Recepteskonyv

Generated by Doxygen 1.12.0

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Data Structure Documentation

3.1 Ingridient Struct Reference

```
#include <Ingridient.h>
```

Data Fields

- char IngName [40]
- MeasurementType MType
- struct Ingridient ∗ NextNode

3.1.1 Detailed Description

Ingridient struct that stores and Ingridient name, Mesurment Type of the ingridient in the form of an enum and an ingridient pointer that is used to make a linked list.

3.1.2 Field Documentation

3.1.2.1 IngName

```
char Ingridient::IngName[40]
```

The ingridients name.

3.1.2.2 MType

```
MeasurementType Ingridient::MType
```

The type of the ingridnient.

3.1.2.3 NextNode

```
struct Ingridient* Ingridient::NextNode
```

The pointer to the next Ingridient.

The documentation for this struct was generated from the following file:

- C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/Ingridient.h

3.2 Recipe Struct Reference

```
#include <Recipe.h>
```

Data Fields

- char RecName [100]
- struct Recipe ∗ NextNode
- RecipeIngridient ∗ RecIng
- bool bIsFavorite

3.2.1 Detailed Description

The struct that stores recipe values.

3.2.2 Field Documentation

3.2.2.1 bIsFavorite

```
bool Recipe::bIsFavorite
```

A boolean value that dictates if the recipe is Favorite or not.

3.2.2.2 NextNode

```
struct Recipe* Recipe::NextNode
```

The pointer to the next recipe in the recipe linked list.

3.2.2.3 RecIng

```
RecipeIngridient* Recipe::RecIng
```

The pointer to the first element of the RecipeIngredient linked list.

3.2.2.4 RecName

```
char Recipe::RecName[100]
```

The name of the recipe

The documentation for this struct was generated from the following file:

- C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/Recipe.h

3.3 RecipeIngridient Struct Reference

```
#include <RecipeIngridient.h>
```

Data Fields

- int IngAmount
- Ingridient ∗ IngData
- struct RecipeIngridient ∗ NextNode

3.3.1 Detailed Description

A wrapper struct that connects the Ingridient stuct and the Recipe struct and it stores ingridient amount. The Ing↩ Data points to the selected Ingridient. The NextNode is used to point to the next element of the RecipeIngridinent linked list.

3.3.2 Field Documentation

3.3.2.1 IngAmount

```
int RecipeIngridient::IngAmount
```

The amount of the selected ingridient.

3.3.2.2 IngData

```
Ingridient* RecipeIngridient::IngData
```

The pointer of the selected ingridient.

3.3.2.3 NextNode

```
struct RecipeIngridient* RecipeIngridient::NextNode
```

The pointer to then next RecipeIngridient in the chained list.

The documentation for this struct was generated from the following file:

- C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/RecipeIngridient.h

Chapter 4

File Documentation

4.1 C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/Ingridient.c File Reference

```
#include "Ingridient.h"
#include <stdio.h>
#include "Recipe.h"
#include "debugmalloc.h"
```

Functions

- Ingridient ∗ CreateIngridient (char ∗Name, MeasurementType Type)
- void AddIngridientTo (Ingridient ∗list, Ingridient ∗ToAdd)
- void AddOrCreateIng (Ingridient ∗∗list, Ingridient ∗ToAdd)
- void DeleteIngridient (Ingridient ∗∗listPtr, Ingridient ∗ToDelete, Recipe ∗∗RecListPtr)
- Ingridient ∗ FindIngridientByName (Ingridient ∗list, char ∗IngName)
- void PrintIngNameByIndex (Ingridient ∗list, int IngridientIndex)
- void PrintAllIng (Ingridient ∗list)
- void DeleteAllIng (Ingridient ∗∗listPtr, Recipe ∗∗RecListPtr)

4.1.1 Function Documentation

4.1.1.1 AddIngridientTo()

```
void AddIngridientTo (
            Ingridient ∗ list,
            Ingridient ∗ ToAdd)
```

Adds an Ingridient to an already existing Ingridient linked list.

Parameters

list	The first address of the list.
ToAdd	The Pointer of Ingridient that will be added to the list.

4.1.1.2 AddOrCreateIng()

```
void AddOrCreateIng (
             Ingridient ** list,
             Ingridient * ToAdd)
```

This function is not used.

Parameters

list	
ToAdd	

4.1.1.3 CreateIngridient()

```
Ingridient * CreateIngridient (
             char * Name,
             MeasurementType Type)
```

Creates an Ingridient pointer.

Parameters

Name	Name of the ingridient.
Type	Mesurment type of the ingridient.

Returns

Ingridient pointer.

4.1.1.4 DeleteAllIng()

```
void DeleteAllIng (
             Ingridient ** listPtr,
             Recipe ** RecListPtr)
```

Deletes all ingrideients. This will delete all the recipes too

Parameters

listPtr	The address of the first element of the Recipe linked list.
RecListPtr	The address of the first element of the Ingridient linked list.

4.1.1.5 DeleteIngridient()

```
void DeleteIngridient (
             Ingridient ** listPtr,
             Ingridient * ToDelete,
             Recipe ** RecListPtr)
```

Deletes and element of the linked list.

Parameters

listPtr	Pointer of the first address.
ToDelete	The list element that will be deleted.
RecListPtr	Pointer of the first element of the Recipe linked list.

