My Project

Generated by Doxygen 1.9.8

Chapter 1

Class Index

1.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

Chapter 3

Class Documentation

3.1 Action Class Reference

< Class for storing actions

`#include <Action.hpp>`

Public Member Functions

- void Set_New_State (std::string newstate)
- void Set_New_Word (std::string newword)
- void Set_Move (char move)
- std::string Get_New_State () const
- std::string Get_New_Word () const
- char Get_Move () const

3.1.1 Detailed Description

< Class for storing actions

3.1.2 Member Function Documentation

3.1.2.1 Get_Move()

```
char Action::Get_Move ( ) const
```

Method to get move

Returns

Move

3.1.2.2 Get_New_State()

```
std::string Action::Get_New_State ( ) const
```

Method to get new state

Returns

New state

3.1.2.3 Get_New_Word()

```
std::string Action::Get_New_Word ( ) const
```

Method to get new word

Returns

New word

3.1.2.4 Set_Move()

```
void Action::Set_Move (
            char move )
```

Method for setting a move

Parameters

in	*move*	Move

3.1.2.5 Set_New_State()

```
void Action::Set_New_State (
            std::string newstate )
```

Method for setting a new state

Parameters

in	*newstate*	New state

3.1.2.6 Set_New_Word()

```
void Action::Set_New_Word (
            std::string newword )
```

Method for setting a new word

Parameters

in	*newword*	New word

The documentation for this class was generated from the following files:

- Action.hpp
- Action.cpp

3.2 Head Class Reference

< Class for implementing the machine head

`#include <Head.hpp>`

Public Member Functions

- void Set_Position (int position)
- void Set_State (std::string state)
- int Get_Position () const
- std::string Get_State () const

3.2.1 Detailed Description

< Class for implementing the machine head

3.2.2 Member Function Documentation

3.2.2.1 Get_Position()

```
int Head::Get_Position ( ) const
```

Method to get head position

Returns

Head position

3.2.2.2 Get_State()

```
std::string Head::Get_State ( ) const
```

Method to get head state

Returns

Head state

3.2.2.3 Set_Position()

```
void Head::Set_Position (
            int position )
```

Method for setting the head position

Parameters

in	*position*	New position

3.2.2.4 Set_State()

```
void Head::Set_State (
            std::string state )
```

Method for setting the head state

Parameters

in	*state*	New state

The documentation for this class was generated from the following files:

- Head.hpp
- Head.cpp

3.3 Rules Class Reference

```
#include <Rules.hpp>
```

Public Member Functions

- ~Rules ()
- std::string Get_State (int position) const
- std::string Get_Word (int position) const
- int Get_Amount_Of_States () const
- int Get_Amount_Of_Words () const
- Action * Get_Rule (int state, int word) const
- void New_Rule (char move, std::string newword, std::string newstate, std::string word, std::string state)
- bool Remove_Rule (std::string word, std::string state)
- std::string View () const

3.3.1 Constructor & Destructor Documentation

3.3.1.1 ~Rules()

```
Rules::~Rules ( )
```

Destructor for clearing the rules table

3.3.2 Member Function Documentation

3.3.2.1 Get_Amount_Of_States()

```
int Rules::Get_Amount_Of_States ( ) const
```

Method for getting the amount of different states

Returns

Amount of different states

3.3.2.2 Get_Amount_Of_Words()

```
int Rules::Get_Amount_Of_Words ( ) const
```

Method for getting the amount of different words

Returns

Amount of different states

3.3.2.3 Get_Rule()

```
Action * Rules::Get_Rule (
            int state,
            int word ) const
```

Method for obtaining actions based on specified conditions

Parameters

in	*state*	State number
in	*word*	Word number

Returns

Action

3.3.2.4 Get_State()

```
std::string Rules::Get_State (
            int position ) const
```

Method for obtaining the state of the number

Parameters

in	*position*	State number

Returns

State

3.3.2.5 Get_Word()

```
std::string Rules::Get_Word (
            int position ) const
```

Method for obtaining the word of the number

Parameters

in	*position*	Word number

Returns

Word

3.3.2.6 New_Rule()

```
void Rules::New_Rule (
            char move,
            std::string newword,
            std::string newstate,
            std::string word,
            std::string state )
```

