DIVOC

MEMORIA PROYECTO

PROGRAMACION I

LOLA MIRANDA LÓPEZ
LAURA GONZÁLEZ LEMOS

Indice

Módulo principal

El módulo principal se llama divoc.c y contiene el código necesario para la inicialización de la aplicación, el intérprete de comandos de la interfaz de usuario y todas las funciones que se consideren necesarias para desarrollar el módulo. Contiene las librerías stdio.h, ctype.h, string.h y stdlib.h, además de los ficheros de cabecera “inout.h” y “database.h”

En este módulo solo se encuentra la función principal main a la que le pasan funciones que se encuentran en los otros dos módulos, la función no recibe ningún parámetro y retorna un 0. En primer lugar, se encuentra la declaración del fichero como fich y se abre ese fichero en modo lectura llamado patients.txt. Si el fichero no está vacío se lee hasta que retorna EOF, es decir hasta que no quedan más elementos en el fichero. Se vuelve al principio del fichero y se abre el fichero en una lista a la que llamamos pHead. Se cierra el fichero. Después este se abre otra vez, pero en modo escritura.

Recibe las funciones stripe, headline, y otra vez stripe que formarán la carátula del programa (solo aparece cuando se entra por primera vez en el programa). La función stripe incluye el símbolo (-) que forma la línea y el tamaño de esta y headline, el nombre de la aplicación (DIVOC_) y el símbolo que va al principio y al final de la línea.

Luego el programa mostrará el menú principal: “Register a patient”,” Search for a patient”,” Discharge a patient”,” List patients by age”,” Mark positive” y “Exit the program” y con la función get_character se le pedirá al usuario que seleccione una opción entre RSDLPX. Según la opción seleccionada se llamará a la función correspondiente (que se encuentran en el database). En el caso de las funciones p_register y p_discharge qué son las que pueden hacer modificaciones en la lista de pacientes se pasan esas funciones a pHead (el propio puntero) y p_exit se le pasa al entero fin. Cuando fin sea igual a 0 se cerrará el fichero y se cerrará la aplicación.

Módulo de entrada/salida

El módulo de entrada/salida se llama inout.c y además de todas las funciones que se consideren necesarias para desarrollar el módulo, consiste en una librería de funciones y su correspondiente fichero de cabecera (inout.h). Incluye las librerias stdio.h, ctype.h, stdlib.h y string.h, además también incluye los ficheros de cabecera “inout.c” y “database.h”.

CreaPaciente()

Prototipo: pPatient CreaPaciente (char *name, char *DNI, int date, int fever, int cough, char symptom)

Esta función se utiliza para crear los datos de un paciente. Recibe como parámetros los datos del paciente (nombre, DNI, fecha, fiebre, tos, síntoma).

Se declara el puntero pAux (y se reserva memoria para él) y los datos que se acaban de introducir se pasan a pAux. Retorna pAux con los datos guardados del nuevo paciente.

CreaLista ()

Prototipo: pPatient CreaLista (FILE *fich)

Esta función se utiliza para crear la lista que contiene los pacientes que registras. Recibe como parámetro un fichero que contiene los datos de uno o varios pacientes. Se declaran las variables correspondientes a cada dato de un paciente. Se meten todos los datos desde el fichero hasta que se hayan acabado. Se han declarado tres punteros, uno que recibe los datos de la función CreaPaciente, otro que apunta al inicio de la lista y otro al final. Si la lista solo contiene un elemento los tres punteros coinciden. Sino se va apuntado al puntero siguiente. Al final retorna un puntero que incluye todos los pacientes.

ImprimePaciente ()

Prototipo: pPatient ImprimePaciente (FILE *fich, pPatient pHead)

Esta función muestra por pantalla los datos de un paciente. Recibe como parámetros un fichero que contiene los datos de uno o varios pacientes y del puntero pHead. Pasa los datos de la lista al fichero.

InsertaPaciente ()

Prototipo: pPatient InsertaPaciente (char *name, char *DNI,int date, int fever, int cough, char symptom, pPatient pHead)

Esta función sirve para introducir al final de la lista de pacientes el paciente que se acaba de crear. Recibe los parámetros correspondientes a los datos del paciente (nombre, DNI, fecha, fiebre, tos y síntoma) además del puntero que incluye la lista de pacientes. Se llama a la función CreaPaciente para conseguir los datos del paciente. Si la lista está vacía el paciente que se acaba de crear pasa a ser el primer elemento de la lista, si no se recorre la lista y el paciente creado pasa a ser el último elemento de la lista. Al final retorna una nueva lista la cual incluye el nuevo paciente.

EiminaPaciente ()

Prototipo:pPatient EliminaPaciente(pPatient pHead, char *DNI)

Esta función elimina a un paciente de la lista de pacientes. Recibe como parámetros la tabla de pacientes (pHead) y un DNI. La función se va moviendo por la lista buscando los datos del paciente cuyo DNI coincide con el introducido por el usuario. Cuando lo encuentra libera memoria y retorna la tabla de pacientes sin ese paciente.

