SMART FRIDGE

Documentazione

Studenti: Murro Giuseppe,
Tartarelli Alessio

Caso di studio per il corso di
“Laboratorio di informatica”

ITPS 2017/18

INDICE

Analisi

1.1. Descrizione del Sistema

Il sistema “Smart Fridge” trova la sua applicazione in ambiente domestico e si affianca a quelle che sono le abitudini alimentari familiari, consigliando piatti in base ai cibi presenti in frigorifero così che si abbia sempre fantasia per la realizzazione di ricette più o meno complesse di giorno in giorno.

Il programma, appena viene avviato, automaticamente acquisisce la data corrente, dopo la quale in modo automatico ogni 24 ore si passerà al giorno successivo così da differenziare l’elaborazione dei dati nei vari giorni della settimana. Il programma rimarrà in esecuzione sempre, a meno che non si decida di spegnere il frigorifero.
Gli elementi principali su cui “Smart Fridge” lavora sono gli utenti, i menù settimanali di ogni utente, gli alimenti e le ricette.
Gli utenti, ovvero i componenti della famiglia sono distinti attraverso un nome utente (che deve essere necessariamente diverso per ogni membro). Ognuno di loro, dopo aver creato un proprio profilo o aver eseguito l’accesso ad uno già esistente, può scegliere una tra le funzionalità disponibili attraverso un menù iniziale.
Quando un membro della famiglia va a fare la spesa, il sistema consente di aggiungere i nuovi alimenti in “Smart Fridge” o di modificare quelli presenti (per esempio nel caso in cui ci sia stato un errore nella digitazione di alcuni campi al momento dell’inserimento).
Inoltre “Smart Fridge” monitora gli alimenti presenti in frigorifero, così che non si abbia mai la mancanza di qualcosa, notifica in caso di prossima scadenza degli alimenti e può anche suggerire una o più ricette contenenti questi ultimi che altrimenti potrebbero marcire o finire per essere gettati.
Difatti, quando la quantità di un alimento esaurisce o quando la data di scadenza supera la data corrente, l’alimento viene tolto dal sistema “Smart Fridge”.

Un’altra funzionalità offerta è quella di gestire una raccolta di ricette, perciò l’utente selezionando l’opportuna opzione dal menù potrà: aggiungere una nuova ricetta al ricettario, modificare o cancellarne una già presente.
Il sistema memorizza anche un menù settimanale per ogni membro della famiglia in base ai piatti che ha consumato durante i vari pasti giornalieri, così che l’utente possa tener traccia di ciò che ha mangiato nei giorni precedenti. Ogni utente quando seleziona una ricetta che vuole preparare dal ricettario, verrà considerata come piatto consumato nel giorno corrente e verrà aggiunto nella giusta sezione del menù settimanale. Per di più “Smart Fridge” consente anche di modificare o cancellare le informazioni relative ad un pasto consumato in un dato giorno.
“Smart Fridge” è anche in grado di rilevare quali alimenti sono presenti in frigorifero in quantità inferiore rispetto a un livello minimo di soglia, così da poter generare automaticamente una lista

della spesa, che potrà essere stampata dall'utente ogni qual volta lo richieda attraverso la specifica funzionalità presente nel menù.

Ciascun utente inoltre può ricercare una determinata ricetta da dover preparare. Ci sono diversi metodi per effettuare la ricerca: inserendo un numero finito di alimenti in input, il sistema mostra all'utente se esiste una o più ricette che contengono tali ingredienti oppure mostra un messaggio di errore nel caso in cui nessuna ricetta corrisponda all'input dato.
Nella scelta del piatto l'utente può tener conto anche delle calorie sprigionate dallo stesso. "Smart Fridge" per agevolare tale scelta può ordinare le ricette in base alle kcal prodotte e mostrarle all'utente.

Accanto alle funzionalità di base previste dal sistema, sono state implementate anche le seguenti funzionalità aggiuntive:

- Possibilità di visualizzare le ricette che possono essere preparate con gli alimenti in scadenza, se presenti;
- Possibilità di ricercare una ricetta inserendo un solo alimento in input e visualizzare le ricette contenenti questo alimento, se presenti;
- Possibilità di ricercare una ricetta inserendo l'ID della ricetta da cercare in input e visualizzare la ricetta corrispondente, se presente;
- Possibilità di assegnare un menù giornaliero di un utente ad un altro utente;
- Possibilità visualizzare tutte le ricette al di sotto di una soglia di kcal data.

1.2. Requisiti Funzionali

Codice	Nome	Descrizione
R01	Registrazione nuovo utente	Il programma deve chiedere all'utente di inserire un nome utente e verificare che altri utenti non abbiano utilizzato lo stesso nome per registrarsi.
R02	Login utente	Il programma deve verificare se il nome utente inserito dall'utente corrisponde a quello di uno tra gli account registrati.
R03	Menu iniziale	Il programma deve mostrare all'utente un menu iniziale dove vengono visualizzate tutte le funzionalità disponibili e deve acquisire in input una scelta.
R04	Caricamento da file dei dati	Il programma deve caricare da file già esistenti il vettore degli alimenti che sono presenti in frigo, il vettore delle ricette preparabili e il vettore dei menu settimanale di ciascun utente.
R05	Cancellazione di un utente	Il programma deve poter cancellare l'utente attualmente loggato e deve eliminare il menù settimanale che si riferisce a tale utente.
R06	Cambio utente	Il programma deve fare il logout per l'utente attualmente loggato, e deve permettere di scegliere un altro utente con il quale fare l'accesso.
R07	Visualizzazione alimenti	Il programma deve mostrare tutti gli alimento presenti nel frigorifero.
R08	Aggiunta nuovo alimento	Il programma deve far inserire dall'utente una serie di campi in input corrispondenti alle informazioni di un alimento e salvarle su file.
R09	Modifica alimento	Il programma deve chiedere all'utente quale alimento modificare e quali campi, ricevere le modifiche come input e modificare il file.

R10	Controlla scadenza alimento	Il programma deve confrontare la data di scadenza dell'alimento e se è scaduto deve eliminarlo dal file.
R11	Controlla esaurimento alimento	Il programma deve controllare le occorrenze di ciascun alimento presente in frigo e stampare un avviso nel caso in cui le occorrenze siano minori di una certa soglia.
R12	Visualizzazione ricette	Il programma deve mostrare tutti le ricette scritte sul file.
R13	Aggiunta nuova ricetta	Il programma deve far inserire dall'utente una serie di campi in input corrispondenti alle informazioni di una ricetta e salvarle su file.
R14	Modifica ricetta	Il programma deve chiedere all'utente il nome della ricetta da modificare e quali campi, ricevere le modifiche come input e modificare il file.
R15	Cancella ricetta	Il programma deve chiedere all'utente quale ricetta cancellare e modificare il file.
R16	Visualizzazione menù settimanale	Il programma deve mostrare il menu settimanale scritto sul file inerente all'utente loggato.
R17	Aggiunta nuovo piatto nel menu settimanale	Il programma deve chiedere all'utente quale ricetta vuole preparare e per quante persone e salvarla sul file nella sezione corrispondente al giorno corrente. Se la ricetta inserita non esiste o se la ricetta non è preparabile, il programma darà un messaggio di errore.
R18	Modifica account a cui il menu giornaliero è riferito	Il programma deve chiedere all'utente quale menù giornaliero vuole modificare e a quale account lo vuole assegnare, ricevere le modifiche come input e modificare il file.
R19	Modifica piatto nel menu settimanale	Il programma deve chiedere all'utente quale menù giornaliero vuole modificare e quale pasto, dopodiché deve fargli scegliere un nuovo piatto da sostituire al precedente. Se il piatto è preparabile, il programma deve modificare il file.

