Testing Approaches

Testing Approaches

Testing

Testing Approaches

Analytics Vidhya

Testing Approaches

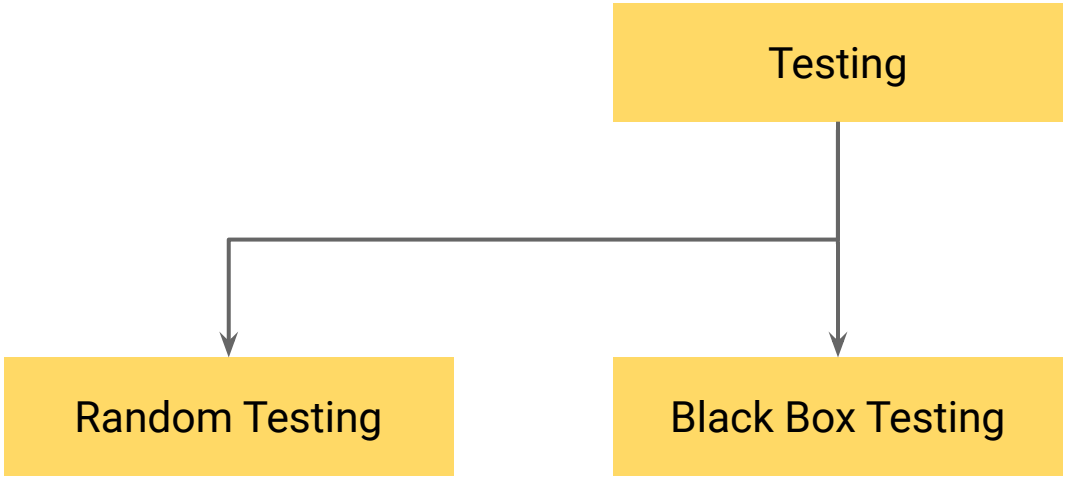
- Random Test cases

Testing Approaches

- Random Test cases
- More test cases, more surety

Analytics Vidhya

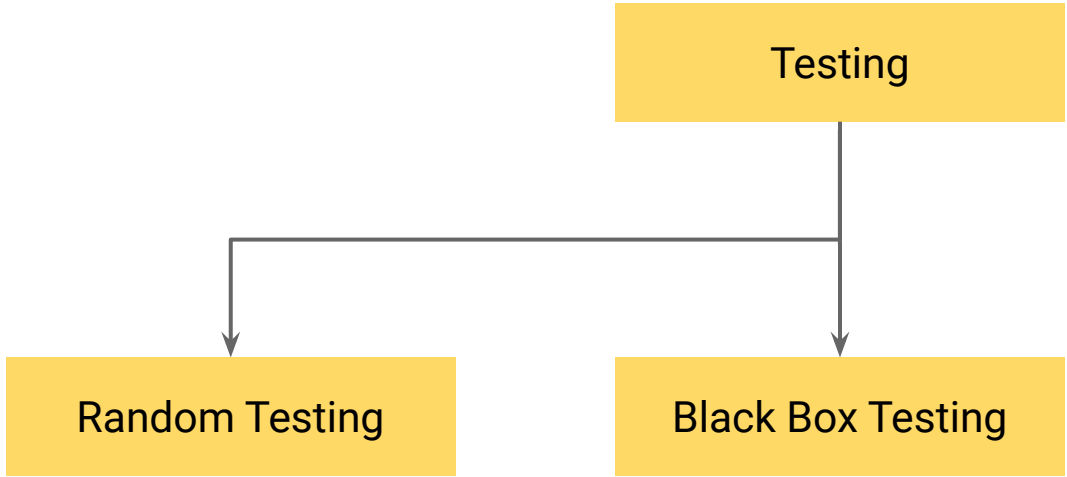
Testing Approaches



- Random Test cases