4.1.1.6 FindIngridientByName()

```
Ingridient * FindIngridientByName (
             Ingridient * list,
             char * IngName)
```

Finds an ingridient by name

Parameters

list	The address of the first element of the ingridient linked list.
IngName	A name of the ingrident that is being searched.

Returns

The address of the Ingridient.

4.1.1.7 PrintAllIng()

```
void PrintAllIng (
             Ingridient * list)
```

Prints all Ingridients.

Parameters

list	The address of the first element of the Ingridient linked list.

4.1.1.8 PrintIngNameByIndex()

```
void PrintIngNameByIndex (
             Ingridient * list,
             int IngridientIndex)
```

Prints the elemnt definded by the index.

Parameters

list	The address first element of the Ingridient linked list.
IngridientIndex	an index.

4.2 C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/Ingridient.h File Reference

```
#include "debugmalloc.h"
```

Data Structures

- struct Ingridient

Typedefs

- typedef struct Recipe Recipe
- typedef enum MeasurementType MeasurementType
- typedef struct Ingridient Ingridient

Enumerations

- enum MeasurementType { kilogramm , liter , darab }

Functions

- Ingridient ∗ CreateIngridient (char ∗Name, MeasurementType Type)
- void AddIngridientTo (Ingridient ∗list, Ingridient ∗ToAdd)
- void AddOrCreateIng (Ingridient ∗∗list, Ingridient ∗ToAdd)
- void DeleteIngridient (Ingridient ∗∗listPtr, Ingridient ∗ToDelete, Recipe ∗∗RecListPtr)
- Ingridient ∗ FindIngridientByName (Ingridient ∗list, char ∗IngName)
- void PrintIngNameByIndex (Ingridient ∗list, int IngridientIndex)
- void PrintAllIng (Ingridient ∗list)
- void DeleteAllIng (Ingridient ∗∗listPtr, Recipe ∗∗RecListPtr)

4.2.1 Typedef Documentation

4.2.1.1 Ingridient

```
typedef struct Ingridient Ingridient
```

Ingridient struct that stores and Ingridient name, Mesurment Type of the ingridient in the form of an enum and an ingridient pointer that is used to make a linked list.

4.2.1.2 MeasurementType

```
typedef enum MeasurementType MeasurementType
```

Enum that indicates the ingridient mesurement type.

4.2.1.3 Recipe

```
typedef struct Recipe Recipe
```

4.2.2 Enumeration Type Documentation

4.2.2.1 MeasurementType

```
enum MeasurementType
```

Enum that indicates the ingridient mesurement type.

Enumerator

kilogramm	
liter	
darab	

4.2.3 Function Documentation

4.2.3.1 AddIngridientTo()

```
void AddIngridientTo (
            Ingridient * list,
            Ingridient * ToAdd)
```

Adds an Ingridient to an already existing Ingridient linked list.

Parameters

list	The first address of the list.
ToAdd	The Pointer of Ingridient that will be added to the list.

4.2.3.2 AddOrCreateIng()

```
void AddOrCreateIng (
            Ingridient ** list,
            Ingridient * ToAdd)
```

This function is not used.

Parameters

list	
ToAdd	

4.2.3.3 CreateIngridient()

```
Ingridient * CreateIngridient (
            char * Name,
            MeasurementType Type)
```

Creates an Ingridient pointer.

Parameters

Name	Name of the ingridient.
Type	Mesurment type of the ingridient.

Returns

Ingridient pointer.

4.2.3.4 DeleteAllIng()

```
void DeleteAllIng (
            Ingridient ** listPtr,
            Recipe ** RecListPtr)
```

Deletes all ingrideients. This will delete all the recipes too

Parameters

listPtr	The address of the first element of the Recipe linked list.
RecListPtr	The address of the first element of the Ingridient linked list.

4.2.3.5 DeleteIngridient()

```
void DeleteIngridient (
            Ingridient ** listPtr,
            Ingridient * ToDelete,
            Recipe ** RecListPtr)
```

Deletes and element of the linked list.

Parameters

listPtr	Pointer of the first address.
ToDelete	The list element that will be deleted.
RecListPtr	Pointer of the first element of the Recipe linked list.

4.2.3.6 FindIngridientByName()

```
Ingridient * FindIngridientByName (
            Ingridient * list,
            char * IngName)
```

Finds an ingridient by name

Parameters

list	The address of the first element of the ingridient linked list.
IngName	A name of the ingrident that is being searched.

Returns

The address of the Ingridient.

4.2.3.7 PrintAllIng()

```
void PrintAllIng (
            Ingridient * list)
```

Prints all Ingridients.

Parameters

list	The address of the first element of the Ingridient linked list.

4.2.3.8 PrintIngNameByIndex()

```
void PrintIngNameByIndex (
              Ingridient * list,
              int IngridientIndex)
```

Prints the elemnt definded by the index.

Parameters

list	The address first element of the Ingridient linked list.
IngridientIndex	an index.

