

## Assignment 3

Generated by Doxygen 1.8.13



# Contents

<b>1</b>	<b>Class Index</b>	<b>1</b>
1.1	Class List . . . . .	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List . . . . .	3
<b>3</b>	<b>Class Documentation</b>	<b>5</b>
3.1	BoardT Class Reference . . . . .	5
3.1.1	Detailed Description . . . . .	6
3.1.2	Member Function Documentation . . . . .	6
3.1.2.1	get_deck() . . . . .	6
3.1.2.2	get_foundation() . . . . .	6
3.1.2.3	get_tab() . . . . .	6
3.1.2.4	get_waste() . . . . .	7
3.1.2.5	is_valid_deck_mv() . . . . .	7
3.1.2.6	is_valid_tab_mv() . . . . .	7
3.1.2.7	is_valid_waste_mv() . . . . .	8
3.1.2.8	is_win_state() . . . . .	8
3.1.2.9	tab_mv() . . . . .	8
3.1.2.10	valid_mv_exists() . . . . .	9
3.1.2.11	waste_mv() . . . . .	9
3.2	CardT Struct Reference . . . . .	9
3.2.1	Detailed Description . . . . .	9
3.3	Stack< T > Class Template Reference . . . . .	10
3.3.1	Detailed Description . . . . .	10
3.3.2	Member Function Documentation . . . . .	10
3.3.2.1	pop() . . . . .	10
3.3.2.2	push() . . . . .	10
3.3.2.3	size() . . . . .	11
3.3.2.4	top() . . . . .	11
3.3.2.5	toSeq() . . . . .	11

<b>4 File Documentation</b>	<b>13</b>
4.1 include/CardStack.h File Reference . . . . .	13
4.1.1 Detailed Description . . . . .	13
4.2 include/CardTypes.h File Reference . . . . .	13
4.2.1 Detailed Description . . . . .	14
4.3 include/GameBoard.h File Reference . . . . .	14
4.3.1 Detailed Description . . . . .	15
4.3.2 Typedef Documentation . . . . .	15
4.3.2.1 SeqCrdStckT . . . . .	15
4.4 include/Stack.h File Reference . . . . .	15
4.4.1 Detailed Description . . . . .	15
<b>Index</b>	<b>17</b>

# Chapter 1

## Class Index

### 1.1 Class List

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

<a href="#">BoardT</a>		
	<a href="#">BoardT</a> abstract data type representing a gameboard of forty thieves . . . . .	<a href="#">5</a>
<a href="#">CardT</a>		
	Describes a card . . . . .	<a href="#">9</a>
<a href="#">Stack&lt; T &gt;</a>		
	<a href="#">Stack</a> abstract data type . . . . .	<a href="#">10</a>



## Chapter 2

# File Index

### 2.1 File List

Here is a list of all documented files with brief descriptions:

include/ <a href="#">CardStack.h</a>	13
include/ <a href="#">CardTypes.h</a>	13
include/ <a href="#">GameBoard.h</a>	14
include/ <a href="#">Stack.h</a>	15





## Chapter 3

# Class Documentation

### 3.1 BoardT Class Reference

**BoardT** abstract data type representing a gameboard of forty thieves.

```
#include <GameBoard.h>
```

#### Public Member Functions

- **BoardT** ()  
*Empty constructor for BoardT.*
- **BoardT** (std::vector< **CardT** > deck)  
*Constructor for BoardT.*
- bool **is\_valid\_tab\_mv** (**CategoryT** c, **nat** n0, **nat** n1) const  
*Checks if a valid move exists for moving a card from tableau to another category or tableau.*
- bool **is\_valid\_waste\_mv** (**CategoryT** c, **nat** n) const  
*Checks if a valid move exists for moving a card from waste to another category or tableau.*
- bool **is\_valid\_deck\_mv** () const  
*Checks if a valid move exists for moving a card from deck to waste.*
- void **tab\_mv** (**CategoryT** c, **nat** n0, **nat** n1)  
*Moves a card from a tableau to another tableau or foundation.*
- void **waste\_mv** (**CategoryT** c, **nat** n)  
*Moves a card from waste to another category or tableau.*
- void **deck\_mv** ()  
*Moves a card from deck to waste.*
- **CardStackT** **get\_tab** (**nat** i) const  
*Accessor - Gets the stack of cards from the tableau at index i.*
- **CardStackT** **get\_foundation** (**nat** i) const  
*Accessor - Gets the stack of cards from the foundation at index i.*
- **CardStackT** **get\_deck** () const  
*Accessor - Gets the deck.*
- **CardStackT** **get\_waste** () const  
*Accessor - Gets the waste.*
- bool **valid\_mv\_exists** () const  
*Checks if a valid move exists.*
- bool **is\_win\_state** () const  
*Checks whether the game has been won.*

### 3.1.1 Detailed Description

`BoardT` abstract data type representing a gameboard of forty thieves.