- More test cases, more surety

Analytics Vidhya

Testing Approaches

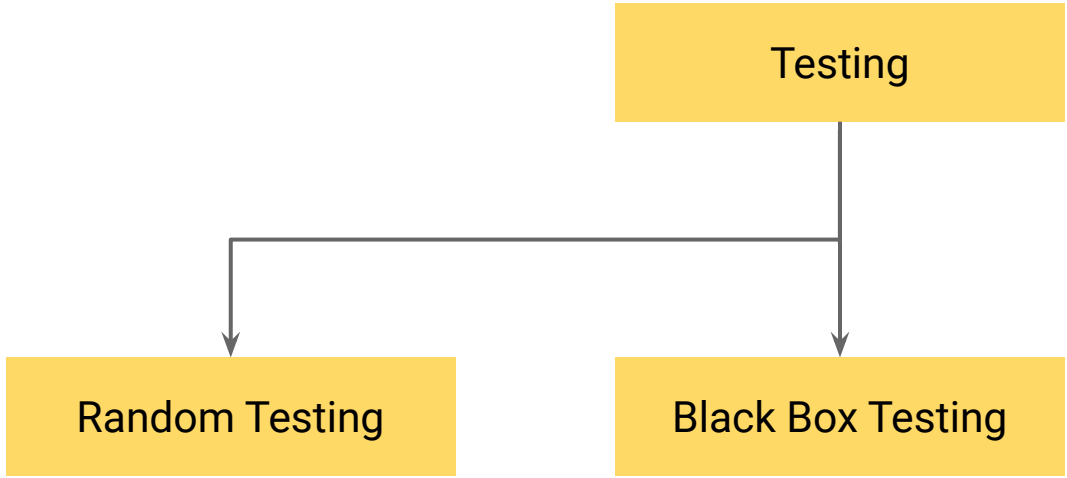


- Random Test cases
- More test cases, more surety

- Code is black Box

Analytics Vidhya

Testing Approaches

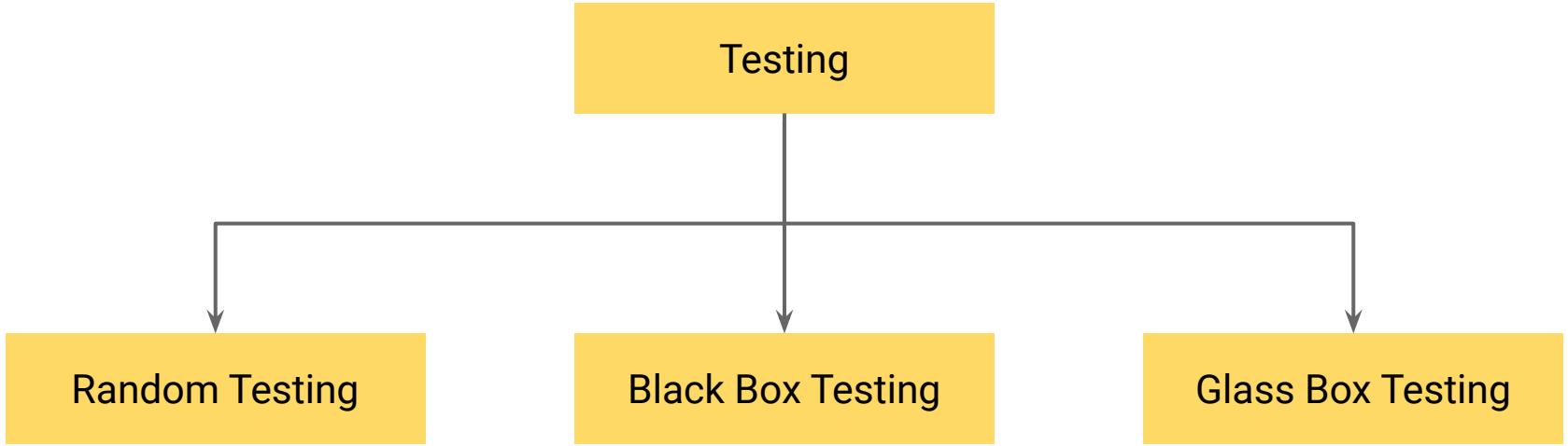


- Random Test cases
- More test cases, more surety

- Code is black Box
