.h, *.c`) compressed in one `.zip` file.

B. Assessment

The final grade of the practical projects represents 40% of the subject's final grade. It is an essential requirement to have a grade of 5 or higher in the practical projects to pass the subject. You will have 2 deliveries (two practical projects) during the semester; this is the first one. The contribution of each project to the final subject's grade is shown in table I.

The following criteria will be used to calculate the grade of this first project:

- [0,5 points] The management of the arguments passed to the program and consideration of possible users' errors.
- [5 points] The 'correct' working of the code. There are 3 functionalities:
  - [1 point] Household chores 1: Sorting books
  - [2 points] Household chores 2: Managing plates
  - [2 points] Household chores 3: Shopping
- [0,5 points] Code style and comments.
- [2 points] Report
  - [1,5 points] Description of the strategy in an accurate and reasoned way.
  - [0,5 points] Criticism of the problems that arose and the solutions found during the development of the project.
- [2 points] Continuous assessment of attendance and participation in class [INDIVIDUAL PART].

VII. CONCLUSIONS

We have already presented you the first of two practical projects you will develop this semester. Please, contact us if you have any doubts or questions: `Manuel.Montoto@uab.cat` and/or `Anna.Sikora@uab.cat`. Do not hesitate to ask for meetings with us!

APPENDIX

Final statistics of the program execution for different configurations

```
program 10
STATISTICS WHEN CLEANING THE SIMULATION:
    Removing books...
        2 books has been removed.
    Cleaning all stacks of plates...
        5 plates has been removed
    Cleaning shopping queue...
        0 robots has been removed

program 100
STATISTICS WHEN CLEANING THE SIMULATION:
        Removing books...
            29 books has been removed.
        Cleaning all stacks of plates...
            2 plates has been removed
        Cleaning shopping queue...
            9 robots has been removed

program 1000
STATISTICS WHEN CLEANING THE SIMULATION:
    Removing books...
        319 books has been removed.
    Cleaning all stacks of plates...
        4 plates has been removed
    Cleaning shopping queue...
        22 robots has been removed
```