R20	Cancella piatto dal menu settimanale	Il programma deve chiedere all'utente quale pasto vuole cancellare dal menu settimanale, dopodiché deve modificare il file.
R21	Visualizzazione della lista della spesa	Il programma deve mostrare i nomi degli alimenti che sono in esaurimento o che sono terminati nel frigorifero.
R22	Cancellazione della lista della spesa	Il programma deve cancellare tutti gli alimenti scritti sul file della lista della spesa.
R23	Suggerimento ricetta	Il programma deve mostrare le ricette che contengono gli ingredienti presenti nel frigorifero che scadono 2 giorni dopo (o meno) la data corrente
R24	Ricerca ricetta per ID	Il programma chiedere all'utente l'ID della ricetta da cercare e se esiste deve mostrala.
R25	Ricerca ricetta per ingrediente	Il programma deve chiedere all'utente il nome di un ingrediente e deve mostrare tutte le ricette che contengono quell'ingrediente.
R26	Ricerca per insieme di ingredienti	Il programma deve chiedere all'utente i nomi di un insieme di ingredienti e deve mostrare tutte le ricette che contengono quegli ingredienti.
R27	Ricerca per kcal	Il programma deve chiedere all'utente di inserire una soglia di kcal e deve mostrare tutte le ricette che hanno le kcal inferiori alla soglia inserita.
R28	Ordinamento per kcal	Il programma deve ordinare le ricette in modo crescente in base alle kcal e visualizzarle.

1.3. Strumenti di Sviluppo

Per lo sviluppo del caso di studio sono stati utilizzati due notebook:

- HP - Pavilion 15-ak002nl (specifiche tecniche);
- MSI - GL62M 7REX (specifiche tecniche).

Entrambi i notebook sono dotati di sistema operativo linux, in particolare Debian 9 utilizzato durante lo sviluppo del sistema. Sono in un'ultima fa se è stato utilizzato Windows per la fase di test con CUnit e per testare in generale il corretto funzionamento del software anche su sistema operativo Windows, inoltre è stato utilizzato anche per generare la versione release del software.

L'IDE adottato è stato Eclipse CDT con plugin "Eclox" per poter utilizzare il tool Doxygen per la generazione della documentazione a partire da determinati commenti e tag inseriti direttamente nel codice sorgente. Un altro plugin adoperato è "EGit" per utilizzare il software git e connetterci alla repository creata su Bitbucket, così da sincronizzare più velocemente e comodamente le varie versioni del caso di studio.

Essendo di due paesi differenti, per la comunicazione abbiamo utilizzato TeamSpeak, software che offre la possibilità di connettersi ad un server per poi entrare in una "stanza" cosi da instaurare una chat vocale con tutti gli utenti presenti nella stessa "stanza".

I requisiti necessari per eseguire il nostro software non sono molto elevati in termini di prestazioni, tuttavia è necessario eseguire il software su macchine con sistema operativo Windows ed è necessaria una connessione ad Internet per usufruire a pieno delle funzionalità di "Smart Fridge", in particolare per godere della funzionalità di apertura di una pagina web contenente il procedimento per realizzare la ricetta scelta.

Progettazione

2.1 Progettazione dei tipi di dato

Tabella per i tipi di dato definiti:

Nome	Tipologia	Descrizione	Tipi/Campi/Valori
Livello	Enum	Tipo di dato per memorizzare la complessità della ricetta	**BASSA** **MEDIA** **ALTA**
Data	Struct	Tipo di dato per memorizzare le data	**Giorno:** unsigned short int **Mese:** unsigned short int **Anno:** unsigned short int
Utente	Struct	Tipo di dato per rappresentare e memorizzare gli utenti	**Id_utente:** unsigned short int **Nome:** char[LUNGHEZZA_NOME]
Ingrediente	Struct	Tipo di dato per rappresentare e memorizzare gli ingredienti delle ricette	**Nome_alimento:** char[LUNGHEZZA_NOME] **quantita:** char[LUNGHEZZA_QUANTITA] **unita_misura:** char
Alimento	Struct	Tipo di dato per rappresentare e memorizzare gli alimenti presenti in frigorifero	**Id_alimento:** unsigned short int **nome_alimento:** char[LUNGHEZZA_NOME] **data_scadenza:** data **numero_pezzi:** short int **peso_unitario:** float

Ricetta	Struct	Tipo di dato per rappresentare e memorizzare le ricette presenti nel ricettario	**Id_ricetta:** unsigned short int **nome_ricetta:** char[LUNGHEZZA_NOME] **ingredienti:** ingrediente[NUMERO_INGREDIENTI] **numero_ingredienti:** unsigned short int **tempo_preparazione:** unsigned short int **link_procedimento:** char[MAX_LUNGHEZZA_LINK] **complessita:** enum livello **kcal:** float **n_persone:** unsigned short int
Pasto	Struct	Tipo di dato per rappresentare e memorizzare i pasti degli utenti	**Id_ricetta:** unsigned short int **n_persone:** unsigned short int
Menu_giornaliero	Struct	Tipo di dato per rappresentare e memorizzare il menù giornaliero degli utenti	**id_menu:** unsigned short int **id_utente:** unsigned short int **data_menu:** data **pasti:** pasto[MAX_NUMERO_PASTI] **numero_pasti:** unsigned short int **kcal_giornaliere:** float

Tabella costanti simboliche:

Nome	Tipologia	Descrizione	Tipi/Campi/Valori
BISESTILE	Costante simbolica	Costante utilizzata per indicare ogni quanto febbraio è bisestile	4
DIVIDI_ARRAY	Costante simbolica	Costante utilizzata per indicare il valore per cui il numero di elementi di un vettore deve essere diviso per ottenere il centro	2
GIORNI_FEBBRAIO	Costante simbolica	Costante utilizzata per indicare il numero di giorni di febbraio nel caso in cui è bisestile	29
LINK_BASE	Costante simbolica	Costante utilizzata per indicare la lunghezza del link di base (https://bit.ly/)	15
LUNGHEZZA_GIORNO_SETTIMANA	Costante simbolica	Costante utilizzata per indicare la lunghezza massima della stringa che indica il giorno delle settimana	10
LUNGHEZZA_INPUT	Costante simbolica	Costante utilizzata per indicare la massima lunghezza della stringa acquisita in input	12

LUNGHEZZA_NOME	Costante simbolica	Costante utilizzata per indicare la lunghezza massima che possono avere i nomi (es nome ricetta) in questo programma	41
LUNGHEZZA_NOME_FILE	Costante simbolica	Costante utilizzata per indicare la lunghezza massima che possono avere i nomi dei file (es elenco_ricette.bin) in questo programma	25
LUNGHEZZA_QUANTITA	Costante simbolica	Costante utilizzata per indicare la lunghezza massima della stringa che indica la quantità di un alimento	5
LUNGHEZZA_RISPOSTA	Costante simbolica	Costante utilizzata per indicare la massima lunghezza che può avere la risposta alla domande poste all'utente	5
MAX_ALIMENTI_ESAURIMENTO	Costante simbolica	Costante utilizzata per indicare il numero massimo di alimenti in esaurimento	30
MAX_ALIMENTI_RICERCA	Costante simbolica	Costante utilizzata per indicare il numero massimo di alimenti inseriti per poter effettuare una ricerca per insieme di alimenti	7

MAX_ANNO	Costante simbolica	Costante utilizzata per indicare l'anno massimo in cui si considera una data valida	2030
MAX_CALORIE	Costante simbolica	Costante utilizzata per indicare il numero massimo di kcal di un piatto	9000
MAX_COMPLESSITA	Costante simbolica	Costante utilizzata per indicare il valore massimo che può assumere la enum livello	2
MAX_GIORNO	Costante simbolica	Costante utilizzata per indicare il numero massimo del giorno in un mese	31
MAX_LUNGHEZZA_DATA	Costante simbolica	Costante utilizzata per indicare la lunghezza massima che può assumere una stringa contenente una data	10
MAX_LUNGHEZZA_LINK	Costante simbolica	Costante utilizzata per indicare la lunghezza massima che può assumere una stringa contenente un link al procedimento della ricetta	27
MAX_LUNGHEZZA_USERNAME	Costante simbolica	Costante utilizzata per indicare la lunghezza massima della stringa contenente il nome utente	15

MAX_MESE	Costante simbolica	Costante utilizzata per indicare il numero massimo del mese in un anno	12
MAX_NUMERO_PASTI	Costante simbolica	Costante utilizzata per indicare il numero massimo di pasti giornalieri per singolo utente	10
MAX_PERSONE	Costante simbolica	Costante utilizzata per indicare il numero massimo di persone per cui si può preparare una ricetta o per quante persone è quella data ricette in fase di aggiunta di una nuova ricetta	10
MAX_QUANTITA	Costante simbolica	Costante utilizzata per indicare la quantità massima di un dato ingrediente in fase di aggiunta di una nuova ricetta	1500
MAX_SCELTA_RICERCA	Costante simbolica	Costante utilizzata per indicare il numero di scelta massimo tra le possibili voci del menù di ricerca ricetta	4
MAX_TEMPO_PREPARAZIONE	Costante simbolica	Costante utilizzata per indicare la quantità di tempo massima (espressa in minuti) per la preparazione di una ricetta	360