- Testing through docstrings

Analytics Vidhya

Testing Approaches

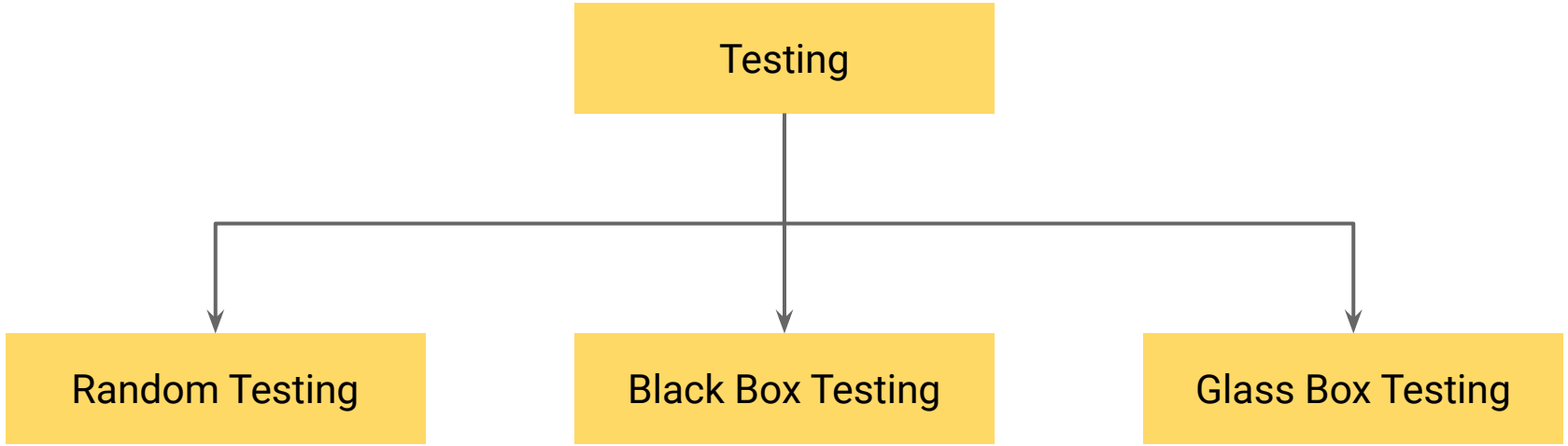


- Random Test cases
- More test cases, more surety

- Code is black Box
- Testing through docstrings

Analytics Vidhya

Testing Approaches



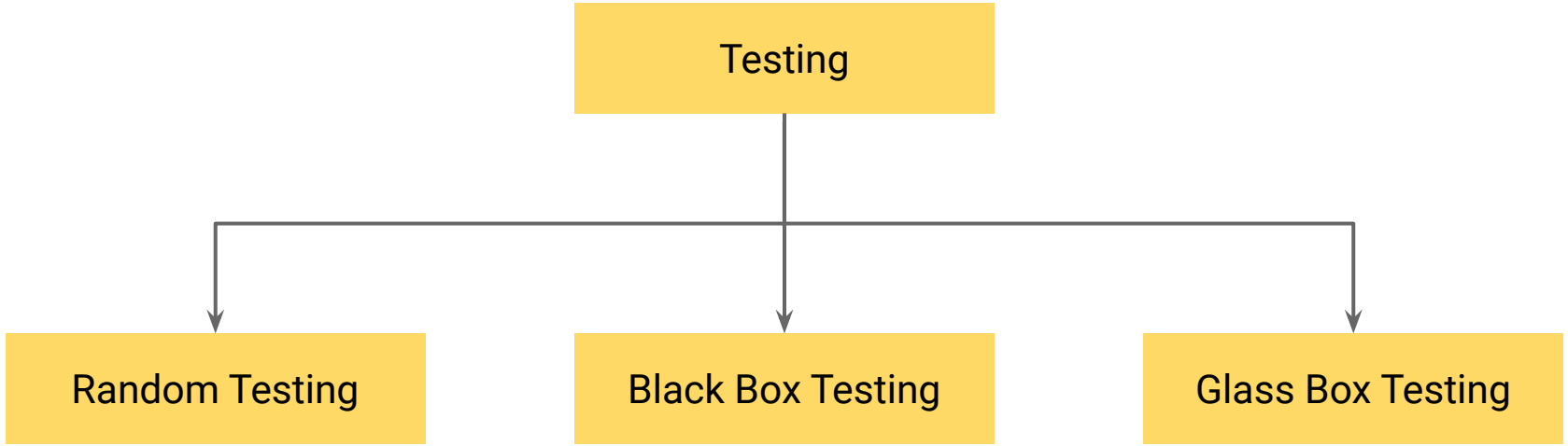
- Random Test cases
- More test cases, more surety