4.3 Ingridient.h

Go to the documentation of this file.

```
//
// Created by hejag on 23/11/2024.
//

#ifndef INGRIDIENT_H
#define INGRIDIENT_H
#include "debugmalloc.h"


typedef struct Recipe Recipe;

typedef enum MeasurementType {
    kilogramm,
    liter,
    darab
} MeasurementType;

typedef struct Ingridient {
 char IngName[40];
 MeasurementType MType;
 struct Ingridient *NextNode;
} Ingridient;

Ingridient *CreateIngridient(char *Name, MeasurementType Type);

void AddIngridientTo(Ingridient *list, Ingridient *ToAdd);

void AddOrCreateIng(Ingridient **list, Ingridient *ToAdd);

void DeleteIngridient(Ingridient **listPtr, Ingridient *ToDelete, Recipe **RecListPtr);

Ingridient *FindIngridientByName(Ingridient *list, char *IngName);

void PrintIngNameByIndex(Ingridient *list, int IngridientIndex);

void PrintAllIng(Ingridient *list);

void DeleteAllIng(Ingridient **listPtr, Recipe **RecListPtr);
#endif //INGRIDIENT_H
```

4.4 C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/input.c File Reference

```
#include "debugmalloc.h"
#include "input.h"
#include <stdio.h>
#include "Ingridient.h"
#include "Recipe.h"
```

Functions

- Ingridient ∗ IngInput ()
- MeasurementType IngTypeEnumerator (char ∗TypeString)
- char ∗ IngTypeDenumerator (MeasurementType Mtype)
- void StrKiller (char ∗str)
- Recipe ∗ RecInput (Ingridient ∗list)
- bool IsFavChecker (char ∗TypeString)

4.4.1 Function Documentation

4.4.1.1 IngInput()

```
Ingridient * IngInput ()
```

4.4.1.2 IngTypeDenumerator()

```
char * IngTypeDenumerator (
            MeasurementType Mtype)
```

Converts MeasurementType enum to a string.

Parameters

Mtype	MeasurementType enum that will be converted.

Returns

Address of the string.

4.4.1.3 IngTypeEnumerator()

```
MeasurementType IngTypeEnumerator (
            char * TypeString)
```

Converts a string to a MeasurementType enum.

Parameters

TypeString	The that will be converted

Returns

MeasurementType.

4.4.1.4 IsFavChecker()

```
bool IsFavChecker (
            char * TypeString)
```

Converts a string to a true or false value.

Parameters

TypeString	The string that will be converted.

Returns

True or false value.

4.4.1.5 RecInput()

```
Recipe * RecInput (
            Ingridient * list)
```

4.4.1.6 StrKiller()

```
void StrKiller (
            char * str)
```

Empties a string

Parameters

str	The string.

4.5 C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/input.h File Reference

```
#include "debugmalloc.h"
#include "Ingridient.h"
#include "Recipe.h"
```

Typedefs

- typedef enum IngInputState IngInputState
- typedef enum RecInputState RecInputState

Enumerations

- enum IngInputState { IName , IType , CreateIng , IReturn }
- enum RecInputState {
  RName , IsFav , IngCount , RecipeIngName ,
  RecipeIngNumber , RecipeIngType , CreateRecipeIng , RecipeCreation ,
  RReturn }

Functions

- Ingridient ∗ IngInput ()
- Recipe ∗ RecInput (Ingridient ∗list)
- MeasurementType IngTypeEnumerator (char ∗TypeString)
- char ∗ IngTypeDenumerator (MeasurementType Mtype)
- bool IsFavChecker (char ∗TypeString)
- void StrKiller (char ∗str)

4.5.1 Typedef Documentation

4.5.1.1 IngInputState

```
typedef enum IngInputState IngInputState
```

The states used for "Osszetevok.txt" file reading.

4.5.1.2 RecInputState

```
typedef enum RecInputState RecInputState
```

The states used for "Receptek.txt" file reading.

4.5.2 Enumeration Type Documentation

4.5.2.1 IngInputState

```
enum IngInputState
```

The states used for "Osszetevok.txt" file reading.

Enumerator

IName	
IType	
CreateIng	
IReturn	

4.5.2.2 RecInputState

```
enum RecInputState
```

The states used for "Receptek.txt" file reading.

Enumerator

RName	
IsFav	
IngCount	
RecipeIngName	
RecipeIngNumber	
RecipeIngType	
CreateRecipeIng	
RecipeCreation	
RReturn	

4.5.3 Function Documentation

4.5.3.1 IngInput()

```
Ingridient * IngInput ()
```

4.5.3.2 IngTypeDenumerator()

```
char * IngTypeDenumerator (
            MeasurementType Mtype)
```

Converts MeasurementType enum to a string.

Parameters

Mtype	MeasurementType enum that will be converted.

Returns

Address of the string.

4.5.3.3 IngTypeEnumerator()

```
MeasurementType IngTypeEnumerator (
            char * TypeString)
```

Converts a string to a MeasurementType enum.

Parameters

TypeString	The that will be converted

Returns

MeasurementType.

4.5.3.4 IsFavChecker()

```
bool IsFavChecker (
            char * TypeString)
```

Converts a string to a true or false value.

Parameters

TypeString	The string that will be converted.

Returns

True or false value.

4.5.3.5 RecInput()

```
Recipe * RecInput (
             Ingridient * list)
```

4.5.3.6 StrKiller()

```
void StrKiller (
             char * str)
```

Empties a string

Parameters

str	The string.