MIN_ALIMENTI_RICERCA	Costante simbolica	Costante utilizzata per indicare il numero minimo di alimenti inseriti per poter effettuare una ricerca per insieme di alimenti	2
MIN_ANNO	Costante simbolica	Costante utilizzata per indicare l'anno minimo in cui si considera una data valida	1990
MIN_GIORNO	Costante simbolica	Costante utilizzata per indicare il numero minimo del giorno in un mese	1
MIN_LUNGHEZZA_DATA	Costante simbolica	Costante utilizzata per indicare la lunghezza minima che può assumere una stringa contenente una data	8
MIN_LUNGHEZZA_LINK	Costante simbolica	Costante utilizzata per indicare la lunghezza minima che può assumere una stringa contenente un link al procedimento della ricetta	16
MIN_LUNGHEZZA_USERNAME	Costante simbolica	Costante utilizzata per indicare la lunghezza minima della stringa contenente il nome utente	5
MIN_MESE	Costante simbolica	Costante utilizzata per indicare il numero minimo del mese in un anno	1

NON_STAMPA_ALIMENTI_RIMESSI	Costante simbolica	Costante utilizzata per indicare il valore booleano “false” per quanto riguarda la stampa degli alimenti rimessi in frigorifero	0
NON_STAMPA_ALIMENTI_USATI	Costante simbolica	Costante utilizzata per indicare il valore booleano “false” per quanto riguarda la stampa degli alimenti usati per la preparazione di una ricetta	0
NUMERO_ALIMENTI	Costante simbolica	Costante utilizzata per indicare il numero massimo di alimenti presenti in frigorifero	200
NUMERO_ELEMENTI_ESAURIMENTO	Costante simbolica	Costante utilizzata per indicare il numero massimo di alimenti in esaurimento	200
NUMERO_GIORNI_SETTIMANA	Costante simbolica	Costante utilizzata per indicare il numero di giorni in una settimana	7
NUMERO_INGREDIENTI	Costante simbolica	Costante utilizzata per indicare il numero massimo di ingredienti presenti in una ricetta	30
NUMERO_MENU_GIORNALIERI	Costante simbolica	Costante utilizzata per indicare il numero massimo di menu giornalieri degli utenti	140

NUMERO_RICETTE	Costante simbolica	Costante utilizzata per indicare il numero massimo di ricette che il ricettario può contenere	50
NUMERO_UTENTI	Costante simbolica	Costante utilizzata per indicare il numero massimo di utenti registrati al sistema	20
NUMERO_VOCI_MENU	Costante simbolica	Costante utilizzata per indicare il numero delle voci del menù principale del sistema	21
SOGLIA_ESAURIMENTO	Costante simbolica	Costante utilizzata per indicare il numero per cui se la quantità di un alimento è <= di questa soglia si considera l'alimento in esaurimento	2
SOGLIA_SCADENZA	Costante simbolica	Costante utilizzata per indicare il numero di giorni prima della data di scadenza, per cui se la data di scadenza è tra due giorni l'alimento verrà notificato tra gli alimenti in scadenza	2
STAMPA_ALIMENTI_RIMESSI	Costante simbolica	Costante utilizzata per indicare il valore booleano "true" per quanto riguarda la stampa degli alimenti rimessi in frigorifero	1

STAMPA_ALIMENTI_USATI	Costante simbolica	Costante utilizzata per indicare il valore booleano “false” per quanto riguarda la stampa degli alimenti usati per la preparazione di una ricetta	1
NUMERO_ELEMENTI_SCADENZA	Costante simbolica	Costante utilizzata per indicare il numero massimo di alimenti in scadenza	200

Tabella dei file:

Nome	Tipologia	Descrizione	Tipi/Campi/Valori
elenco_alimenti.bin	File binario	File utilizzato per memorizzare gli alimenti presenti in frigorifero	Struct **alimento**
elenco_ricette.bin	File binario	File utilizzato per memorizzare le ricette presenti nel ricettario	Struct **ricetta**
elenco_utenti.bin	File binario	File utilizzato per memorizzare gli utenti registrati al sistema	Struct **utente**
lista_spesa.txt	File testuale	File utilizzato per memorizzare gli alimenti che vengono aggiunti alla lista della spesa	**char[**LUNGHEZZA_NOME**]**
menu_settimanale.bin	File binario	File utilizzato per memorizzare il menù settimanale dei vari utenti	Struct **menu_giornaliero**

2.2 Progettazione delle librerie / funzioni

Abbiamo suddiviso il programma nelle seguenti librerie:

1. **carica_file.h**: Questa libreria contiene tutte le funzioni che consentono di leggere i file binari esistenti riguardanti gli alimenti, le ricette e il menu settimanale e ricavarne dei vettori. In ogni elemento del vettore verrà scritto ciascun record del file letto. Normalmente queste funzioni vengono utilizzate all'inizio di un programma per effettuare il caricamento da file. Se i file letti risultano essere vuoti, verranno 'popolati' con le funzioni del tipo popolazione file che inizializzano dei vettori da codice e li scrivono sul file aperto;

2. **controlli.h**: Questa libreria contiene tutte le funzioni che consentono di acquisire in modo corretto interi, numeri decimali, stringhe con e senza spazio e date e funzioni che effettuano controlli su una stringa in input per verificare se è effettivamente un intero, un numero decimale, o una stringa valida o una data corretta. Queste funzioni vengono usate principalmente per acquisire dati da tastiera in modo corretto e svolgono lo stesso ruolo della scanf o della fgets implementando controlli sull'input;

3. **date.h**: Questa libreria contiene tutte le funzioni necessarie per gestire le date in un programma, come ad esempio l'acquisizione della data corrente dal sistema, la modifica di una data ecc. Per la maggior parte tali funzioni lavorano sui campi della struct data: giorno, mese, anno e giorno settimanale;

4. **elaborazione_alimenti.h**: Questa libreria contiene tutte le funzioni necessarie per gestire gli alimenti nel frigo, come ad esempio l'aggiunta di un nuovo alimento, la modifica, la cancellazione, la stampa di tutti gli alimenti, l'ordinamento, ecc. La maggior parte delle funzioni lavorano su vettori di tipo alimento, che vengono passati per riferimento e quindi potranno essere modificati dalle funzioni, a meno che non sia indicato il qualificatore const. I campi della struct alimento su cui le funzioni lavorano sono: id, nome, data scadenza, numero pezzi e peso unitario;

5. **elaborazione_ricette.h**: Questa libreria contiene tutte le funzioni che consentono di interagire con le ricette, dall' aggiunta di una nuova ricetta alla sua eliminazione o modifica. Inoltre ci sono varie funzioni per l'acquisizione di dati sempre inerenti alle ricette, funzioni per la loro scrittura su file, funzioni di ricerca e di ordinamento delle ricette;

6. **gestione_menu_settimanale.h**: Questa libreria contiene tutte le funzioni necessarie per gestire il menu settimanale di ciascun utente, come ad esempio l'aggiunta di un nuovo menu, la modifica, la cancellazione, la stampa di tutti i menu giornalieri, ecc. La maggior parte delle funzioni lavorano su vettori di tipo menu_giornaliero, che vengono passati per riferimento e quindi potranno essere modificati dalle funzioni, a meno che non sia indicato il qualificatore const. I campi della struct menu_giornaliero su cui le funzioni lavorano sono: id del menu e dell'utente, il giorno della settimana, i pasti, il numero di pasti e le kcal totali;

7. **gestione_utenti.h**: Questa libreria contiene tutte le funzioni necessarie per gestire il gli account degli utenti, come ad esempio l'aggiunta di un nuovo utente, la cancellazione, la

scelta dell'utente con cui loggarsi ecc. La maggior parte delle funzioni lavorano direttamente sul file in cui gli utenti vengono registrati. I campi della struct utente su cui le funzioni lavorano sono: id del utente e nome utente;

8. **lista_spesa.h**: Questa libreria contiene tutte le funzioni per la gestione della lista delle spesa del programma. Come l'aggiunta di un alimento alla lista della spesa o la cancellazione di tutta la lista della spesa;

9. **strutture_dati.h**: Questa libreria contiene tutti i tipi di dati da noi definiti, come ad esempio il tipo "ricetta". Oltre ai tipi di dato sono presenti anche tutte le costanti simboliche utilizzate nel programma e ci sono anche delle inclusioni a delle librerie più comuni e utilizzate.