- Code is black Box
- Testing through docstrings

- Testing through code

Analytics Vidhya

Testing Approaches



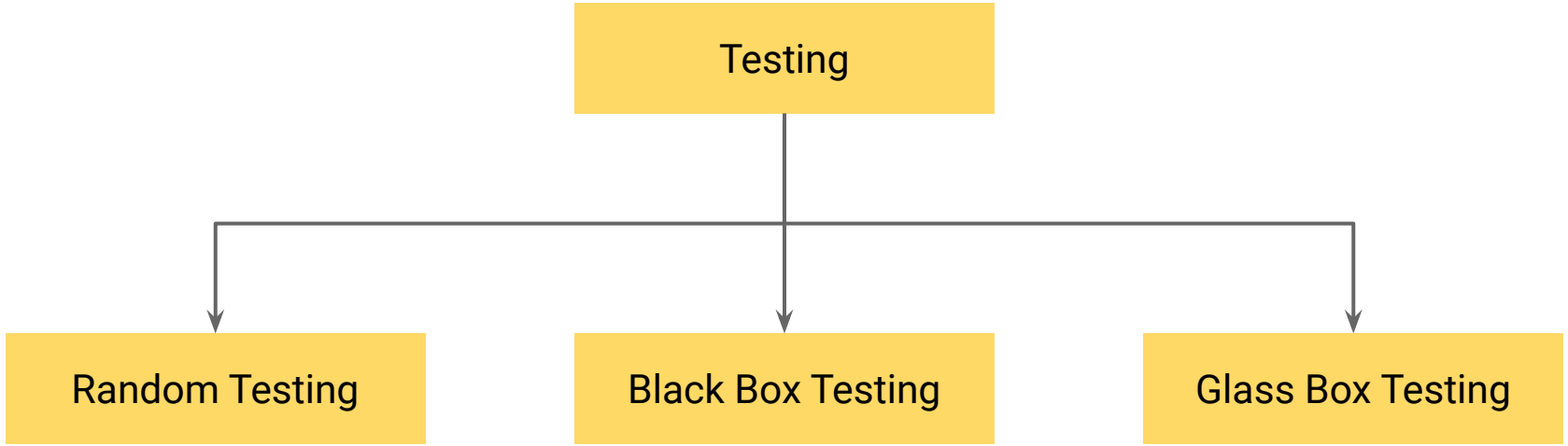
- Random Test cases
- More test cases, more surety

