

# Homework 01

03/21/10

Generated by Doxygen 1.6.3

Sun Mar 21 13:34:48 2010



# Contents

<b>1</b>	<b>Data Structure Index</b>	<b>1</b>
1.1	Data Structures . . . . .	1
<b>2</b>	<b>File Index</b>	<b>3</b>
2.1	File List . . . . .	3
<b>3</b>	<b>Data Structure Documentation</b>	<b>5</b>
3.1	PrimeStartEnd Struct Reference . . . . .	5
3.1.1	Detailed Description . . . . .	5
3.1.2	Field Documentation . . . . .	5
3.1.2.1	end . . . . .	5
3.1.2.2	start . . . . .	5
<b>4</b>	<b>File Documentation</b>	<b>7</b>
4.1	main.c File Reference . . . . .	7
4.1.1	Define Documentation . . . . .	7
4.1.1.1	N . . . . .	7
4.1.1.2	WORKTAG . . . . .	7
4.1.2	Function Documentation . . . . .	7
4.1.2.1	main . . . . .	7
4.2	prime.c File Reference . . . . .	8
4.2.1	Function Documentation . . . . .	8
4.2.1.1	do_work . . . . .	8
4.2.1.2	is_odd . . . . .	8
4.2.1.3	isPrime . . . . .	8
4.3	prime.h File Reference . . . . .	9
4.3.1	Function Documentation . . . . .	9
4.3.1.1	do_work . . . . .	9
4.3.1.2	is_odd . . . . .	9

4.3.1.3	<a href="#">isPrime</a>	9
---------	-------------------------	---

# Chapter 1

## Data Structure Index

### 1.1 Data Structures

Here are the data structures with brief descriptions:

[PrimeStartEnd](#) . . . . . 5



# Chapter 2

## File Index

### 2.1 File List

Here is a list of all files with brief descriptions:

<a href="#">main.c</a>	7
<a href="#">prime.c</a>	8
<a href="#">prime.h</a>	9





## Chapter 3

# Data Structure Documentation

### 3.1 PrimeStartEnd Struct Reference

```
#include <prime.h>
```

#### Data Fields

- [int start](#)
- [int end](#)

#### 3.1.1 Detailed Description

This holds the start and end of the prime calculations.

#### 3.1.2 Field Documentation

##### 3.1.2.1 [int end](#)

##### 3.1.2.2 [int start](#)

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

- [prime.h](#)



# Chapter 4

## File Documentation

### 4.1 main.c File Reference

```
#include <stdio.h>
#include "prime.h"
#include <math.h>
#include "mpi.h"
```

#### Defines

- #define [WORKTAG](#) 1
- #define [N](#) 100000000

#### Functions

- int [main](#) (int argc, char \*\*argv)

#### 4.1.1 Define Documentation

##### 4.1.1.1 #define N 100000000

##### 4.1.1.2 #define WORKTAG 1

#### 4.1.2 Function Documentation

##### 4.1.2.1 int main (int *argc*, char \*\* *argv*)

Find how many odd consecutive numbers there are between 1 and 100,000,000.

Splits the data evenly into ranks. It then processes and find how many odd consecutive numbers there is for a given rank.

## 4.2 prime.c File Reference

```
#include "prime.h"
```

### Functions

- int [isPrime](#) (int num)
- int [do\\_work](#) (struct [PrimeStartEnd](#) primeStartEnd)
- int [is\\_odd](#) (int num)

### 4.2.1 Function Documentation

#### 4.2.1.1 int do\_work (struct PrimeStartEnd *primeStartEnd*)

Does the actual calculations of the odd consecutive numbers.

##### Parameters

*primeStartEnd* A struct that holds where to start and stop the calculations of prime numbers.

##### Returns

How many odd consecutive numbers there are.

#### 4.2.1.2 int is\_odd (int *num*)

Checks if a given number is odd or not.

##### Parameters

*num* The number to check if it is a odd number.

##### Returns

True if it is a odd number.

#### 4.2.1.3 int isPrime (int *num*)

Tests for prime number.

##### Parameters

*num* The number to test if it is a prime number.

##### Returns

This will return true if it a prime false otherwise.

## 4.3 prime.h File Reference

```
#include <math.h>
```

### Data Structures

- struct [PrimeStartEnd](#)

### Functions

- int [isPrime](#) (int num)
- int [do\\_work](#) (struct [PrimeStartEnd](#) primeStartEnd)
- int [is\\_odd](#) (int num)

#### 4.3.1 Function Documentation

##### 4.3.1.1 int do\_work (struct PrimeStartEnd *primeStartEnd*)

Does the actual calculations of the odd consecutive numbers.

###### Parameters

*primeStartEnd* A struct that holds where to start and stop the calculations of prime numbers.

###### Returns

How many odd consecutive numbers there are.

##### 4.3.1.2 int is\_odd (int *num*)

Checks if a given number is odd or not.

###### Parameters

*num* The number to check if it is a odd number.

###### Returns

True if it is a odd number.

##### 4.3.1.3 int isPrime (int *num*)

Tests for prime number.

###### Parameters

*num* The number to test if it is a prime number.

###### Returns

This will return true if it a prime false otherwise.

# Index

- do\_work
  - prime.c, [8](#)
  - prime.h, [9](#)
- end
  - PrimeStartEnd, [5](#)
- is\_odd
  - prime.c, [8](#)
  - prime.h, [9](#)
- isPrime
  - prime.c, [8](#)
  - prime.h, [9](#)
- main
  - main.c, [7](#)
- main.c, [7](#)
  - main, [7](#)
  - N, [7](#)
  - WORKTAG, [7](#)
- N
  - main.c, [7](#)
- prime.c, [8](#)
  - do\_work, [8](#)
  - is\_odd, [8](#)
  - isPrime, [8](#)
- prime.h, [9](#)
  - do\_work, [9](#)
  - is\_odd, [9](#)
  - isPrime, [9](#)
- PrimeStartEnd, [5](#)
  - end, [5](#)
  - start, [5](#)
- start
  - PrimeStartEnd, [5](#)
- WORKTAG
  - main.c, [7](#)