Method for adding a new rule

Parameters

in	*move*	Symbol with move
in	*newword*	String with new word
in	*newstate*	String with new state
in	*word*	String with word
in	*state*	String with state

3.3.2.7 Remove_Rule()

```
bool Rules::Remove_Rule (
            std::string word,
            std::string state )
```

Method for deleting a rule

Parameters

in	*word*	Word of the rule to be deleted
in	*state*	State of the rule to be deleted

Returns

TRUE if such a rule exists else FALSE

3.3.2.8 View()

```
std::string Rules::View ( ) const
```

Method for viewing rules

Returns

Rules recorded in string

The documentation for this class was generated from the following files:

- Rules.hpp
- Rules.cpp

3.4 Tape Class Reference

Class for implementing the machine tape.

`#include <Tape.hpp>`

Public Member Functions

- void Add_Cell (std::string word)
- void Add_Cell (int position, std::string word)
- bool Remove_Cell ()
- bool Remove_Cell (int position)
- void Set_Cell (int position, std::string word)
- std::string Get_Cell (int position) const
- int Get_Size () const
- std::string View (Head head) const

3.4.1 Detailed Description

Class for implementing the machine tape.

3.4.2 Member Function Documentation

3.4.2.1 Add_Cell() [1/2]

```
void Tape::Add_Cell (
            int position,
            std::string word )
```

Method for adding a cell by position

Parameters

in	*position*	Cell position
in	*word*	Added cell

3.4.2.2 Add_Cell() [2/2]

```
void Tape::Add_Cell (
            std::string word )
```

Method for adding a cell to the end of a tape

Parameters

in	*word*	Added cell

3.4.2.3 Get_Cell()

```
std::string Tape::Get_Cell (
            int position ) const
```

Method for receiving cell

Parameters

in	*position*	Cell position

Returns

A cell

3.4.2.4 Get_Size()

```
int Tape::Get_Size ( ) const
```

Method for obtaining tape length

Returns

A tape lenght

3.4.2.5 Remove_Cell() [1/2]

```
bool Tape::Remove_Cell ( )
```

Method to delete the last cell

Returns

TRUE if such a cell exists else FALSE

3.4.2.6 Remove_Cell() [2/2]

```
bool Tape::Remove_Cell (
            int position )
```

Method to delete a cell by position

Parameters

in	*position*	Cell position

Returns

TRUE if such a cell exists else FALSE

3.4.2.7 Set_Cell()

```
void Tape::Set_Cell (
            int position,
            std::string word )
```

Method for cell replacement

Parameters

in	*position*	Cell position
in	*word*	New cell

3.4.2.8 View()

```
std::string Tape::View (
            Head head ) const
```

Method for viewing tape with head position

Parameters

in	*head*	Machine head \retutn Tape recorded in string

The documentation for this class was generated from the following files:

- Tape.hpp
- Tape.cpp

Chapter 4

File Documentation

4.1 Action.cpp File Reference

```
#include "Action.hpp"
```

4.2 Action.hpp File Reference

```
#include <string>
```

Classes

- class Action

  < *Class for storing actions*

4.3 Action.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002 #include <string>
00004 class Action {
00005 public:
00010     void Set_New_State(std::string newstate);
00015     void Set_New_Word(std::string newword);
00020     void Set_Move(char move);
00025     std::string Get_New_State() const;
00030     std::string Get_New_Word() const;
00035     char Get_Move() const;
00036 private:
00037     std::string new_state;
00038     std::string new_word;
00039     char move;
00040 };
```

4.4 Execution.cpp File Reference