4.6 input.h

Go to the documentation of this file.

```
//
// Created by hejag on 23/11/2024.
//

#ifndef INPUT_H
#define INPUT_H
#include "debugmalloc.h"
#include "Ingridient.h"
#include "Recipe.h"

Ingridient *IngInput();

Recipe *RecInput(Ingridient *list);

typedef enum IngInputState {
    IName,
    IType,
    CreateIng,
    IReturn
} IngInputState;

typedef enum RecInputState {
    RName,
    IsFav,
    IngCount,
    RecipeIngName,
    RecipeIngNumber,
    RecipeIngType,
    CreateRecipeIng,
    RecipeCreation,
    RReturn
} RecInputState;

MeasurementType IngTypeEnumerator(char *TypeString);

char *IngTypeDenumerator(MeasurementType Mtype);

bool IsFavChecker(char *TypeString);

void StrKiller(char *str);

#endif //INPUT_H
```

4.7 C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/main.c File Reference

```
#include <stdio.h>
#include <string.h>
#include "Ingridient.h"
#include "input.h"
#include "Menu.h"
#include "debugmalloc.h"
```

Functions

- int main (void)

4.7.1 Function Documentation

4.7.1.1 main()

```
int main (
            void )
```

4.8 C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/Menu.c File Reference

```
#include "Menu.h"
#include <stdio.h>
#include "input.h"
#include "debugmalloc.h"
```

Functions

- void LogoPrinter ()
- int MainMenu (Ingridient **IngListPtr, Recipe **RecipeList)
- void SearchMenu (Ingridient **IngList, Recipe **RecipeList)
- void RecipeMenu (Ingridient **IngList, Recipe **RecipeList)
- void IngManageMenu (Ingridient **IngList, Recipe **RecipeList)
- void FavMenu (Recipe **RecipeList)
- void RandomRecipe (Recipe *RecipeList)

4.8.1 Function Documentation

4.8.1.1 FavMenu()

```
void FavMenu (
            Recipe ** RecipeList)
```

The menu for modifying the favorite parameter of the recipe stuct.

Parameters

RecipeList	The address of the first element of the Recipe linked list.

4.8.1.2 IngManageMenu()

```
void IngManageMenu (
            Ingridient ** IngList,
            Recipe ** RecipeList)
```

Ingridient menu for modifying ingridients.

Parameters

IngList	The address of the first element of the Ingridient linked list.
RecipeList	The address of the first element of the Recipe linked list.

4.8.1.3 LogoPrinter()

```
void LogoPrinter ()
```

Prints out the logo.

4.8.1.4 MainMenu()

```
int MainMenu (
            Ingridient ** IngListPtr,
            Recipe ** RecipeList)
```

This is the main menu function this from this function the user can access the other menus.

Parameters

IngListPtr	The address of the first element of the Ingridient linked list.
RecipeList	The address of the first element of the Recipe linked list.

Returns

If the user exits the program with the exit option the return will be 0.

4.8.1.5 RandomRecipe()

```
void RandomRecipe (
            Recipe * RecipeList)
```

This function will print out a random recipe.

Parameters

RecipeList	The address of the first element of the Recipe linked list.

4.8.1.6 RecipeMenu()

```
void RecipeMenu (
            Ingridient ** IngList,
            Recipe ** RecipeList)
```

Recipe menu for modifying recipes.

Parameters

IngList	The address of the first element of the Ingridient linked list.
RecipeList	The address of the first element of the Recipe linked list.

4.8.1.7 SearchMenu()

```
void SearchMenu (
            Ingridient ** IngList,
            Recipe ** RecipeList)
```

The Recipe search menu.

Parameters

IngList	The address of the first element of the Ingridient linked list.
RecipeList	The address of the first element of the Recipe linked list.

4.9 C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/Menu.h File Reference

```
#include "Recipe.h"
#include "debugmalloc.h"
```

Functions

- void LogoPrinter ()
- int MainMenu (Ingridient **IngListPtr, Recipe **RecipeList)
- void SearchMenu (Ingridient **IngList, Recipe **RecipeList)
- void RecipeMenu (Ingridient **IngList, Recipe **RecipeList)
- void IngManageMenu (Ingridient **IngList, Recipe **RecipeList)
- void FavMenu (Recipe **RecipeList)
- void RandomRecipe (Recipe *RecipeList)

4.9.1 Function Documentation

4.9.1.1 FavMenu()

```
void FavMenu (
            Recipe ** RecipeList)
```

The menu for modifying the favorite parameter of the recipe stuct.

Parameters

RecipeList	The address of the first element of the Recipe linked list.

4.9.1.2 IngManageMenu()

```
void IngManageMenu (
            Ingridient ** IngList,
            Recipe ** RecipeList)
```

Ingridient menu for modifying ingridients.

Parameters

IngList	The address of the first element of the Ingridient linked list.
RecipeList	The address of the first element of the Recipe linked list.

4.9.1.3 LogoPrinter()

```
void LogoPrinter ()
```

Prints out the logo.

4.9.1.4 MainMenu()

```
int MainMenu (
            Ingridient ** IngListPtr,
            Recipe ** RecipeList)
```

This is the main menu function this from this function the user can access the other menus.

Parameters

IngListPtr	The address of the first element of the Ingridient linked list.
RecipeList	The address of the first element of the Recipe linked list.