- Code is black Box
- Testing through docstrings

- Testing through code
- All paths are explored

Analytics Vidhya

Testing Approaches



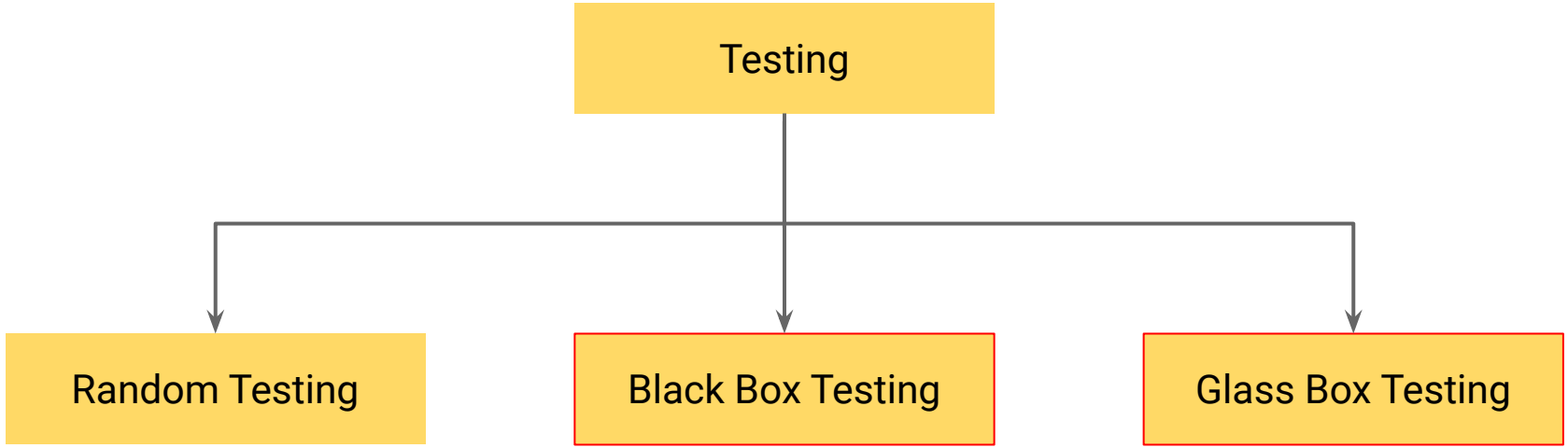
- Random Test cases
- More test cases, more surety

- Code is black Box
- Testing through docstrings

- Testing through code
- All paths are explored
- Exhaustive

Analytics Vidhya

Testing Approaches



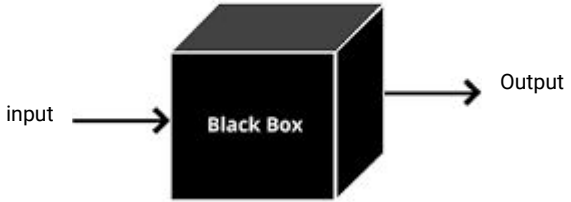
- Random Test cases
- More test cases, more surety

- Code is black Box
- Testing through docstrings

- Testing through code
- All paths are explored
- Exhaustive

Analytics Vidhya

Black Box Testing



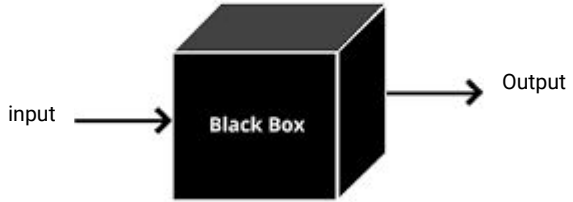
Analytics Vidhya

Black Box Testing

```
def absolute(value):
    """
    returns value if value is positive else returns negative value

    Parameters:
    value(float): value to calculate absolute of

    returns(float):absolute of value
    """
```



Analytics Vidhya

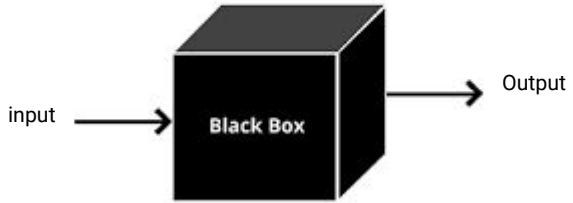
Black Box Testing

What are the natural boundaries in this logic?

```
def absolute(value):
    """
    returns value if value is positive else returns negative value

    Parameters:
    value(float): value to calculate absolute of

    returns(float):absolute of value
    """
```



Test Input	Expected Output

Analytics Vidhya

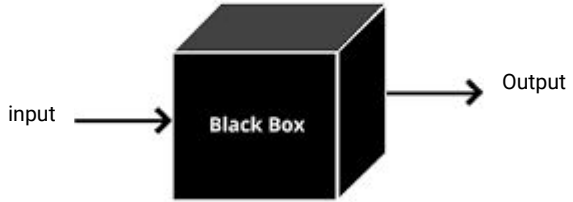
Black Box Testing

What are the natural boundaries in this logic?

```
def absolute(value):
    """
    returns value if value is positive else returns negative value

    Parameters:
    value(float): value to calculate absolute of

    returns(float):absolute of value
    """
```