```
#include <windows.h>
#include "Execution.hpp"
```

Macros

- #define LIMIT 100

Functions

- bool Turing_Mashine (Head &head, Tape &tape, Rules &rules)

4.4.1 Macro Definition Documentation

4.4.1.1 LIMIT

```
#define LIMIT 100
```

4.4.2 Function Documentation

4.4.2.1 Turing_Mashine()

```
bool Turing_Mashine (
            Head & head,
            Tape & tape,
            Rules & rules )
```

Method for performing Turing machine simulation

Parameters

in,out	*head*	Mashine head
in,out	*tape*	Machine tape
in	*rules*	Machine rules

Returns

FALSE if the number of steps is more than 100 else TRUE

4.5 Execution.hpp File Reference

```
#include "Tape.hpp"
#include "Rules.hpp"
```

Functions

- bool Turing_Mashine (Head &head, Tape &tape, Rules &rules)

4.5.1 Function Documentation

4.5.1.1 Turing_Mashine()

```
bool Turing_Mashine (
            Head & head,
            Tape & tape,
            Rules & rules )
```

Method for performing Turing machine simulation

Parameters

in,out	*head*	Mashine head
in,out	*tape*	Machine tape
in	*rules*	Machine rules

Returns

FALSE if the number of steps is more than 100 else TRUE

4.6 Execution.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002 #include "Tape.hpp"
00003 #include "Rules.hpp"
00011 bool Turing_Mashine(Head& head, Tape& tape, Rules& rules);
```

4.7 Head.cpp File Reference

```
#include "Head.hpp"
```

4.8 Head.hpp File Reference

```
#include <iostream>
```

Classes

- class Head

  < *Class for implementing the machine head*

4.9 Head.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002 #include <iostream>
00004 class Head {
00005 public:
00010     void Set_Position(int position);
00015     void Set_State(std::string state);
00020     int Get_Position() const;
00025     std::string Get_State() const;
00026 private:
00027     int position;
00028     std::string state;
00029 };
```

4.10 Interface.cpp File Reference

```
#include "Interface.hpp"
```

Functions

- bool Interface (Head &head, Tape &tape, Rules &rules, std::string choice)

4.10.1 Function Documentation

4.10.1.1 Interface()

```
bool Interface (
            Head & head,
            Tape & tape,
            Rules & rules,
            std::string choice )
```

Method for interaction between user and program

Parameters

in,out	*head*	Mashine head
in,out	*tape*	Machine tape
in,out	*rules*	Machine rules
in	*choice*	Action choice string

Returns

FALSE if such action do not exist, TRUE if Action choice string is Exit

4.11 Interface.hpp File Reference

```
#include "Loading_From_File.hpp"
```

```
#include "Execution.hpp"
```

Macros

- #define ADD_CELL "AddCell"
- #define GO "Go"
- #define SET_CELL "SetCell"
- #define REMOVE_CELL "RemoveCell"
- #define NEW_RULE "NewRule"
- #define REMOVE_RULE "RemoveRule"
- #define EDIT_HEAD "EditHead"
- #define VIEW_RULES "ViewRules"
- #define VIEW_TAPE "ViewTape"
- #define HELP "Help"
- #define EXIT "Exit"

Functions

- bool Interface (Head &head, Tape &tape, Rules &rules, std::string choice)

4.11.1 Macro Definition Documentation

4.11.1.1 ADD_CELL

```
#define ADD_CELL "AddCell"
```

4.11.1.2 EDIT_HEAD

```
#define EDIT_HEAD "EditHead"
```

4.11.1.3 EXIT

```
#define EXIT "Exit"
```

4.11.1.4 GO

```
#define GO "Go"
```

4.11.1.5 HELP

```
#define HELP "Help"
```

4.11.1.6 NEW_RULE

```
#define NEW_RULE "NewRule"
```

4.11.1.7 REMOVE_CELL

```
#define REMOVE_CELL "RemoveCell"
```

4.11.1.8 REMOVE_RULE

```
#define REMOVE_RULE "RemoveRule"
```

4.11.1.9 SET_CELL

```
#define SET_CELL "SetCell"
```

4.11.1.10 VIEW_RULES

```
#define VIEW_RULES "ViewRules"
```

4.11.1.11 VIEW_TAPE

```
#define VIEW_TAPE "ViewTape"
```

4.11.2 Function Documentation

4.11.2.1 Interface()