Returns

If the user exits the program with the exit option the return will be 0.

4.9.1.5 RandomRecipe()

```
void RandomRecipe (
            Recipe * RecipeList)
```

This function will print out a random recipe.

Parameters

RecipeList	The address of the first element of the Recipe linked list.

4.9.1.6 RecipeMenu()

```
void RecipeMenu (
             Ingridient ** IngList,
             Recipe ** RecipeList)
```

Recipe menu for modifying recipes.

Parameters

IngList	The address of the first element of the Ingridient linked list.
RecipeList	The address of the first element of the Recipe linked list.

4.9.1.7 SearchMenu()

```
void SearchMenu (
             Ingridient ** IngList,
             Recipe ** RecipeList)
```

The Recipe search menu.

Parameters

IngList	The address of the first element of the Ingridient linked list.
RecipeList	The address of the first element of the Recipe linked list.

4.10 Menu.h

Go to the documentation of this file.

```
//
// Created by hejag on 24/11/2024.
//

#ifndef MENU_H
#define MENU_H
#include "Recipe.h"
#include "debugmalloc.h"

void LogoPrinter();

int MainMenu(Ingridient** IngListPtr, Recipe** RecipeList);

void SearchMenu(Ingridient** IngList, Recipe** RecipeList);

void RecipeMenu(Ingridient** IngList, Recipe** RecipeList);

void IngManageMenu(Ingridient** IngList, Recipe** RecipeList);

void FavMenu(Recipe** RecipeList);

void RandomRecipe(Recipe* RecipeList);




#endif //MENU_H
```

4.11 C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/Recipe.c File Reference

```
#include "Recipe.h"
#include <stdio.h>
#include "input.h"
#include "Ingridient.h"
#include "debugmalloc.h"
```

Functions

- Recipe ∗ CreateRecipe (char ∗Name, bool bIsFav)
- void AssignIngridients (Recipe ∗Rec, RecipeIngridient ∗RecIngridient)
- void AddRecipeTo (Recipe ∗list, Recipe ∗ToAdd)
- void AddOrCreateRecipe (Recipe ∗∗list, Recipe ∗ToAdd)
- void DeleteRecipe (Recipe ∗∗listPtr, Recipe ∗ToDelete)
- Recipe ∗ FindRecipeByName (Recipe ∗list, char ∗RecipeName)
- void RecipePrint (Recipe ∗PRecipe)
- void PrintRecipeByIng (Recipe ∗list)
- void PrintFavRecipes (Recipe ∗list)
- void SetFavorite (Recipe ∗list, char ∗SetFavName, bool isFav)
- void PrintAllRecipes (Recipe ∗list)
- void AddRecipeFromIO (Ingridient ∗IngList, Recipe ∗RecList)
- bool AddRecipeFromIOIsFav (char ∗IsFav)
- void DeleteAllRecipe (Recipe ∗list)
- int RecipeListLen (Recipe ∗list)
- char ∗ BoolFavConverter (bool IsFav)

4.11.1 Function Documentation

4.11.1.1 AddOrCreateRecipe()

```
void AddOrCreateRecipe (
            Recipe ** list,
            Recipe * ToAdd)
```

This function is not used in the program.

Parameters

list	
ToAdd	

4.11.1.2 AddRecipeFromIO()

```
void AddRecipeFromIO (
            Ingridient * IngList,
            Recipe * RecList)
```

This will add a new recipe to the Recipe linked list. This function gets input from the user.

Parameters

IngList	The address of the first element of the recipe linked list.
RecList	The address of the first element of the ingridient linked list.

4.11.1.3 AddRecipeFromIOIsFav()

```
bool AddRecipeFromIOIsFav (
             char * IsFav)
```

Gets a string and determines that is it a true or false value when converted to a boolean.

Parameters

IsFav	The string.

Returns

The boolean value.

4.11.1.4 AddRecipeTo()

```
void AddRecipeTo (
             Recipe * list,
             Recipe * ToAdd)
```

Adds a Recipe to the Recipe linked list.

Parameters

list	The address of the first element of the recipe linked list.
ToAdd	The address of the Recipe that will be added to the list.

4.11.1.5 AssignIngridients()

```
void AssignIngridients (
             Recipe * Rec,
             RecipeIngridient * RecIngridient)
```

Assigns the RecipeIngridient linked list to and elemnet of Recipe linkec list.

Parameters

Rec	An element of the Recipe linked list.
RecIngridient	The first element of the RecinpeIngridient lined list.

4.11.1.6 BoolFavConverter()

```
char * BoolFavConverter (
             bool IsFav)
```

Converts bool to char value.

Parameters

IsFav	bool to be converted.

Returns

char pointer (string).

4.11.1.7 CreateRecipe()

```
Recipe * CreateRecipe (
            char * Name,
            bool bIsFav)
```

This function creates a new recipe.

Parameters

Name	The name of the recipe to create.
bIsFav	Whether to mark this recipe or not.

Returns

Newly created recipe.

4.11.1.8 DeleteAllRecipe()

```
void DeleteAllRecipe (
            Recipe * list)
```

Deletes all Recipe from the Recipe linked list.

Parameters

list	The address of the first element of the recipe linked list.

4.11.1.9 DeleteRecipe()

```
void DeleteRecipe (
            Recipe ** listPtr,
            Recipe * ToDelete)
```

Deletes an element of the Recipe linked list specified by the ToDelete pointer.