BuscaPaciente ()

Prototipo: pPatient BuscaPaciente (pPatient pHead, char *DNI)

Esta función recorre la lista de pacientes y busca a un paciente determinado. Recibe como parámetros la tabla de pacientes y un DNI (que ha sido introducido previamente por el usuario). Cuando la lista no está vacía lo que hace es recorrerla comparando los DNI con el que recibió de parámetro para encontrar al paciente. No retorna nada.

headline()

Prototipo: void headline(char *mensaje, char simbolo)

Esta función muestra por pantalla el título de la aplicación. Recibe como parámetros el título de la aplicación y el símbolo (|) que aparece en los laterales de esa línea. Primero imprime el símbolo, se dejan los correspondientes espacios en blanco, escribe el nombre (DIVOC_), vuelve a dejar espacios en blanco y termina la línea con otro símbolo.

stripe ()

Prototipo: void stripe(char simbolo, int size)

Dibuja una línea de guiones en la carátula de la aplicación. Recibe como parámetros el símbolo (-) y el tamaño de la línea, que en este caso es 50. La función entra en un bucle que imprime guiones hasta que se hayan escrito todos los símbolos necesarios.

get_string()

Prototipo: void get_string(char *opcion, char *datoleido, int MIN, int MAX)

Esta función se utiliza para leer una cadena que se ha introducido desde el teclado. La función recibe como parámetros la frase de invitación que se debe mostrar, la dirección de memoria donde se debe dejar la cadena leída y la longitud mínima y máxima admitida para la cadena. Primero la función imprime la frase de invitación y los límites de longitud. Lee la cadena y reserva espacio en la memoria. Se comprueba su longitud. Si la longitud no es correcta se repite este proceso. Al final la cadena se copia en la la dirección de memoria donde se debe dejar la cadena leída.

get_integer ()

Prototipo: int get_integer(char *opcion,int MIN,int MAX)

Esta función lee un entero que se introduce desde el teclado. Recibe como parámetros la frase de invitación, el valor mínimo admisible para el entero leído y el máximo. Primero mostrará una línea con la frase de invitación y el valor mínimo y máximo. Después leerá lo que ha introducido el usuario y solo tendrá en cuenta el entero introducido al principio, y reserva memoria. Si este entero no está entre los valores mínimo y máximo se vuelve a imprimir la línea de invitación y así hasta que el entero sea correcto. Retorna el entero.

get_character()

Prototipo: char get_character(char *opcion, char *comprobacion)

Esta función lee un carácter que se introduce por el teclado. Recibe como parámetros la frase de invitación que se debe mostrar y la palabra de comprobación , que es una cadena de caracteres. Se imprime en una línea la frase de invitación y la palabra de comprobación. Después leerá lo que ha introducido el usuario y solo tendrá en cuenta el primer carácter. Reserva memoria. Este se pasa a mayúscula y se comprueba si la letra que se ha introducido está dentro de la palabra de comprobación. Si no está se vuelve a imprimir la línea de invitación hasta que la letra sea correcta. Al final la función retorna el carácter.

yes_or_no()

Prototipo: int yes_or_no(char *pregunta)

Esta función confirma que el usuario ha respondido a una pregunta con sí o no. Recibe como parámetro la pregunta. Primero imprime la pregunta, después la lee y reserva memoria. Sólo lee el primer carácter que introduce el usuario. Pasa esta letra a mayúscula y si esta letra es distinta a Y o N, vuelve a imprimir la pregunta. Si se responde Y la función retorna un 1 y si no retorna un 0.

display_patient()

Prototipo: void display_patient(pPatient pHead)

Esta función se utiliza para mostrar la información sobre un paciente. Recibe como parámetro un puntero que contiene la información del paciente del cual queremos imprimir la información. Reserva memoria. Se imprimen los datos a partir del puntero que apunta a cada dato del paciente.

verify_DNI()

Prototipo: int verify_dni (char *DNI)

Esta función se utiliza cuando se debe verificar si un DNI es válido. Recibe como parámetro la cadena de 9 caracteres que contiene el DNI que se quiere verificar. Al inicio se copia la cadena DNI en otra cadena. Se comprueba que los ocho primeros caracteres de la cadena recibida son números y se convierten en un entero. Se calcula el resto de dividir dicho entero por 23. A este resto le llamamos índice y comprobamos si la letra que ocupa el puesto del número índice en la cadena "TRWAGMYFPDXBNJZSQVHLCKE" coincide con la letra del DNI. Si es así la función retorna un 1, en caso contrario retorna un 0.

