

# 1 main — MIR Walkthrough

**Purpose:** TODO: Describe why this walkthrough exists

## 1.1 Source Context

```
fn main() {
    let tup:(i32, i32) = (42, 99);

    assert!(tup.0 != tup.1);
}
```

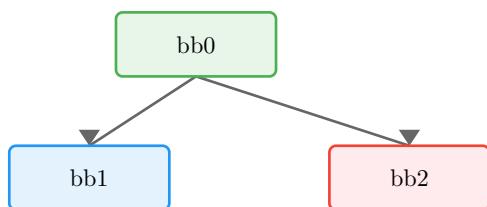
## 1.2 Function Overview

- **Function:** main
- **Basic blocks:** 3
- **Return type:** ()
- **Notable properties:**
  - Contains panic path
  - Has conditional branches

## 1.3 Locals

Local	Type	Notes
0	()	Return place
1	(i32, i32)	
2	bool	
3	i32	
4	i32	
5	!	

## 1.4 Control-Flow Overview



## 1.5 Basic Blocks

### 1.5.1 bb0 — entry

*Entry point of the function.*

MIR	Annotation
\_1 = Tuple(42, 99)	Construct aggregate
\_3 = \_1.0	Copy value
\_4 = \_1.1	Copy value
\_2 = move \_3 != move \_4	Not equal operation
→ switch(move \_2) \[0→bb2; else→bb1\]	Branch on move \_2

### 1.5.2 bb1 — return / success

*Normal return path.*

MIR	Annotation
<code>→ return</code>	Return from function

### 1.5.3 bb2 — panic path

*Panic/diverging path.*

MIR	Annotation
<code>→ \_5 = panic(\[16 bytes\])</code>	Call panic

## 1.6 Key Observations

TODO: Add bullet points summarizing what this MIR teaches

- 
- 

## 1.7 Takeaways

TODO: One or two sentences to generalize this example