### 3.1.2 Member Function Documentation

#### 3.1.2.1 `get_deck()`

```
CardStackT BoardT::get_deck ( ) const
```

Accessor - Gets the deck.

##### Returns

the stack of cards which is the deck

#### 3.1.2.2 `get_foundation()`

```
CardStackT BoardT::get_foundation (
    nat i ) const
```

Accessor - Gets the stack of cards from the foundation at index i.

##### Parameters

<code>i</code>	- the index of the foundation we want to get
----------------	--

##### Returns

the stack of cards from the foundation at index i

#### 3.1.2.3 `get_tab()`

```
CardStackT BoardT::get_tab (
    nat i ) const
```

Accessor - Gets the stack of cards from the tableau at index i.

##### Parameters

<code>i</code>	- the index of the tableau we want to get
----------------	---

**Returns**

a stack of cards from the tableau at index i

**3.1.2.4 get\_waste()**

```
CardStackT BoardT::get_waste ( ) const
```

Accessor - Gets the waste.

**Returns**

the stack of cards which is the waste

**3.1.2.5 is\_valid\_deck\_mv()**

```
bool BoardT::is_valid_deck_mv ( ) const
```

Checks if a valid move exists for moving a card from deck to waste.

**Returns**

boolean value representing whether a valid move from deck exists

**3.1.2.6 is\_valid\_tab\_mv()**

```
bool BoardT::is_valid_tab_mv (
    CategoryT c,
    nat n0,
    nat n1 ) const
```

Checks if a valid move exists for moving a card from tableau to another category or tableau.

**Parameters**

<i>c</i>	- the category we want to move a card from tableau to
<i>n0</i>	- the index of the tableau we want to move from
<i>n1</i>	- the index of the category we want to move to

**Returns**

boolean value representing whether a valid move from tableau exists

### 3.1.2.7 is\_valid\_waste\_mv()

```
bool BoardT::is_valid_waste_mv (
    CategoryT c,
    nat n ) const
```

Checks if a valid move exists for moving a card from waste to another category or tableau.

#### Parameters

<i>c</i>	- the category we want to move a card to
<i>n</i>	- the index of the category we want to move to

#### Returns

boolean value representing whether a valid move from waste exists

### 3.1.2.8 is\_win\_state()

```
bool BoardT::is_win_state ( ) const
```

Checks whether the game has been won.

#### Returns

a boolean value representing whether the game has been won

### 3.1.2.9 tab\_mv()

```
void BoardT::tab_mv (
    CategoryT c,
    nat n0,
    nat n1 )
```

Moves a card from a tableau to another tableau or foundation.

#### Parameters

<i>c</i>	- the category we want to move a card from tableau to
<i>n0</i>	- the index of the tableau we want to move from
<i>n1</i>	- the index of the category we want to move to

### 3.1.2.10 valid\_mv\_exists()

```
bool BoardT::valid_mv_exists ( ) const
```

Checks if a valid move exists.

#### Returns

a boolean value representing whether a move exists

### 3.1.2.11 waste\_mv()

```
void BoardT::waste_mv (
    CategoryT c,
    nat n )
```

Moves a card from waste to another category or tableau.

#### Parameters

<i>c</i>	- the category we want to move a card to
<i>n</i>	- the index of the category we want to move to

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

- include/[GameBoard.h](#)

## 3.2 CardT Struct Reference

Describes a card.

```
#include <CardTypes.h>
```

#### Public Attributes

- [SuitT](#) *s*
- [RankT](#) *r*

### 3.2.1 Detailed Description

Describes a card.

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

- include/[CardTypes.h](#)

### 3.3 Stack< T > Class Template Reference

Stack abstract data type.

```
#include <Stack.h>
```

#### Public Member Functions

- [Stack](#) ()  
*Empty constructor for a stack.*
- [Stack](#) (std::vector< T > s)  
*Empty constructor for a stack.*
- [Stack push](#) (T e)  
*Mutator - push element into stack.*
- [Stack pop](#) ()  
*Mutator - pop element off stack.*
- T [top](#) () const  
*Accessor - gets top of stack.*
- nat [size](#) () const  
*Gets size of stack.*
- std::vector< T > [toSeq](#) () const  
*Accessor - get stack in form of sequence.*

#### 3.3.1 Detailed Description

```
template<class T>
class Stack< T >
```

Stack abstract data type.

#### 3.3.2 Member Function Documentation

##### 3.3.2.1 pop()

```
template<class T>
Stack Stack< T >::pop ( )
```

Mutator - pop element off stack.

##### Returns

Ouput a new stack

##### 3.3.2.2 push()

```
template<class T>
Stack Stack< T >::push (
    T e )
```

Mutator - push element into stack.