Parameters

listPtr	The address of the first element of the recipe linked list.
ToDelete	The address of the element of the recipe linked list that will be deleted.

4.11.1.10 FindRecipeByName()

```
Recipe * FindRecipeByName (
            Recipe * list,
            char * RecipeName)
```

Finds a recipe by name

Parameters

list	The address of the first element of the recipe linked list.
RecipeName	The name of the recipe.

Returns

Address of the Recipe specified by name.

4.11.1.11 PrintAllRecipes()

```
void PrintAllRecipes (
            Recipe * list)
```

Prints all recipes.

Parameters

list	The address of the first element of the recipe linked list.

4.11.1.12 PrintFavRecipes()

```
void PrintFavRecipes (
            Recipe * list)
```

Prints the favorite recipes

Parameters

list	The address of the first element of the recipe linked list.

4.11.1.13 PrintRecipeByIng()

```
void PrintRecipeByIng (
            Recipe * list)
```

Prints a recipe containing the specified ingridient.

Parameters

list	The address of the first element of the recipe linked list.

4.11.1.14 RecipeListLen()

```
int RecipeListLen (
            Recipe * list)
```

Gets the lenght of the recipe linked list.

Parameters

list	The address of the first element of the recipe linked list.

Returns

an intiger value that is the lenght of the list.

4.11.1.15 RecipePrint()

```
void RecipePrint (
            Recipe * PRecipe)
```

Prints the recipe of the specified address.

Parameters

PRecipe	The recipe address.

4.11.1.16 SetFavorite()

```
void SetFavorite (
            Recipe * list,
            char * SetFavName,
            bool isFav)
```

Sets an element of the Recipe linked list to favorite or not favorite.

Parameters

list	The address of the first element of the recipe linked list.
SetFavName	The name of the SelectFavorite Recipe.
isFav	Favorite or not.

4.12 C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/Recipe.h File Reference

```
#include <stdbool.h>
#include "RecipeIngridient.h"
#include "Ingridient.h"
#include "debugmalloc.h"
```

Data Structures

- struct Recipe

Typedefs

- typedef struct Recipe Recipe

Functions

- Recipe ∗ CreateRecipe (char ∗Name, bool bIsFav)
- void AssignIngridients (Recipe ∗Rec, RecipeIngridient ∗RecIngridient)
- void AddRecipeTo (Recipe ∗list, Recipe ∗ToAdd)
- void AddOrCreateRecipe (Recipe ∗∗list, Recipe ∗ToAdd)
- void DeleteRecipe (Recipe ∗∗listPtr, Recipe ∗ToDelete)
- Recipe ∗ FindRecipeByName (Recipe ∗list, char ∗RecipeName)
- void RecipePrint (Recipe ∗PRecipe)
- void PrintRecipeByIng (Recipe ∗list)
- void PrintFavRecipes (Recipe ∗list)
- void SetFavorite (Recipe ∗list, char ∗SetFavName, bool isFav)
- void PrintAllRecipes (Recipe ∗list)
- void AddRecipeFromIO (Ingridient ∗IngList, Recipe ∗RecList)
- bool AddRecipeFromIOIsFav (char ∗IsFav)
- void DeleteAllRecipe (Recipe ∗list)
- int RecipeListLen (Recipe ∗list)
- char ∗ BoolFavConverter (bool IsFav)

4.12.1 Typedef Documentation

4.12.1.1 Recipe

```
typedef struct Recipe Recipe
```

The struct that stores recipe values.

4.12.2 Function Documentation

4.12.2.1 AddOrCreateRecipe()

```
void AddOrCreateRecipe (
            Recipe ** list,
            Recipe * ToAdd)
```

This function is not used in the program.

Parameters

list	
ToAdd	

4.12.2.2 AddRecipeFromIO()

```
void AddRecipeFromIO (
            Ingridient * IngList,
            Recipe * RecList)
```

This will add a new recipe to the Recipe linked list. This function gets input from the user.

Parameters

IngList	The address of the first element of the recipe linked list.
RecList	The address of the first element of the ingridient linked list.

4.12.2.3 AddRecipeFromIOIsFav()

```
bool AddRecipeFromIOIsFav (
             char * IsFav)
```

Gets a string and determines that is it a true or false value when converted to a boolean.

Parameters

IsFav	The string.

Returns

The boolean value.

4.12.2.4 AddRecipeTo()

```
void AddRecipeTo (
             Recipe * list,
             Recipe * ToAdd)
```

Adds a Recipe to the Recipe linked list.

Parameters

list	The address of the first element of the recipe linked list.
ToAdd	The address of the Recipe that will be added to the list.

4.12.2.5 AssignIngridients()

```
void AssignIngridients (
             Recipe * Rec,
             RecipeIngridient * RecIngridient)
```

Assigns the RecipeIngridient linked list to and elemnet of Recipe linkec list.

Parameters

Rec	An element of the Recipe linked list.
RecIngridient	The first element of the RecinpeIngridient lined list.

4.12.2.6 BoolFavConverter()

```
char * BoolFavConverter (
             bool IsFav)
```

Converts bool to char value.

Parameters

IsFav	bool to be converted.

Returns

char pointer (string).

