Homework No 1.

Patient Management in a Medical Office

We aim to model the management of patients in a medical office. A patient is characterized by their last name, first name, and a field "**app**" (for appointment) of type integer indicating whether the patient has an appointment or not: 0 if the patient is without an appointment, 1 if the patient has an appointment.

Before being consulted by the doctor, patients are entered into a waiting room which will be modeled by a linked list of patients. A secretary admits patients with **app** = 1 in the order of their arrival, then admits other patients (those with **app** = 0) in the order of their arrival as well.

Initially, you are asked to propose the data structures that will be used in the program.

1) Define the **Patient** data structure to describe a patient's information.
2) Define the **node** data structure representing a node of the patient list. It consists of two fields: a **patient** field and a **next** field.
3) Define the **list** type as a pointer to the node structure.

In a second step, you are asked to write the following sub-programs:

1) The **addPatient()** function, which adds a new patient to the end of a linked list. The patient to add and the list are passed as parameters.
2) The **nbWithWithoutAppointment()** procedure, which counts and returns the number of patients with appointments and the number of patients without appointments in a list of patients passed as a parameter.
3) The **enterPatient()** function, which allows a patient to enter for consultation, returns them as a result, and removes them from the list. This operation is performed as follows: if there are no patients with appointments, then the first patient in the list is entered, returned as a result, and removed from the list. Otherwise, the first patient with an appointment is returned and removed from the list.
4) The **consultWaitingRoom()** procedure, which first displays patients with appointments, then patients without appointments.
5) The **main()** function, which allows testing all the previous sub-programs. The program displays a menu indicating the different operations that can be performed and waits for the user's choice. The menu can be in the following form:

```
Welcome to our medical office management program
1) Add a patient
2) Calculate the number of patients with and without appointments
3) Enter a patient for consultation
4) Consult the waiting room
0) Quit the program
What is your choice?
```

Once a choice is made, the desired operation is executed by calling the corresponding sub-program. This process is repeated until the user chooses to exit the program.

For further clarification, please refer to the attached executable file. It describes the expected result of the homework.

واجب منزلي 1.

إدارة المرضى في عيادة طبية

نهدف في هذا الواجب المنزلي إلى نمذجة إدارة المرضى في عيادة طبية. يتميز كل مريض بلقب واسم بالإضافة إلى خانة "**app**" (ترمز للموعد) من النوع الصحيح (**integer**) والتي تشير إلى ما إذا كان المريض لديه موعد أم لا: 0 إذا كان المريض بدون موعد، و 1 إذا كان المريض لديه موعد.

قبل أن يتم معاينتهم من طرف الطبيب، يُدخل المرضى إلى غرفة انتظار ممثّلة بواسطة قائمة مترابطة (Linked list) للمرضى. تقوم الممرّضة بإدخال المرضى الذين لديهم موعد أولا (**app** = 1) حسب ترتيب وصولهم، ثم تقوم بقبول المرضى الآخرين (أولئك الذين لديهم **app** = 0) حسب ترتيب وصولهم أيضًا.

في المرحلة الأولى، يُطلب منك اقتراح هياكل البيانات التي ستُستخدم في البرنامج.

1) قم بتعريف هيكل البيانات **Patient** لوصف معلومات المريض.
2) قم بتعريف هيكل البيانات **node** التي تمثل عنصر من قائمة المرضى. يتكون كل عنصر من حقلين أو خانتين: حقل **patient** وحقل **next**
3) قم بتعريف نوع **list** كمؤشر إلى هيكل **node**

في الخطوة الثانية، يُطلب منك كتابة البرامج الفرعية التالية:

1) دالة **addPatient()** التي تضيف مريضًا جديدًا إلى نهاية قائمة مترابطة. يتم تمرير المريض المراد إضافته والقائمة كمداخل.
2) إجراء **nbWithWithoutAppointment()** الذي يحسب ويُرجع عدد المرضى الذين لديهم مواعيد وعدد المرضى الذين ليس لديهم مواعيد في قائمة المرضى الممررة كمدخل.
3) دالة **enterPatient()** التي تسمح للمريض بالدخول للمعاينة وترجعه كنتيجة وتزيله من القائمة. يتم تنفيذ هذه العملية على النحو التالي: إذا لم يكن هناك مريض لديه موعد، فسيتم إدخال أول مريض في القائمة وإرجاعه كنتيجة وحذفه من القائمة. وإلا، يتم إرجاع أول مريض لديه موعد وحذفه من القائمة.
4) إجراء **consultWaitingRoom()** الذي يعرض أولاً المرضى الذين لديهم مواعيد، ثم المرضى الذين ليس لديهم مواعيد.
5) الدالة الرئيسية **main()** التي تسمح باختبار جميع البرامج الفرعية السابقة. يقوم البرنامج بعرض قائمة تُشير إلى العمليات المختلفة التي يمكن تنفيذها وينتظر اختيار المستخدم. يمكن أن تكون القائمة على النحو التالي:

```
6) Welcome to our medical office management program
7) 1) Add a patient
8) 2) Calculate the number of patients with and without appointments
9) 3) Enter a patient for consultation
10) 4) Consult the waiting room
11) 0) Quit the program
12) What is your choice?
```

بمجرد اتخاذ الاختيار، يتم تنفيذ العملية المطلوبة عن طريق استدعاء البرنامج الفرعي المقابل. يتم تكرار هذه العملية حتى يختار المستخدم إنهاء البرنامج.

لمزيد من التوضيح، يرجى الرجوع إلى الملف التنفيذي المرفق. يصف هذا الملف النتيجة المنتظرة من الواجب.