Módulo gestor de la base de datos

El módulo gestor se llama database.c y además de las funciones se encuentran las librerias stdio.h,stdlib.h,ctype.h y string.h, también los ficheros de cabecera "database.h" y "inout.h". Además, se encuentra declarada la struct aPatient y los typedef Patient y *pPatient.

p_register()

Prototipo: pPatient p_register(pPatient pHead)
Esta función registra un nuevo paciente en la tabla de pacientes. Recibe como parámetro el puntero (pHead) que contiene la lista de pacientes.
La función informa de la operación que se va a realizar con la línea “Register”, invita al usuario a introducir una cadena (debe estar entre 1 y 24 caracteres) con el nombre del paciente, luego un DNI válido (debe tener como longitud mínima admisible y como longitud máxima admisible el valor 9) el cual debe ser comprobado por la función verify_DNI, si es inválido mostrará “Invalid DNI”. Luego invita al usuario a introducir la fecha de nacimiento del paciente (debe estar entre 1900 y 2020). Después le pedirá que indique si tiene fiebre y tos mediante la función yes_or_no y si tiene algún síntoma mediante get_character. Llama a la función CreaPaciente y se la pasa a pAux y InsertaPaciente a pHead. La función mostrará “New patient” y con la función display_patient, que recibe pAux, mostrará en una sola línea la información del nuevo paciente registrado. Retorna pHead, que es el puntero que contiene la nueva lista de pacientes que incluye el nuevo paciente registrado.

p_search()

Prototipo: int p_search(pPatient pHead)
Esta función busca a un paciente con un DNI dado, recibe pHead, que contiene la lista de pacientes y retorna un 0, es decir vuelve a enseñar el menú principal.
La función informa de la operación que se va a realizar con la línea “Search”, invita al usuario a introducir un DNI y le pasa BuscaPaciente a pAux. Si no hay ningún paciente en la tabla, es decir pHead está vacío, mostrará “No patients yet” y volverá al menú. Si el DNI dado no coincide con ningún paciente, es decir pAux está vacío, mostrará “Unknown patient” y volverá al menú. Si el DNI coincide con el dado mostrará “Patients data” y con la función display_patient mostrará en una sola línea la información de ese paciente.

p_discharge()

Prototipo: pPatient p_discharge (pPatient pHead)
Esta función da de baja a un paciente de la tabla de pacientes, recibe pHead, que contiene la lista de los pacientes y retorna la lista sin el paciente que se ha dado de baja.
La función informa de la operación que se va a realizar con la línea “Discharge”, invita al usuario a introducir un DNI y le pasa la función EliminaPaciente a pAux, si no hay ningún paciente en la tabla, es decir pHead está vacío, mostrará “No patients yet” y volvera al

menú. Si el DNI dado no coincide con ningún paciente, es decir si pAux está vacío,mostrará “Unknown patient” y volverá al menú. Si el DNI coincide con el dado mostrará “Discharged patient”, le pasa pAux a pHead y dará de baja al paciente de la tabla

p_list()

Prototipo: int p_list (pPatient pHead)
Esta función muestra los pacientes nacidos antes de un año dado, recibe pHead, que contiene la lista con los pacientes y retorna un 0, es decir al acabar vuelve al menú principal.
La función informa de la operación que se va a realizar con la línea “List”, si no hay ningún paciente en la tabla, es decir si pHead está vacío, mostrará “No patients yet” y volverá al menú. Si no invita al usuario a introducir un año válido con la función get_integer, mostrará “Patients born before ...” y mostrará con la función display_patient la información, en una sola línea, de los pacientes nacidos antes o en ese año. El puntero pHead avanza por la lista.

p_mark()

Prototipo: int p_mark (pPatient pHead)
Esta función identifica y muestra los pacientes que son positivos, es decir, que tienen tos, fiebre y algún síntoma, recibe pHead, que contiene la lista de pacientes, y retorna un 0, es decir, cuando acaba vuelve al menú principal.
Esta función informa de la operación que se va a realizar con la línea “Positives”, si no hay ningún paciente en la tabla, es decir, pHead está vacio, mostrará “No patients yet” y volverá al menú. Si no, muestra “Positive patients” y buscará pacientes que sean positivos. De los pacientes identificados como positivos mostrará con la función display_patient, en una sola línea cada paciente, su información. El puntero pHead avanza por la lista.

p_exit()

Prototipo: int p_exit(FILE *fich, pPatient pHead)
Esta función sale del programa, recibe el puntero pHead, que contiene la lista con los pacientes, y el fichero fich y retorna un 0 si no se sale del programa y un 1 si se sale.
Esta función informa de la operación que se va a realizar con la linea “Exit”, después con la función yes_or_no() pregunta “Are you sure you want to exit?”, si la respuesta es afirmativa guarda en el fichero la información de los pacientes que estaba en la tabla. Si la respuesta es negativa vuelve al menu .