**Parameters**

<code>e</code>	- Element to be pushed of type T
----------------	----------------------------------

**Returns**

Ouput a new stack

**3.3.2.3 size()**

```
template<class T>
nat Stack< T >::size ( ) const
```

Gets size of stack.

**Returns**

Ouput size of stack

**3.3.2.4 top()**

```
template<class T>
T Stack< T >::top ( ) const
```

Accessor - gets top of stack.

**Returns**

Ouput element at top of stack

**3.3.2.5 toSeq()**

```
template<class T>
std::vector<T> Stack< T >::toSeq ( ) const
```

Accessor - get stack in form of sequence.

**Returns**

[Stack](#) in form of sequence

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

- include/[Stack.h](#)





## Chapter 4

# File Documentation

### 4.1 include/CardStack.h File Reference

```
#include "CardTypes.h"  
#include "Stack.h"
```

#### Typedefs

- typedef [Stack](#)< [CardT](#) > [CardStackT](#)  
*Describes a stack of cards.*

#### 4.1.1 Detailed Description

##### Author

Leon So | macid: sol4

##### Date

2019-03-26

### 4.2 include/CardTypes.h File Reference

#### Classes

- struct [CardT](#)  
*Describes a card.*

## Macros

- `#define ACE 1`  
*RankT for an Ace.*
- `#define JACK 11`  
*RankT for an Jack.*
- `#define QUEEN 12`  
*RankT for a Queen.*
- `#define KING 13`  
*RankT for a King.*
- `#define TOTAL_CARDS 104`  
*Total number of cards.*

## Typedefs

- `typedef unsigned int nat`  
*Describes a natural number.*
- `typedef unsigned short int RankT`  
*Describes the rank of a card.*

## Enumerations

- `enum CategoryT { Tableau, Foundation, Deck, Waste }`  
*Describes the category of a stack of cards.*
- `enum SuitT { Heart, Diamond, Club, Spade }`  
*Describes the suit of a card.*

### 4.2.1 Detailed Description

#### Author

Leon So | macid: sol4

#### Date

2019-03-26

## 4.3 include/GameBoard.h File Reference

```
#include "CardTypes.h"
#include "CardStack.h"
#include <functional>
```

## Classes

- class `BoardT`  
*`BoardT` abstract data type representing a gameboard of forty thieves.*

## Typedefs

- typedef std::vector< [CardStackT](#) > [SeqCrdStckT](#)

### 4.3.1 Detailed Description

#### Author

Leon So | macid: sol4

#### Date

2019-03-26

### 4.3.2 Typedef Documentation

#### 4.3.2.1 SeqCrdStckT

```
typedef std::vector<CardStackT> SeqCrdStckT
```

Describes a sequence of card stacks

## 4.4 include/Stack.h File Reference

```
#include <vector>
```

## Classes

- class [Stack< T >](#)  
*[Stack](#) abstract data type.*

## Typedefs

- typedef unsigned int [nat](#)  
*Describes a natural number.*

### 4.4.1 Detailed Description

#### Author

Leon So | macid: sol4

#### Date

2019-03-26



# Index

BoardT, [5](#)  
    get\_deck, [6](#)  
    get\_foundation, [6](#)  
    get\_tab, [6](#)  
    get\_waste, [7](#)  
    is\_valid\_deck\_mv, [7](#)  
    is\_valid\_tab\_mv, [7](#)  
    is\_valid\_waste\_mv, [7](#)  
    is\_win\_state, [8](#)  
    tab\_mv, [8](#)  
    valid\_mv\_exists, [8](#)  
    waste\_mv, [9](#)

CardT, [9](#)

GameBoard.h  
    SeqCrdStckT, [15](#)

get\_deck  
    BoardT, [6](#)

get\_foundation  
    BoardT, [6](#)

get\_tab  
    BoardT, [6](#)

get\_waste  
    BoardT, [7](#)

include/CardStack.h, [13](#)  
include/CardTypes.h, [13](#)  
include/GameBoard.h, [14](#)  
include/Stack.h, [15](#)  
is\_valid\_deck\_mv  
    BoardT, [7](#)  
is\_valid\_tab\_mv  
    BoardT, [7](#)  
is\_valid\_waste\_mv  
    BoardT, [7](#)  
is\_win\_state  
    BoardT, [8](#)

pop  
    Stack, [10](#)

push  
    Stack, [10](#)

SeqCrdStckT  
    GameBoard.h, [15](#)

size  
    Stack, [11](#)

Stack  
    pop, [10](#)  
    push, [10](#)  
    size, [11](#)  
    toSeq, [11](#)  
    top, [11](#)  
Stack< T >, [10](#)

tab\_mv  
    BoardT, [8](#)

toSeq  
    Stack, [11](#)

top  
    Stack, [11](#)

valid\_mv\_exists  
    BoardT, [8](#)

waste\_mv  
    BoardT, [9](#)