Ognuna di queste librerie ha i seguenti metodi:

1. carica_file.h:
   - carica_elenco_alimenti();
   - carica_elenco_ricette();
   - carica_menu_settimanale();
   - popolazione_file_alimenti();
   - popolazione_file_ricette().

2. controlli.h:
   - inserisci_input_intero();
   - inserisci_input_caratteri();
   - inserisci_input_caratteri_spazio();
   - inserisci_input_data();
   - inserisci_input_float();
   - inserisci_input_float_qb();
   - inserisci_input_intero();
   - inserisci_input_link();
   - is_compreso_tra();
   - is_compreso_tra_float();
   - is_data_corretta();
   - is_float();
   - is_intero();
   - is_link();
   - is_stringa_caratteri();
   - is_stringa_caratteri_spazio();
   - split_data();
   - svuota_buffer_input().

3. date.h:
   - calcola_data_corrente();
   - confronta_date();
   - get_giorno_settimana_char();

- get_giorno_settimana_intero();
- giorno_precedente();
- giorno_successivo();
- stampa_data_corrente().

4. elaborazione_alimenti.h:
   - aggiungi_alimento();
   - apri_frigo();
   - cancella_alimento();
   - controlla_esaurimento_alimenti();
   - controlla_scadenza_alimenti();
   - get_nuovo_id_alimento();
   - get_ultimo_indice_alimento();
   - indice_id_alimento();
   - input_aggiungi_alimento();
   - input_modifica_alimento();
   - is_alimenti_scaduti();
   - is_alimento_esistente();
   - is_scaduto();
   - modifica_alimento();
   - shell_sort_date();
   - stampa_alimenti_esaurimento();
   - stampa_alimenti_scadenza().

5. elaborazione_ricette.h:
   - aggiungi_ricetta();
   - apri_ricettario();
   - cancella_ricetta();
   - cerca_ricetta_insieme();
   - copia_ricettario();
   - get_ultimo_id_ricetta();
   - get_ultimo_indice_ricetta();
   - indice_id_ricetta();
   - input_aggiungi_ricetta();
   - input_alimento();
   - input_cancella_ricetta();
   - input_ingredienti();
   - input_inserisci_id();
   - input_insieme_alimenti();
   - input_modifica_ingredienti();
   - input_modifica_ricetta();
   - input_numero_alimenti();
   - input_ricerca_ricetta();
   - is_ricetta_esistente();

- modifica_ricetta();
- partition();
- quickSort();
- ricerca_ricetta_alimento();
- ricerca_ricetta_kcal();
- swap_ricette();
- visualizza_ricetta().

6. gestione_menu_settimanale.h:
   - aggiungi_menu_giornaliero();
   - cancella_menu_giornaliero();
   - copia_vettore_alimenti();
   - elimina_ingredienti_usati();
   - get_nuovo_id_menu();
   - get_ultimo_indice_menu();
   - indice_id_menu();
   - input_cancella_menu();
   - input_modifica_menu();
   - input_prepara_ricetta();
   - input_scelta_pasto();
   - is_menu_esistente();
   - is_ricetta_preparabile();
   - modifica_giorno();
   - modifica_pasto();
   - rimetti_alimenti_in_frigo();
   - visualizza_menu_settimanale().

7. gestione_utenti.h:
   - aggiungi_utente();
   - cancella_menu_settimanale_utente();
   - cancella_utente();
   - input_cancella_utente();
   - is_utente_esistente();
   - mostra_utenti();
   - ricerca_utente();
   - seleziona_utente();
   - stampa_utente().

8. lista_spesa.h:
   - aggiungi_a_lista();
   - cancella_lista_spesa();
   - stampa_lista_spesa();

Per maggiori informazioni andare al punto 3.1 della documentazione.

2.3 Dipendenza tra funzioni

Di seguito sono riportate tutte le dipendenze tra le varie funzioni del programma:

1. carica_file.h:
   - carica_elenco_alimenti():
     - popolazione_file_alimenti();
     - carica_elenco_alimenti();
   - carica_elenco_ricette():
     - popolazione_file_ricette();
     - carica_elenco_ricette();
2. controlli.h:
   - inserisci_input_intero():
     - svuota_buffer_input();
   - inserisci_input_caratteri_spazio():
     - is_stringa_caratteri_spazio();
     - svuota_buffer_input();
   - inserisci_input_caratteri();
     - svuota_buffer_input();
     - is_stringa_caratteri();
   - inserisci_input_data():
     - svuota_buffer_input();
     - slit_data();
     - is_data_corretta();
   - inserisci_input_float():
     - svuota_buffer_input();
     - is_float();
   - inserisci_input_float_qb():
     - svuota_buffer_input();
     - is_float();
   - inserisci_input_intero():
     - svuota_buffer_input();
     - is_intero();
   - inserisci_input_link():
     - svuota_buffer_input();
     - is_link();

- is_data_corretta():
  - is_compreso_tra();

3. date.h:
   - giorno_precedente():
     - is_compreso_tra();
     - get_giorno_settimana_intero();
     - get_giorno_settimana_char();

   - giorno_successivo():
     - is_compreso_tra();
     - get_giorno_settimana_intero();
     - get_giorno_settimana_char();

4. elaborazione_alimenti.h:
   - controlla_esaurimento_alimenti():
     - aggiungi_a_lista();

   - controlla_scadenza_alimenti():
     - giorno_successivo();
     - confronta_date();

   - input_aggiungi_alimento():
     - get_nuovo_id_alimento();
     - inserisci_input_caratteri_spazio;
     - inserisci_input_data();
     - inserisci_input_intero();
     - inserisci_input_float();
     - aggiungi_alimento();

   - input_modifica_alimento():
     - inserisci_input_intero();
     - is_alimento_esistente();
     - indice_id_alimento();
     - inserisci_input_caratteri_spazio();
     - inserisci_input_data();
     - inserisci_input_float();
     - modifica_alimento();

   - is_alimenti_scaduti():
     - is_scaduto();
     - cancella_alimento();

   - shell_sort_date():

- confronta_date();

5. elaborazione_ricette.h:
   - input_aggiungi_ricetta():
     - get_ultimo_id_ricetta();
     - inserisci_input_caratteri_spazi();
     - inserisci_input_intero();
     - is_compreso_tra();
     - inserisci_input_link();
     - is_compreso_tra();
     - inserisci_input_float();
     - aggiungi_ricetta();
   - cerca_ricetta_insieme():
     - visualizza_ricetta();
   - input_cancella_ricetta():
     - inserisci_input_intero();
     - is_ricetta_esistente();
     - cancella_ricetta();
   - input_ingredienti():
     - inserisci_input_caratteri_spazio();
     - inserisci_input_float_qb();
     - is_compreso_tra_float();
   - input_inserisci_id():
     - inserisci_input_intero();
   - input_insieme_alimenti():
     - inserisci_input_caratteri_spazio();
   - input_modifica_ingredienti():
     - inserisci_input_intero();
     - inserisci_input_caratteri_spazio();
     - inserisci_input_float_qb();
     - is_compreso_tra_float();
   - input_modifica_ricetta():
     - inserisci_input_intero();
     - is_ricetta_esistente();
     - indice_id_ricetta();
     - inserisci_input_link();
     - is_compreso_tra_float();
     - inserisci_input_float();
     - input_modifica_ingredienti();

- modifica_ricetta();

- input_numero_alimenti():
  - inserisci_input_intero();
  - is_compreso_tra();

- input_ricerca_ricetta():
  - inserisci_input_intero();
  - is_compreso_tra();

- partition():
  - swamp_ricette();

- quickSort():
  - partition();
  - quickSort();
- ricerca_ricetta_alimento():
  - visualizza_ricetta();

- ricerca_ricetta_kcal():
  - quickSort();
  - visualizza_ricetta();