```
bool Interface (
            Head & head,
            Tape & tape,
            Rules & rules,
            std::string choice )
```

Method for interaction between user and program

Parameters

in,out	*head*	Mashine head
in,out	*tape*	Machine tape
in,out	*rules*	Machine rules
in	*choice*	Action choice string

Returns

FALSE if such action do not exist, TRUE if Action choice string is Exit

4.12 Interface.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002 #include "Loading_From_File.hpp"
00003 #include "Execution.hpp"
00004 #define ADD_CELL "AddCell"
00005 #define GO "Go"
00006 #define SET_CELL "SetCell"
00007 #define REMOVE_CELL "RemoveCell"
00008 #define NEW_RULE "NewRule"
00009 #define REMOVE_RULE "RemoveRule"
00010 #define EDIT_HEAD "EditHead"
00011 #define VIEW_RULES "ViewRules"
00012 #define VIEW_TAPE "ViewTape"
00013 #define HELP "Help"
00014 #define EXIT "Exit"
00015
00025 bool Interface(Head& head, Tape& tape, Rules& rules, std::string choice);
```

4.13 Lab1.cpp File Reference

```
#include "Interface.hpp"
```

Functions

- int main ()

4.13.1 Function Documentation

4.13.1.1 main()

```
int main ( )
```

4.14 Loading_From_File.cpp File Reference

```
#include "Loading_From_File.hpp"
```

Functions

- bool Edit_Head (std::string str, Head &head, Tape &tape)
- bool New_Rule (std::string str, Rules &rules)
- std::string Get_Word (std::string str, int &position)
- bool Loading_From_File (std::string file_name, Head &head, Tape &tape, Rules &rules)

4.14.1 Function Documentation

4.14.1.1 Edit_Head()

```
bool Edit_Head (
            std::string str,
            Head & head,
            Tape & tape )
```

Method for setting the properties of the head, the value of which is taken from the string

Parameters

in	*str*	A string containing the properties of the machine head
out	*head*	Machine head
out	*tape*	Machine tape \retutn FALSE if there was an error reading from the string else TRUE

4.14.1.2 Get_Word()

```
std::string Get_Word (
            std::string str,
            int & position )
```

Helper method to get a word from a string

Parameters

in	*str*	A string
in	*position*	Position of the first letter of a word \retutn Received word

4.14.1.3 Loading_From_File()

```
bool Loading_From_File (
            std::string file,
            Head & head,
            Tape & tape,
            Rules & rules )
```

Method for reading car data from a file

Parameters

in	*file*	File name
out	*head*	Machine head
out	*tape*	Machine tape
out	*rules*	Machine rules \retutn TRUE if this file exist else FALSE

4.14.1.4 New_Rule()

```
bool New_Rule (
            std::string str,
            Rules & rules )
```

Method for adding a new rule from a string

Parameters

in	*str*	String with rules
out	*rules*	Machine rules \retutn FALSE if there was an error reading from the string else TRUE

4.15 Loading_From_File.hpp File Reference

```
#include <fstream>
#include "Tape.hpp"
#include "Rules.hpp"
```

Functions

- bool Edit_Head (std::string str, Head &, Tape &)
- bool New_Rule (std::string str, Rules &rules)
- std::string Get_Word (std::string str, int &position)
- bool Loading_From_File (std::string file, Head &head, Tape &tape, Rules &rules)

4.15.1 Function Documentation

4.15.1.1 Edit_Head()

```
bool Edit_Head (
            std::string str,
            Head & head,
            Tape & tape )
```