Test Input	Expected Output
1	1

Analytics Vidhya

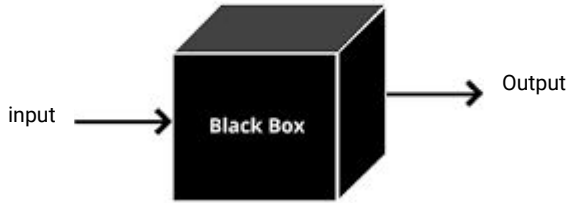
Black Box Testing

What are the natural boundaries in this logic?

```
def absolute(value):
    """
    returns value if value is positive else returns negative value

    Parameters:
    value(float): value to calculate absolute of

    returns(float):absolute of value
    """
```



Test Input	Expected Output
1	1
0	0

Analytics Vidhya

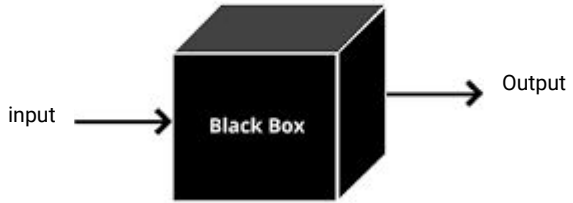
Black Box Testing

What are the natural boundaries in this logic?

```
def absolute(value):
    """
    returns value if value is positive else returns negative value

    Parameters:
    value(float): value to calculate absolute of

    returns(float):absolute of value
    """
```



Test Input	Expected Output
1	1
0	0
-1	1

Analytics Vidhya

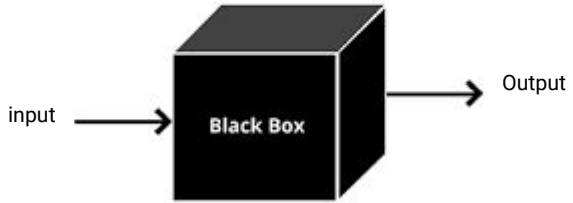
Black Box Testing

What are the natural boundaries in this logic?

```
def absolute(value):
    """
    returns value if value is positive else returns negative value

    Parameters:
    value(float): value to calculate absolute of

    returns(float):absolute of value
    """
```