6. gestione_menu_settimanale.h:
   - elimina_ingredienti_usati():
     - cancella_alimento();
     - modifica_alimento();

   - input_cancella_menu():
     - inserisci_input_intero();
     - is_menu_esistente();
     - indice_id_menu();
     - is_compreso_tra();
     - cancella_menu_giornaliero();
     - aggiungi_menu_giornaliero();

   - input_modifica_menu():
     - inserisci_input_intero();
     - is_menu_esistente();
     - indice_id_menu();
     - ricerca_utente();
     - modifica_giorno();
     - is_menu_modificato();
     - is_compreso_tra();
     - copia_vettore_alimenti();
     - rimetti_alimenti_in_frigo();

- is_alimenti_scaduti;
- input_scelta_pasto();
- indice_id_ricetta();
- modifica_pasto();
- elimina_ingredienti_usati();

- input_prepara_ricetta():
  - get_ultimo_indice_menu();
  - get_nuovo_id_menu();
  - confronta_date();
  - input_scelta_pasto();
  - indice_id_ricetta();
  - is_ricetta_preparabile();
  - inserisci_input_intero();
  - is_compreso_tra();
  - elimina_ingredienti_usati();
  - aggiungi_menu_giornaliero();

- input_scelta_pasto():
  - inserisci_input_intero();
  - is_ricetta_esistente();
  - is_compreso_tra();

- modifica_giorno():
  - cancella_menu_giornaliero();
  - aggiungi_menu_giornaliero();

- modifica_pasto():
  - is_ricetta_preparabile();
  - elimina_ingredienti_usati();
  - aggiungi_menu_giornaliero();

- rimetti_alimenti_in_frigo():
  - is_alimento_modificato();
  - get_nuovo_id_alimento();
  - modifica_alimento();

- visualizza_menu_settimanale():
  - indice_id_ricetta();

7. gestione_utenti.h:
   - cancella_utente():
     - ricerca_utente();
     - svuota_buffer_input();
     - cancella_utente();
     - cancella_menu_settimananale_utente();

- seleziona_utente();

- mostra_utenti():
  - stampa_utente();

- ricerca_utente():
  - is_compreso_tra();

- seleziona_utente():
  - mostra_utenti();
  - inserisci_input_intero();
  - svuota_buffeer_input();
  - is_stringa_caratteri();
  - is_compreso_tra();
  - is_utente_esistente();
  - is_nome_utente_corretto();
  - aggiungi_utente();
  - ricerca_utente();

8. main.c:
   - carica_elenco_alimenti();
   - carica_elenco_ricette();
   - copia_ricettario();
   - carica_menu_settimanale();
   - calcola_data_corrente();
   - stampa_data_corrente();
   - inserisci_input_intero();
   - shell_sort_date();
   - controlla_esaurimento_alimenti();
   - controlla_scadenza_alimenti();
   - giorno_successivo();
   - seleziona_utente();
   - input_cancella_utente();
   - apri_frigo();
   - stampa_alimenti_esaurimento();
   - stampa_alimenti_scadenza();
   - input_aggiunti_alimento();
   - input_modifica_alimento();
   - apri_ricettario();
   - input_cancella_ricetta();
   - visualizza_menu_settimanale();
   - input_prepara_ricetta();
   - input_modifica_menu();
   - input_cancella_menu();
   - stampa_lista_spesa();
   - ricerca_ricetta_alimento();

- input_inserisci_id();
- input_numero_alimenti();
- input_insime_alimenti();
- cerca_ricetta_insieme();
- inserisci_input_float();
- ricerca_ricetta_kcal();
- quickSort();
- cancella_lista_spesa();
- is_alimenti_scaduti();

2.4 Scelte di progetto

Durante la progettazione del caso di studio, è stato necessario puntualizzare alcune scelte progettuali, al fine di modellare la realtà e prenderla effettivamente sviluppabile:

- Tutti gli alimenti che vengono "acquistati" dagli utenti e che sono necessari per preparare le ricette, verranno conservati in frigo, anche se come ad esempio l'olio, non ne hanno bisogno.
- Tutti gli utenti possono accedere agli stessi alimenti del frigo e alle stesse ricette; l'unico criterio di separazione tra un utente e l'altro riguarda i menù settimanali, che sono personali.
- Per poter testare il programma più facilmente il giorno della settimana non cambia in modo automatico ogni 24 ore ma l'utente dal menù può passare al giorno successivo.
- Quando il numero_pezzi di ciascun alimento presente in frigo raggiunge la soglia minima di 2, l'alimento si considera in esaurimento e viene inserito nella lista della spesa.
- Quando il numero_pezzi di ciascun alimento diventa 0, l'alimento viene cancellato dal vettore alimenti_frigo e dal file.
- Il file del menù settimanale inizialmente sarà vuoto, e verrà scritto man mano che l'utente prepara i vari piatti.
- Il giorno della settimana e la data corrente all'avvio del programma vengono acquisisti in modo automatico dal sistema (attraverso le funzioni della libreria time.h).
- Man mano che l'utente consuma i pasti e passa da un giorno all'altro, viene riempito il menu settimanale. Per esempio, se si parte da "Martedì", andando avanti coi giorni e arrivando al successivo "Martedì", i dati precedenti riguardanti il menu giornaliero vengono sostituiti ma si continua a tenere traccia ad esempio del menu del "Mercoledì" precedente. Dunque la "memoria" del menù settimanale di ogni utente sarà al massimo di 7 giorni.
- Gli alimenti possono essere consumati solo attraverso le ricette (es. non posso mangiare una carota cruda).
- Gli alimenti una volta tolti dal frigo devono essere consumati completamente. Per esempio, se nel frigo si hanno due confezioni di zucchero da 100 g ciascuna, e si prepara lo spumone di caffè (la cui ricetta richiede 150 g di zucchero), verranno consumate entrambe le confezioni, anche se in realtà ne avanzano 50g.
- Quando deve essere preparata una ricetta, l'utente può decidere per quante persone prepararla e i base a ciò, verranno stampate delle quantità diverse di dosi.
- Il procedimento, essendo un link, può essere visualizzato sul browser nel momento in cui una ricetta viene preparata, ma non si deve tener conto delle dosi degli alimenti prescritte dal sito web, ma bensì delle dosi date da "Smart Fridge".
- Tutti gli alimenti vengono misurati in grammi (anche se sono liquidi).
- La quantità di ingredienti in una ricetta può essere espressa in grammi oppure indicata con "qb" (quanto basta).

- Se viene preparata una ricetta che ha un ingrediente indicato con “qb”, per tale ingrediente non verrà tolto alcun alimento dal frigorifero.
- Quando viene modificato l’ID utente di un menù settimanale, se l’utente a cui deve essere assegnato il menù, ne ha uno nello stesso giorno settimanale (es. “Mercoledì”), ma più recente, la modifica non verrà apportata e verrà stampato un messaggio di errore.
- Se due utenti hanno un menù giornaliero nello stesso giorno (es “Venerdì 8/6/2018), la modifica dell’id utente di uno, per assegnarlo all’altro, comporta l’unione dei pasti dei due menù giornalieri.
- Quando viene modificato un pasto dal menù giornaliero, e il nuovo pasto da preparare non è preparabile poiché gli ingredienti non sono disponibili a sufficienza nel frigo, verrà stampa un messaggio di errore e la modifica non verrà apportata.
- Quando un alimento, una ricetta o un menù giornaliero vengono cancellati, l’ID viene posto a 0.
- Quando viene inserito un alimento già scaduto, verrà immediatamente tolto dal frigo

Codifica

3.1. Documentazione doxigen

Clicca qui per visualizzare la documentazione generata da doxigen.