Method for setting the properties of the head, the value of which is taken from the string

Parameters

in	*str*	A string containing the properties of the machine head
out	*head*	Machine head
out	*tape*	Machine tape \retutn FALSE if there was an error reading from the string else TRUE

4.15.1.2 Get_Word()

```
std::string Get_Word (
            std::string str,
            int & position )
```

Helper method to get a word from a string

Parameters

in	*str*	A string
in	*position*	Position of the first letter of a word \retutn Received word

4.15.1.3 Loading_From_File()

```
bool Loading_From_File (
            std::string file,
            Head & head,
            Tape & tape,
            Rules & rules )
```

Method for reading car data from a file

Parameters

in	*file*	File name
out	*head*	Machine head
out	*tape*	Machine tape
out	*rules*	Machine rules \retutn TRUE if this file exist else FALSE

4.15.1.4 New_Rule()

```
bool New_Rule (
            std::string str,
            Rules & rules )
```

Method for adding a new rule from a string

Parameters

in	*str*	String with rules
out	*rules*	Machine rules \retutn FALSE if there was an error reading from the string else TRUE

4.16 Loading_From_File.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002 #include <fstream>
00003 #include "Tape.hpp"
00004 #include "Rules.hpp"
00012 bool Edit_Head(std::string str, Head&, Tape&);
00019 bool New_Rule(std::string str, Rules& rules);
00026 std::string Get_Word(std::string str, int& position);
00035 bool Loading_From_File(std::string file, Head& head, Tape& tape, Rules& rules);
```

4.17 Rules.cpp File Reference

```
#include "Rules.hpp"
```

4.18 Rules.hpp File Reference

```
#include <vector>
#include "Action.hpp"
```

Classes

- class Rules

Macros

- #define R 'R'
- #define L 'L'
- #define N 'N'

  Rule class.

4.18.1 Macro Definition Documentation

4.18.1.1 L

```
#define L 'L'
```

4.18.1.2 N

```
#define N 'N'
```

Rule class.

4.18.1.3 R

```
#define R 'R'
```

4.19 Rules.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002 #include <vector>
00003 #include "Action.hpp"
00004 #define R 'R'
00005 #define L 'L'
00006 #define N 'N'
00008 class Rules {
00009 public:
00013     ~Rules();
00019     std::string Get_State(int position) const;
00025     std::string Get_Word(int position) const;
00030     int Get_Amount_Of_States() const;
00035     int Get_Amount_Of_Words() const;
00042     Action* Get_Rule(int state, int word) const;
00051     void New_Rule(char move, std::string newword, std::string newstate, std::string word, std::string
      state);
00058     bool Remove_Rule(std::string word, std::string state);
00063     std::string View() const;
00064 private:
00065     std::vector <std::string> states;
00066     std::vector <std::string> words;
00067     std::vector <std::vector <Action*>> table;
00068 };
```

4.20 Tape.cpp File Reference

```
#include "Tape.hpp"
```

4.21 Tape.hpp File Reference

```
#include <vector>
#include "Head.hpp"
```

Classes

- class Tape

  Class for implementing the machine tape.

Macros

- #define VOID ""

4.21.1 Macro Definition Documentation

4.21.1.1 VOID

```
#define VOID ""
```

4.22 Tape.hpp

Go to the documentation of this file.

```
00001 #pragma once
00002 #include <vector>
00003 #include "Head.hpp"
00004 #define VOID ""
00008 class Tape {
00009 public:
00014     void Add_Cell(std::string word);
00020     void Add_Cell(int position, std::string word);
00025     bool Remove_Cell();
00031     bool Remove_Cell(int position);
00037     void Set_Cell(int position, std::string word);
00043     std::string Get_Cell(int position) const;
00048     int Get_Size() const;
00054     std::string View(Head head) const;
00055 private:
00056     std::vector <std::string> cells;
00061     void Search(int& position);
00062 };
```

Index