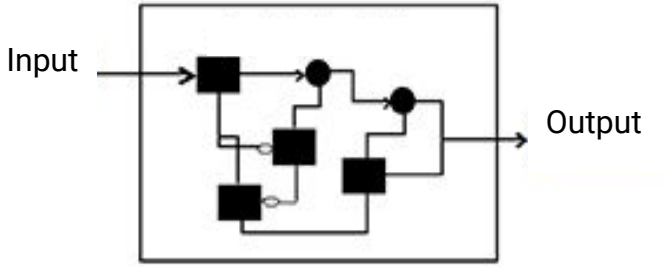


Test Input	Expected Output
1	1
0	0
-1	1

We never pass through through code

Analytics Vidhya

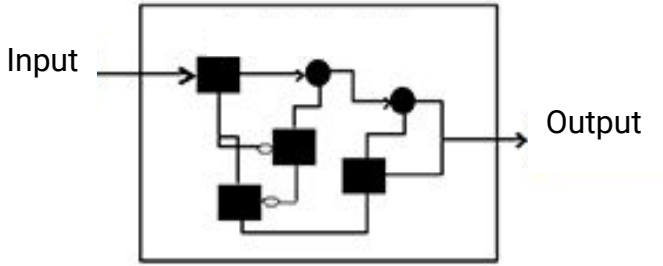
Glass Box Testing



Analytics Vidhya

Glass Box Testing

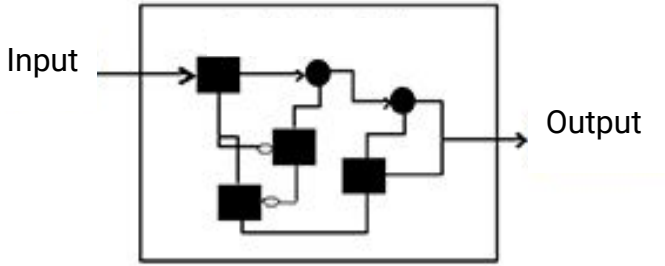
- Using code to design test cases



Analytics Vidhya

Glass Box Testing

- Using code to design test cases
- Check every potential path

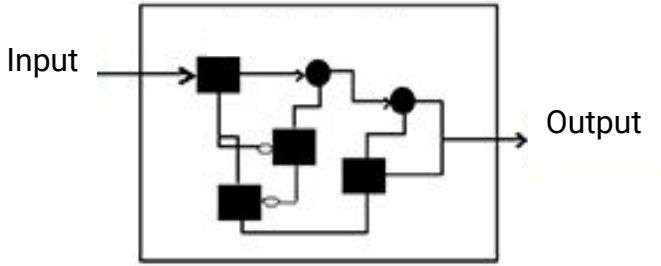


Analytics Vidhya

Glass Box Testing

- Using code to design test cases
- Check every potential path

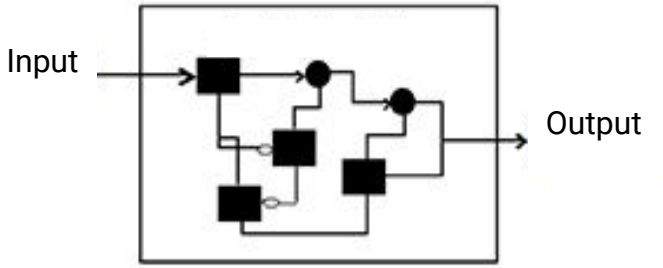
- Can be time taking and difficult to catch every path



Analytics Vidhya

Glass Box Testing

- Using code to design test cases
- Check every potential path
- Can be time taking and difficult to catch every path
- Guidelines for test cases

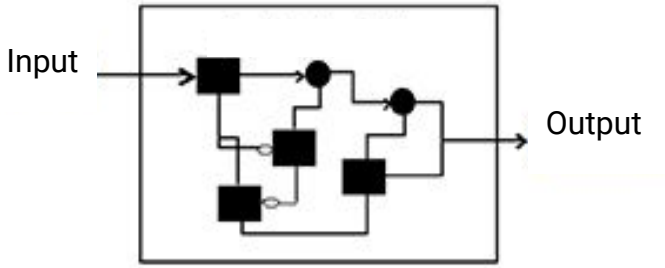


Analytics Vidhya

Glass Box Testing

- Using code to design test cases
- Check every potential path

- Can be time taking and difficult to catch every path



- Guidelines for test cases
  - branches: check for **if-else**
  - loops: check for **break, continue** etc..

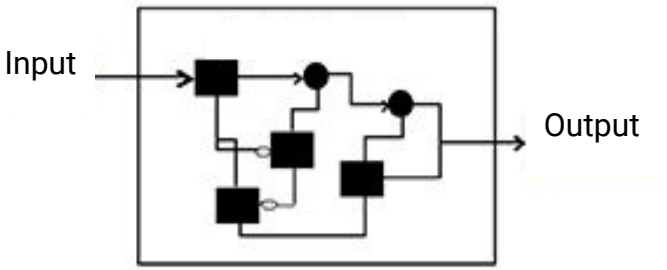
Analytics Vidhya

Glass Box Testing: Example

```python
def absolute(value):
    """Returns value if value>=0 and -value otherwise

    param:
    value(int): Integer for which the absolute value is to be calculated

    returns(int): Absolute value
    """
    if value < -1:
        return -value
    else:
        return value
```