Testing

4.1. Definizione del Piano di Test

Codice Requisito	Codice Test	Nome	Descrizione Test	Eventuali Input	Risultato Atteso	Risultato Ottenuto
R01	1.1	Registrazione nuovo utente	File degli utenti esistente	File: “utenti.bin” Nome: “Antonio” Ultimo ID: 1	Id nuovo utente: 2	Id nuovo utente: 2
R01	1.2	Registrazione nuovo utente	File degli utenti non esistente	File: “abcd.bin” Nome: “Antonio” Ultimo ID: 1	Errore: 0	Errore: 0
R01	1.3	Registrazione nuovo utente	Nome utente nullo	File: “utenti.bin” Nome: “ ” Ultimo ID: 1	Id nuovo utente: 2	Id nuovo utente: 2
R01	1.4	Registrazione nuovo utente	Nome utente esistente	File: “utenti.bin” Nome: “Antonio” Ultimo ID: 2	“Errore, nome utente già esistente!”	“Errore, nome utente già esistente!”
R02	2.1	Login utente	File degli utenti esistente	File: “utenti.bin” ID Utente: 2	“Benvenuto Antonio!”	“Benvenuto Antonio!”
R02	2.2	Login utente	File degli utenti non esistente	File: “abcd.bin” ID Utente: 1	“Errore nell’apertura del file.”	“Errore nell’apertura del file.”
R02	2.3	Login utente	File pieno e ID utente massimo	File: “utenti.bin” ID Utente: 20	“Benvenuto Filippo!”	“Benvenuto Filippo!”
R02	2.4	Login utente	File vuoto e ID utente minimo	File: “utenti.bin” ID Utente: 1	“Utente non esistente!”	“Utente non esistente!”
R03	3.1	Menu iniziale	Input massimo	Scelta: 21	“Fine programma.”	“Fine programma.”
R03	3.2	Menu iniziale	Input minimo	Scelta: 1	Passa al giorno successivo.	Passa al giorno successivo.
R03	3.3	Menu iniziale	Input errato	Scelta: 26	Errore, reinserire scelta.	Errore, reinserire scelta.

R03	3.4	Menu iniziale	Input nullo	Scelta: 0	Errore, reinserire scelta.	Errore, reinserire scelta.
R04	4.1	Caricamento da file dei dati	Nome file esistente	File: “ricette.bin” Numero ricette: 50	Restituisce un vettore di ricette caricato.	Restituisce un vettore di ricette caricato.
R04	4.2	Caricamento da file dei dati	Nome file non esistente	File: “abcd.bin” Numero ricette: 50	“Errore, file non esistente.”	“Errore, file non esistente.”
R05	5.1	Cancellazione di un utente	File degli utenti esistente	File: “utenti.bin” ID Utente: 2	“Utente 2 cancellato!”	“Utente 2 cancellato!”
R05	5.2	Cancellazione di un utente	File degli utenti non esistente	File: “abcd.bin” ID Utente: 1	“Errore, file non esistente.”	“Errore, file non esistente.”
R05	5.3	Cancellazione di un utente	File pieno e ID utente massimo	File: “utenti.bin” ID Utente: 20	“Utente 20 cancellato!”	“Utente 20 cancellato!”
R05	5.4	Cancellazione di un utente	File pieno e ID utente minimo	File: “utenti.bin” ID Utente: 1	“Utente 1 cancellato!”	“Utente 1 cancellato!”
R05	5.5	Cancellazione di un utente	ID utente errato	File: “utenti.bin” ID Utente: 50	“Utente non esistente, cancellazione annullata!”	“Utente 50 cancellato!”
R06	6.1	Cambio utente	File degli utenti esistente	File: “utenti.bin” ID Utente: 2	“Benvenuto Antonio!”	“Benvenuto Antonio!”
R06	6.2	Cambio utente	File degli utenti non esistente	File: “abcd.bin” ID Utente: 1	“Errore nell’apertura del file.”	“Errore nell’apertura del file.”
R06	6.3	Cambio utente	File pieno e ID utente massimo	File: “utenti.bin” ID Utente: 20	“Benvenuto Filippo!”	“Benvenuto Filippo!”
R06	6.4	Cambio utente	File vuoto e ID utente minimo	File: “utenti.bin” ID Utente: 1	“Utente non esistente!”	“Utente non esistente!”
R06	6.5	Cambio utente	ID utente nullo	File: “utenti.bin” ID Utente: 0	Registrazione di un nuovo utente.	Registrazione di un nuovo utente.

R07	7.1	Visualizzazione alimenti	Vettore di alimenti pieno	Vettore: alimenti_frigo [ ] n_alimenti: 100	Stampa tutti i campi degli alimenti	Stampa tutti i campi degli alimenti
R07	7.2	Visualizzazione alimenti	Vettore di alimenti vuoto	Vettore: alimenti_frigo [ ] n_alimenti: 100	Non stampa nulla.	Non stampa nulla.
R07	7.3	Visualizzazione alimenti	n_alimenti diverso dalla lunghezza effettiva del vettore	Vettore: alimenti_frigo [ ] n_alimenti: 80	Stampa i primi 80 alimenti.	Stampa i primi 80 alimenti.
R07	7.4	Visualizzazione alimenti	n_alimenti nullo	Vettore: alimenti_frigo [ ] n_alimenti: 0	Non stampa nulla.	Non stampa nulla.
R08	8.1	Aggiunta nuovo alimento	File degli alimenti esistente	File: “alimenti.bin” Alimento: nuovo_alimento	“Alimento aggiunto.”	“Alimento aggiunto.”
R08	8.2	Aggiunta nuovo alimento	File degli alimenti non esistente	File: “abcd.bin” Alimento: nuovo_alimento	“Errore, file non esistente.”	“Errore, file non esistente.”
R08	8.3	Aggiunta nuovo alimento	Alimento nullo	File: “alimenti.bin” Alimento: alimento_nullo (nome = “”, quantità = 0, ecc...)	“Alimento aggiunto.”	“Alimento aggiunto.”
R09	9.1	Modifica alimento	File degli alimenti esistente	File: “alimenti.bin” Alimento: alimento_modificato	“Alimento modificato.”	“Alimento modificato .”
R09	9.2	Modifica alimento	File degli alimenti non esistente	File: “abcd.bin” Alimento: alimento_modificato	“Errore, file non esistente.”	“Errore, file non esistente.”
R09	9.3	Modifica alimento	Alimento modificato con campi nulli	File: “alimenti.bin” Alimento: alimento_nullo (nome = “”)	“Alimento modificato .”	“Alimento modificato .”
R10	10.1	Controllo scadenza alimento	Data scadenza < data corrente	Alimento: alimento_scaduto (scadenza: 1/6/2018) Data corrente: 18/6/2018	L’alimento viene tolto dal vettore e dal file degli alimenti.	L’alimento viene tolto dal vettore e dal file degli alimenti.

R10	10.2	Controllo scadenza alimento	Data scadenza > data corrente	Alimento: alimento_nonscaduto (scadenza: 1/7/2018) Data corrente: 18/6/2018	Non succede nulla.	Non succede nulla.
R10	10.3	Controllo scadenza alimento	Data scadenza = data corrente	Alimento: alimento_scaduto (scadenza: 9/6/2018) Data corrente: 9/6/2018	L'alimento viene tolto dal vettore e dal file degli alimenti.	L'alimento viene tolto dal vettore e dal file degli alimenti.
R10	10.4	Controllo scadenza alimento	Data scadenza nulla	Alimento: alimento_nullo (scadenza: 0/0/0) Data corrente: 9/6/2018	L'alimento viene tolto dal vettore e dal file degli alimenti.	L'alimento viene tolto dal vettore e dal file degli alimenti.
R11	11.1	Controllo esaurimento alimento	Vettore con alimenti diversi e soglia esaurimento massima	Vettore: alimenti_frigo [ ] Soglia: 65535	Tutti gli elementi del vettore sono in esaurimento.	Tutti gli elementi del vettore sono in esaurimento.
R11	11.2	Controllo esaurimento alimento	Vettore con alimenti diversi e soglia esaurimento minima.	Vettore: alimenti_frigo [ ] Soglia: 0	Nessun elemento del vettore è in esaurimento.	Nessun elemento del vettore è in esaurimento.
R11	11.3	Controllo esaurimento alimento	Vettore nullo	Vettore: alimenti_nulli [ ] (con n_pezzi = 0) Soglia: 2	Nessun elemento del vettore è in esaurimento.	Nessun elemento del vettore è in esaurimento.
R11	11.4	Controllo esaurimento alimento	Vettore con tutti gli alimenti uguali	Vettore: alimenti_uguali [ ] (con n_pezzi = 1) Soglia: 2	Nessun elemento del vettore è in esaurimento.	Nessun elemento del vettore è in esaurimento.
R12	12.1	Visualizzazione ricette	Vettore di ricette pieno	Vettore: ricettario [ ] n_ricette: 50	Stampa tutti i campi delle ricette.	Stampa tutti i campi delle ricette.
R12	12.2	Visualizzazione ricette	Vettore di ricette vuoto	Vettore: ricettario [ ] n_ricette: 50	Non stampa nulla.	Non stampa nulla.

