**Code of detecting dead code using python :**

import ast

def find\_dead\_code(source\_code):

tree = ast.parse(source\_code)

analyzer = DeadCodeAnalyzer()

analyzer.visit(tree)

return analyzer.dead\_code

class DeadCodeAnalyzer(ast.NodeVisitor):

def \_\_init\_\_(self):

self.dead\_code = set()

self.visited\_nodes = set()

def visit\_FunctionDef(self, node):

self.visited\_nodes.add(node)

self.generic\_visit(node)

def visit\_Call(self, node):

self.visited\_nodes.add(node)

self.generic\_visit(node)

def visit\_Assign(self, node):

self.visited\_nodes.add(node)

self.generic\_visit(node)

def visit\_Name(self, node):

if isinstance(node.ctx, ast.Store) and node not in self.visited\_nodes:

self.dead\_code.add(node.id)

self.generic\_visit(node)

# Example usage

source\_code = """

def main():

x = 5

y = 10

print(y)

if \_\_name\_\_ == "\_\_main\_\_":

main()

"""

dead\_code = find\_dead\_code(source\_code)

print("Dead code:", dead\_code)