4.12.2.7 CreateRecipe()

```
Recipe * CreateRecipe (
            char * Name,
            bool bIsFav)
```

This function creates a new recipe.

Parameters

Name	The name of the recipe to create.
bIsFav	Whether to mark this recipe or not.

Returns

Newly created recipe.

4.12.2.8 DeleteAllRecipe()

```
void DeleteAllRecipe (
            Recipe * list)
```

Deletes all Recipe from the Recipe linked list.

Parameters

list	The address of the first element of the recipe linked list.

4.12.2.9 DeleteRecipe()

```
void DeleteRecipe (
            Recipe ** listPtr,
            Recipe * ToDelete)
```

Deletes an element of the Recipe linked list specified by the ToDelete pointer.

Parameters

listPtr	The address of the first element of the recipe linked list.
ToDelete	The address of the element of the recipe linked list that will be deleted.

4.12.2.10 FindRecipeByName()

```
Recipe * FindRecipeByName (
            Recipe * list,
            char * RecipeName)
```

Finds a recipe by name

Parameters

list	The address of the first element of the recipe linked list.
RecipeName	The name of the recipe.

Returns

Address of the Recipe specified by name.

4.12.2.11 PrintAllRecipes()

```
void PrintAllRecipes (
            Recipe * list)
```

Prints all recipes.

Parameters

list	The address of the first element of the recipe linked list.

4.12.2.12 PrintFavRecipes()

```
void PrintFavRecipes (
            Recipe * list)
```

Prints the favorite recipes

Parameters

list	The address of the first element of the recipe linked list.

4.12.2.13 PrintRecipeByIng()

```
void PrintRecipeByIng (
            Recipe * list)
```

Prints a recipe containing the specified ingridient.

Parameters

list	The address of the first element of the recipe linked list.

4.12.2.14 RecipeListLen()

```
int RecipeListLen (
            Recipe * list)
```

Gets the lenght of the recipe linked list.

Parameters

list	The address of the first element of the recipe linked list.

Returns

an intiger value that is the lenght of the list.

4.12.2.15 RecipePrint()

```
void RecipePrint (
            Recipe * PRecipe)
```

Prints the recipe of the specified address.

Parameters

PRecipe	The recipe address.

4.12.2.16 SetFavorite()

```
void SetFavorite (
            Recipe * list,
            char * SetFavName,
            bool isFav)
```

Sets an element of the Recipe linked list to favorite or not favorite.

Parameters

list	The address of the first element of the recipe linked list.
SetFavName	The name of the SelectFavorite Recipe.
isFav	Favorite or not.

4.13 Recipe.h

Go to the documentation of this file.

```
//
// Created by hejag on 23/11/2024.

#ifndef RECIPE_H
#define RECIPE_H
#include <stdbool.h>
#include "RecipeIngridient.h"
#include "Ingridient.h"
#include "debugmalloc.h"

typedef struct Recipe {
 char RecName[100];
 struct Recipe *NextNode;
 RecipeIngridient *RecIng;
 bool bIsFavorite;
} Recipe;
```

```

Recipe *CreateRecipe(char *Name, bool bIsFav);

void AssignIngridients(Recipe *Rec, RecipeIngridient *RecIngridient);

void AddRecipeTo(Recipe *list, Recipe *ToAdd);

void AddOrCreateRecipe(Recipe **list, Recipe *ToAdd);

void DeleteRecipe(Recipe **listPtr, Recipe *ToDelete);

Recipe *FindRecipeByName(Recipe *list, char *RecipeName);

void RecipePrint(Recipe *PRecipe);

void PrintRecipeByIng(Recipe* list);

void PrintFavRecipes(Recipe *list);

void SetFavorite(Recipe *list, char *SetFavName, bool isFav);

void PrintAllRecipes(Recipe *list);

void AddRecipeFromIO(Ingridient *IngList, Recipe *RecList);

bool AddRecipeFromIOIsFav(char *IsFav);

void DeleteAllRecipe(Recipe *list);

int RecipeListLen(Recipe *list);

char* BoolFavConverter(bool IsFav);

#endif //RECIPE_H
```

4.14 C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/RecipeIngridient.c File Reference

```
#include "RecipeIngridient.h"
#include "debugmalloc.h"
```

Functions

- RecipeIngridient ∗ CreateRecipeIngridient (int IngAmount, Ingridient ∗IngData)
- void AddRITo (RecipeIngridient ∗list, RecipeIngridient ∗ToAdd)
- void DeleteRI (RecipeIngridient ∗list, RecipeIngridient ∗ToDelete)
- RecipeIngridient ∗ FindRecipeIngridientByName (RecipeIngridient ∗list, char ∗IngName)
- void DeleteAllRecIng (RecipeIngridient ∗list)

4.14.1 Function Documentation

4.14.1.1 AddRITo()

```
void AddRITo (
            RecipeIngridient * list,
            RecipeIngridient * ToAdd)
```

Adds a RecipeIngridient struct to the RecipeIngridient linked list.

Parameters

list	The address of the first element of the RecipeIngridient linked list.
ToAdd	The Pointer of theRecipeIngridient that is added to the RecipeIngridient linked list.

4.14.1.2 CreateRecipeIngridient()

```
RecipeIngridient * CreateRecipeIngridient (
             int IngAmount,
             Ingridient * IngData)
```

Creates a RecipeIngrident struct.