R12	12.3	Visualizzazione ricette	n_ricette diverso dalla lunghezza effettiva del vettore	Vettore: ricettario [ ] n_ricette: 30	Stampa i primi 80 alimenti.	Stampa i primi 80 alimenti.
R12	12.4	Visualizzazione ricette	n_ricette nullo	Vettore: ricettario [ ] n_ricette: 0	Non stampa nulla.	Non stampa nulla.
R13	13.1	Aggiunta nuova ricetta	File delle ricette esistente	File: “ricette.bin” Ricetta: nuova_ricetta	“Ricetta aggiunta.”	“Ricetta aggiunta.”
R13	13.2	Aggiunta nuova ricetta	File delle ricette non esistente	File: “abcd.bin” Ricetta: nuova_ricetta	“Errore, file non esistente.”	“Errore, file non esistente.”
R13	13.3	Aggiunta nuova ricetta	Ricetta nulla	File: “ricette.bin” Ricetta: ricetta_nulla (nome = “”, kcal = 0, ecc...)	“Ricetta aggiunta.”	“Ricetta aggiunta.”
R13	13.3	Aggiunta nuova ricetta	Ricetta già esistente	File: “ricette.bin” Ricetta: ricetta_esistente (nome = “Pasta al forno”)	“Ricetta già esistente!”	“Ricetta aggiunta.”
R14	14.1	Modifica ricetta	File delle ricette esistente	File: “ricette.bin” Ricetta: ricetta_modificata	“Ricetta modificata.”	“Ricetta modificata.”
R14	14.2	Modifica ricetta	File delle ricette non esistente	File: “abcd.bin” Ricetta: ricetta_modificata	“Errore, file non esistente.”	“Errore, file non esistente.”
R14	14.3	Modifica ricetta	Ricetta nulla	File: “alimenti.bin” Alimento: Ricetta_nulla (nome = “”)	“Ricetta modificata.”	“Ricetta modificata.”
R15	15.1	Cancella ricetta	File delle ricette esistente	File: “ricette.bin” ID ricetta: 1	“Ricetta cancellata.”	“Ricetta cancellata.”
R15	15.2	Cancella ricetta	File delle ricette non esistente	File: “abcd.bin” ID ricetta: 1	“Errore, file non esistente.”	“Errore, file non esistente.”

R15	15.3	Cancella ricetta	Ricetta non esistente	File: “ricette.bin” ID ricetta: 100	“Ricetta non esistente, cancellazione annullata!”	“Ricetta cancellata.”
R16	16.1	Visualizzazione menù settimanale	Vettore dei menù giornalieri pieno	Vettore: mesu_sett[ ] n_menu: 100 ID utente: 1	Stampa tutti i campi dei menu giornalieri che hanno 1 come id_utente.	Stampa tutti i campi dei menu giornalieri che hanno 1 come id_utente.
R16	16.2	Visualizzazione menù settimanale	Vettore dei menù giornalieri vuoto	Vettore: mesu_sett[ ] n_menu: 100 ID utente: 1	Non stampa nulla.	Non stampa nulla.
R16	16.3	Visualizzazione menù settimanale	n_menu nullo	Vettore: mesu_sett[ ] n_menu: 0 ID utente: 1	Non stampa nulla.	Non stampa nulla.
R16	16.4	Visualizzazione menù settimanale	ID utente errato	Vettore: mesu_sett[ ] n_menu: 100 ID utente: 53	Non stampa nulla.	Non stampa nulla.
R17	17.1	Aggiunta nuovo piatto nel menu settimanale	File dei menu settimanali esistente	File: “menu.bin” Menu: nuovo_menu Data menu: 7/6/2018	“Menù aggiunto!”	“Menù aggiunto!”
R17	17.2	Aggiunta nuovo piatto nel menu settimanale	File dei menu settimanali non esistente	File: “abcd.bin” Menu: nuovo_menu Data menu: 7/6/2018	“Errore, file non esistente.”	“Errore, file non esistente.”
R17	17.3	Aggiunta nuovo piatto nel menu settimanale	Ricetta non esistente	File: “menu.bin” Menu: nuovo_menu (id ricetta = 120) Data menu: 7/6/2018	“Ricetta errata!”	“Ricetta errata!”
R17	17.4	Aggiunta nuovo piatto nel menu settimanale	Ricetta non preparabile	File: “menu.bin” Menu: nuovo_menu (id ricetta = 3, i cui ingredienti non sono presenti in frigo) Data menu: 7/6/2018	“Ricetta non preparabile!”	“Ricetta non preparabile!”

R18	18.1	Modifica account a cui il menu giornaliero è riferito	File dei menu settimanali esistente	File: “menu.bin” ID menu da modificare: 2 ID nuovo utente: 4	“Menù aggiunto all’utente 4!”	“Menù aggiunto all’utente 4!”
R18	18.2	Modifica account a cui il menu giornaliero è riferito	File dei menu settimanali non esistente	File: “abcd.bin” ID menu da modificare: 2 ID nuovo utente: 4	“Errore, file non esistente.”	“Errore, file non esistente.”
R18	18.3	Modifica account a cui il menu giornaliero è riferito	ID utente errato	File: “menu.bin” ID menu da modificare: 2 ID nuovo utente: 49	“Utente non esistente, impossibile modificare!”	“Utente non esistente, impossibile modificare!”
R18	18.4	Modifica account a cui il menu giornaliero è riferito	ID utente uguale a quello del menu da modificare	File: “menu.bin” ID menu da modificare: 2 ID nuovo utente: 1	“L’ID utente deve essere diverso da sé stesso!”	“L’ID utente deve essere diverso da sé stesso!”
R19	19.1	Modifica piatto nel menu settimanale	File dei menu settimanali esistente	File: “menu.bin” ID menu da modificare: 2 Pasto da modificare: 1 Nuovo pasto: 4 (x 2 pers)	“Pasto modificato correttamente!”	“Pasto modificato correttamente”
R19	19.2	Modifica piatto nel menu settimanale	File dei menu settimanali non esistente	File: “abcd.bin” ID menu da modificare: 2 Pasto da modificare: 1 Nuovo pasto: 4 (x 2 pers)	“Errore, file non esistente.”	“Errore, file non esistente.”
R19	19.3	Modifica piatto nel menu settimanale	Nuovo pasto non preparabile	File: “menu.bin” ID menu da modificare: 2 Pasto da modificare: 1 Nuovo pasto: 3 (x 9 pers)	“Pasto non preparabile, impossibile modificare!”	“Pasto non preparabile, impossibile modificare!”
R20	20.1	Cancella piatto dal menu settimanale	File dei menu settimanali esistente	File: “menu.bin” ID utente: 2 Data menu: 6/6/2018	“Pasto del 6/6/2018 cancellato.”	“Pasto del 6/6/2018 cancellato.”

R20	20.2	Cancella piatto dal menu settimanale	File dei menu settimanali non esistente	File: “abcd.bin” ID utente: 2 Data menu: 6/6/2018	“Errore, file non esistente.”	“Errore, file non esistente.”
R20	20.3	Cancella piatto dal menu settimanale	ID utente errato	File: “menu.bin” ID utente: -1 Data menu: 6/6/2018	“Utente inesistente. Impossibile cancellare pasto”	“Pasto del 6/6/2018 cancellato.”
R21	21.1	Visualizzazione della lista della spesa	File della lista della spesa esistente	File: “lista.bin”	Stampa tutti gli alimenti scritti sul file.	Stampa tutti gli alimenti scritti sul file.
R21	21.2	Visualizzazione della lista della spesa	File dei menu settimanali non esistente	File: “abcd.bin”	“Errore, file non esistente.”	“Errore, file non esistente.”
R22	22.1	Cancellazione della lista della spesa	File della lista della spesa esistente	File: “lista.bin”	“Lista della spesa cancellata.”	“Lista della spesa cancellata.”
R22	22.2	Cancellazione della lista della spesa	File dei menu settimanali non esistente	File: “abcd.bin”	“Errore, file non esistente.”	“Errore, file non esistente.”
R23	23.1	Suggerimento ricetta	Nessun alimento in scadenza	Vettore scadenze: alimenti_scadenza [ ] (tutti gli elementi “”) n_alimenti: 100 Vettore ricette: ricettario [ ]	“Non ci sono alimenti in scadenza.”	“Non ci sono alimenti in scadenza.”
R23	23.2	Suggerimento ricetta	Tutti gli alimenti sono in scadenza	Vettore scadenze: alimenti_scadenza [ ] n_alimenti: 100 Vettore ricette: ricettario [ ]	Stampa tutte le ricette che hanno come ingredienti almeno un alimento in scadenza.	Stampa tutte le ricette che hanno come ingredienti almeno un alimento in scadenza.