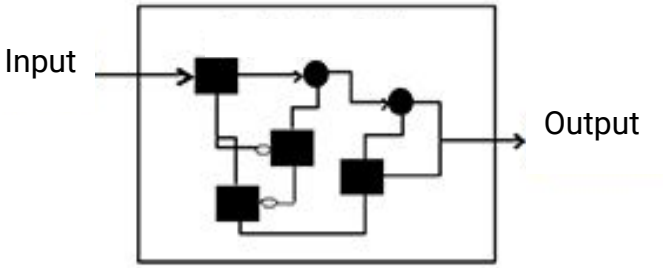
Analytics Vidhya

Glass Box Testing: Example

```python
def absolute(value):
    """Returns value if value>=0 and -value otherwise

    param:
    value(int): Integer for which the absolute value is to be calculated

    returns(int): Absolute value
    """
    if value < -1:
        return -value
    else:
        return value
```



Test Input	Expected Output	Actual Output
2	2	2
-2	2	2

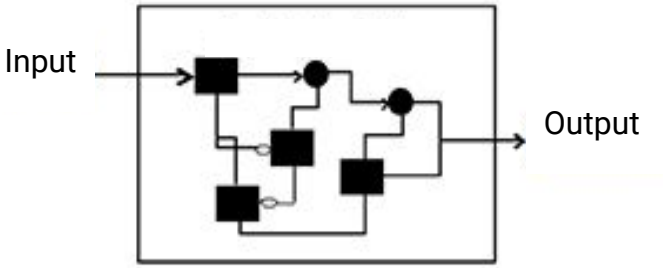
Analytics Vidhya

Glass Box Testing: Example

```python
def absolute(value):
    """Returns value if value>=0 and -value otherwise

    param:
    value(int): Integer for which the absolute value is to be calculated

    returns(int): Absolute value
    """
    if value < -1:
        return -value
    else:
        return value
```



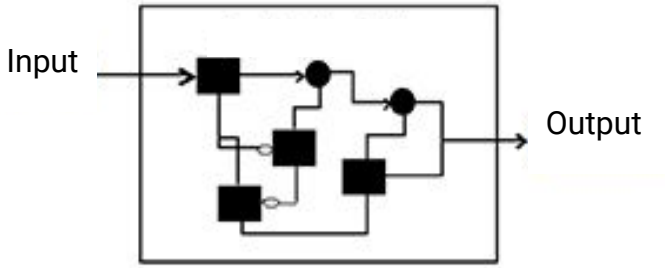
Test Input	Expected Output	Actual Output
2	2	2
-2	2	2

Table covers all the branches.

Analytics Vidhya

Glass Box Testing: Example

```
1  def absolute(value):
2      """Returns value if value>=0 and -value otherwise
3
4      param:
5      value(int): Integer for which the absolute value is to be calculated
6
7      returns(int): Absolute value
8      """
9      if value < -1:
10         return -value
11     else:
12         return value
```



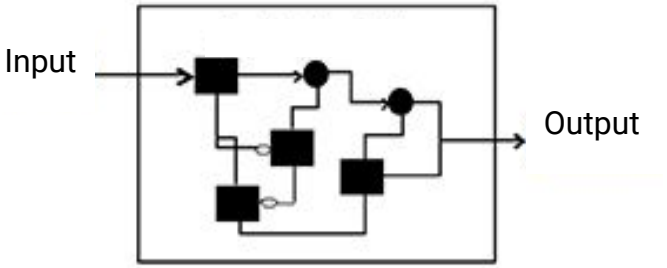
Test Input	Expected Output	Actual Output
2	2	2
-2	2	2
-1	-1	-1

Table covers all the branches.

Analytics Vidhya

Glass Box Testing: Example

```
1  def absolute(value):
2      """Returns value if value>=0 and -value otherwise
3
4      param:
5      value(int): Integer for which the absolute value is to be calculated
6
7      returns(int): Absolute value
8      """
9      if value < -1:
10         return -value
11     else:
12         return value
```



Test Input	Expected Output	Actual Output
2	2	2
-2	2	2
-1	1	-1

Table covers all the branches.

All partitions/boundaries still needs checking

Analytics Vidhya

Thank You