Parameters

IngAmount	The amount of the ingridient specified in the ing data.
IngData	Points to the selected Ingridient in the Ingridient linked list.

Returns

Pointer RecipeIngridient struct.

4.14.1.3 DeleteAllRecIng()

```
void DeleteAllRecIng (
             RecipeIngridient * list)
```

Deletes all RecipeIngdients from the RecipeIngridient linked list.

Parameters

list	The address of the first element of the RecipeIngridient linked list.

4.14.1.4 DeleteRI()

```
void DeleteRI (
             RecipeIngridient * list,
             RecipeIngridient * ToDelete)
```

Deletes a RecipeIngridient from the RecipeIngridient linked list.

Parameters

list	The address of the first element of the RecipeIngridient linked list.
ToDelete	The address of the RecipeIngridient that will be deleted.

4.14.1.5 FindRecipeIngridientByName()

```
RecipeIngridient * FindRecipeIngridientByName (
             RecipeIngridient * list,
             char * IngName)
```

Finds a RecipeIngridient by Name.

Parameters

list	The address of the first element of the RecipeIngridient linked list.
IngName	The name of the searched Ingridient.

Returns

A RecipeIngridient pointer.

4.15 C:/Users/hejag/Documents/School Projects/ProgAlap01/NHF/NHFSave/RecipeIngridient.h File Reference

```
#include "Ingridient.h"
#include "debugmalloc.h"
```

Data Structures

- struct RecipeIngridient

Typedefs

- typedef struct RecipeIngridient RecipeIngridient

Functions

- RecipeIngridient ∗ CreateRecipeIngridient (int IngAmount, Ingridient ∗IngData)
- void AddRITo (RecipeIngridient ∗list, RecipeIngridient ∗ToAdd)
- void DeleteRI (RecipeIngridient ∗list, RecipeIngridient ∗ToDelete)
- RecipeIngridient ∗ FindRecipeIngridientByName (RecipeIngridient ∗list, char ∗IngName)
- void DeleteAllRecIng (RecipeIngridient ∗list)

4.15.1 Typedef Documentation

4.15.1.1 RecipeIngridient

```
typedef struct RecipeIngridient RecipeIngridient
```

A wrapper struct that connects the Ingridient stuct and the Recipe struct and it stores ingridient amount. The IngData points to the selected Ingridient. The NextNode is used to point to the next element of the RecipeIngridinent linked list.

4.15.2 Function Documentation

4.15.2.1 AddRITo()

```
void AddRITo (
            RecipeIngridient ∗ list,
            RecipeIngridient ∗ ToAdd)
```

Adds a RecipeIngridient struct to the RecipeIngridient linked list.

Parameters

list	The address of the first element of the RecipeIngridient linked list.
ToAdd	The Pointer of theRecipeIngridient that is added to the RecipeIngridient linked list.

4.15.2.2 CreateRecipeIngridient()

```
RecipeIngridient * CreateRecipeIngridient (
            int IngAmount,
            Ingridient * IngData)
```

Creates a RecipeIngrident struct.

Parameters

IngAmount	The amount of the ingridient specified in the ing data.
IngData	Points to the selected Ingridient in the Ingridient linked list.

Returns

Pointer RecipeIngridient struct.

4.15.2.3 DeleteAllRecIng()

```
void DeleteAllRecIng (
            RecipeIngridient * list)
```

Deletes all RecipeIngdients from the RecipeIngridient linked list.

Parameters

list	The address of the first element of the RecipeIngridient linked list.

4.15.2.4 DeleteRI()

```
void DeleteRI (
            RecipeIngridient * list,
            RecipeIngridient * ToDelete)
```

Deletes a RecipeIngridient from the RecipeIngridient linked list.

Parameters

list	The address of the first element of the RecipeIngridient linked list.
ToDelete	The address of the RecipeIngridient that will be deleted.

4.15.2.5 FindRecipeIngridientByName()

```
RecipeIngridient * FindRecipeIngridientByName (
            RecipeIngridient * list,
            char * IngName)
```

Finds a RecipeIngridient by Name.

Parameters

list	The address of the first element of the RecipeIngridient linked list.
IngName	The name of the searched Ingridient.

Returns

A RecipeIngridient pointer.

4.16 RecipeIngridient.h

Go to the documentation of this file.

```
//
// Created by hejag on 24/11/2024.

#ifndef RECIPEINGRIDIENT_H
#define RECIPEINGRIDIENT_H
#include "Ingridient.h"
#include "debugmalloc.h"
typedef struct RecipeIngridient {
 int IngAmount;
 Ingridient *IngData;
 struct RecipeIngridient *NextNode;
} RecipeIngridient;

RecipeIngridient *CreateRecipeIngridient(int IngAmount, Ingridient *IngData);

void AddRITo(RecipeIngridient *list, RecipeIngridient *ToAdd);

void DeleteRI(RecipeIngridient *list, RecipeIngridient *ToDelete);

RecipeIngridient *FindRecipeIngridientByName(RecipeIngridient *list, char *IngName);

void DeleteAllRecIng(RecipeIngridient *list);

//void PrintIngName(Ingridient *list, int IngridientIndex);

#endif //RECIPEINGRIDIENT_H
```

Index