R23	23.3	Suggerimento ricetta	Un solo alimento in scadenza	Vettore scadenze: alimenti_scadenza [ ] n_alimenti: 100 Vettore ricette: ricettario [ ]	Stampa tutte le ricette che hanno come ingrediente l'alimento in scadenza.	Stampa tutte le ricette che hanno come ingrediente l'alimento in scadenza.
R24	24.1	Ricerca ricetta per ID	ID ricetta massimo	Vettore ricette: ricettario [ ] ID ricetta: 50	Stampa la ricetta con ID 50	Stampa la ricetta con ID 50
R24	24.2	Ricerca ricetta per ID	ID ricetta minimo	Vettore ricette: ricettario [ ] ID ricetta: 1	Stampa la ricetta con ID 1	Stampa la ricetta con ID 1
R24	24.3	Ricerca ricetta per ID	ID ricetta errato	Vettore ricette: ricettario [ ] ID ricetta: 398	"La ricetta non è presente nel ricettario."	"La ricetta non è presente nel ricettario."
R25	25.1	Ricerca ricetta per ingrediente	Ingrediente esistente	Vettore ricette: ricettario [ ] Ingrediente: Latte	Stampa tutte le ricette che contengono l'ingrediente "Latte".	Stampa tutte le ricette che contengono l'ingrediente "Latte".
R25	25.2	Ricerca ricetta per ingrediente	Ingrediente errato	Vettore ricette: ricettario [ ] Ingrediente: Veleno	"Non esiste una ricetta che contiene l'ingrediente Veleno."	"Non esiste una ricetta che contiene l'ingrediente Veleno."
R25	25.3	Ricerca ricetta per ingrediente	Vettore vuoto	Vettore ricette: ricettario_vuoto [ ] Ingrediente: Sale	"Non esiste una ricetta che contiene l'ingrediente Sale."	"Non esiste una ricetta che contiene l'ingrediente Sale."
R26	26.1	Ricerca per insieme di ingredienti	Ingredienti esistenti	Vettore ricette: ricettario [ ] Vettore ingredienti: { "Latte", "Zucchero" }	Stampa tutte le ricette che contengono il vettore di ingredienti.	Stampa tutte le ricette che contengono il vettore di ingredienti.
R26	26.2	Ricerca per insieme di ingredienti	Ingredienti errati	Vettore ricette: ricettario [ ] Vettore ingredienti: { "Tartufo", "Cavallette" }	"Non esiste una ricetta che contiene il vettore di ingredienti."	"Non esiste una ricetta che contiene il vettore di ingredienti."
R26	26.3	Ricerca per insieme di ingredienti	Ingredienti errati	Vettore ricette: ricettario_vuoto [ ] Vettore ingredienti: { "Latte", "Zucchero" }	"Non esiste una ricetta che contiene il vettore di ingredienti."	"Non esiste una ricetta che contiene il vettore di ingredienti."

R27	27.1	Ricerca per kcal	Massimo kcal	Vettore ricette: ricettario [ ] Soglia kcal: 9000	Stampa tutte le ricette contenute nel vettore.	Stampa tutte le ricette contenute nel vettore.
R27	27.2	Ricerca per kcal	Minimo kcal	Vettore ricette: ricettario [ ] Soglia kcal: 0	"Non c'è alcuna ricetta con kcal al di sotto di 0"	"Non c'è alcuna ricetta con kcal al di sotto di 0"
R27	27.3	Ricerca per kcal	Ricetta con kcal = soglia	Vettore ricette: ricettario [ ] Soglia kcal: 209	Stampa tutte le ricette con kcal <= 209.	Stampa tutte le ricette con kcal <= 209.
R28	28.1	Ordinamento per kcal	Vettore di ricette pieno con kcal tutte diverse	Vettore ricette: ricettario [ ] n_ricette: 50	Stampa tutte le ricette ordinate in modo crescente in base alle kcal.	Stampa tutte le ricette ordinate in modo crescente in base alle kcal.
R28	28.1	Ordinamento per kcal	Vettore di ricette vuoto	Vettore ricette: ricettario_vuoto [ ] n_ricette: 50	"Il ricettario è vuoto."	"Il ricettario è vuoto."
R28	28.1	Ordinamento per kcal	Vettore di ricette pieno con kcal tutte uguali	Vettore ricette: ricettario [ ] n_ricette: 50	Stampa il vettore così com'e.	Stampa il vettore così com'e.

4.2. Esiti del Piano di Test

Il piano di test è risultato positivo, a parte per alcuni punti critici (evidenziati in giallo) nei quali i risultati attesi sono diversi da quelli ottenuti:

- nella cancellazione dell'utente, se l'ID dell'utente che deve essere cancellato è errato, non verrà mostrato alcun messaggio di errore e potrebbe essere cancellato un record del file non voluto;
- nella cancellazione di una ricetta, se l'ID della ricetta che deve essere cancellata è errato, non verrà mostrato alcun messaggio di errore e potrebbe essere cancellato un record del file non voluto;
- nella cancellazione di un pasto dal menù settimanale, se l'ID dell'utente per il quale deve essere cancellata il pasto è errato, non verrà mostrato alcun messaggio di errore e potrebbe essere cancellato un record del file non voluto;

Tuttavia questi non sono veri e propri punti di debolezza del programma, poiché prima di richiamare le funzioni che svolgono la cancellazione, ci sono istruzioni che verificano la correttezza degli ID, dunque non si potranno presentare mai bug di questo genere.

L'unico bug possibile è quello relativo all'aggiunta di una nuova ricetta, per la quale non viene effettuato un controllo sulla pre-esistenza del nome della ricetta inserita. Perciò avendo due o più ricette con lo stesso nome si potrebbero verificare anomalie in altre funzioni che utilizzano il vettore di ricette.

Mentre per quanto riguarda il testing delle funzioni con cUnit, le asserzioni che sono risultate false sono:

- giorno_precedente (data_errata) → passando come parametro alla funzione giorno_precedente ( ) una data errata (es. 30/2/2018), il calcolo del giorno precedente risulta errato, dunque sarà diverso da 29/2/2018.
- get_giorno_settimana_intero (data_errata) → passando come parametro alla funzione get_giorno_settimana_intero ( ) una data errata (es. 31/2/2018), l'intero restituito sarà lo stesso di get_giorno_settimana_intero (3/3/2018). Sarebbe meglio che la funzione controllasse se la data è errata e in tal caso restituisse un valore diverso da tutti gli altri (es. -1) .
- indice_id_alimento (array_alimenti_pieno, NUMERO_ALIMENTI, 130) → passando come parametro alla funzione indice_id_alimento ( ) un ID non esistente nel vettore (es. 130), la funzione restituirà esattamente il numero di alimenti (100) in quanto l'indice viene incrementato finché il vettore non finisce o non si trova l'id cercato. Sarebbe meglio che la funzione restituisse un indice diverso dagli altri (es. -1) nel caso in cui l'ID non esista nel vettore.
- vale lo stesso discorso per anche per le funzioni indice_id_menu ( ) e indice_id_ricetta ( ).

- ricerca_utente ("elenco_utenti.bin", 1000) → passando come parametro alla funzione ricerca_utente ( ) un ID utente non esistente (es. 1000), la funzione dovrebbe restituire NULL, ma non lo fa correttamente.

In realtà anche questi non sono veri e propri errori per il programma, in quanto non verrà generata mai una data errata (poiché viene presa dal sistema), e l'esistenza degli ID viene controllata prima di chiamare le funzioni sopra citate.

Perciò il programma rimane solido nonostante queste piccole imperfezioni.