Lab - 05

ACT : Advanced Compiler Techniques

AIM : Calculate Reach_In and Reach_Out sets using Pre-decessor, Generate and Kill sets.

Name : Ambalia Harshit
Roll no : MT001
Date : 29 Sept 2023

Question 01 : Calculate Reach_In and Reach_Out sets using Pre-decessor, Generate and Kill sets.

Input : Gen(), Kill() and Predecessor() set as matrix
Output : RCHin() and RCHout() set

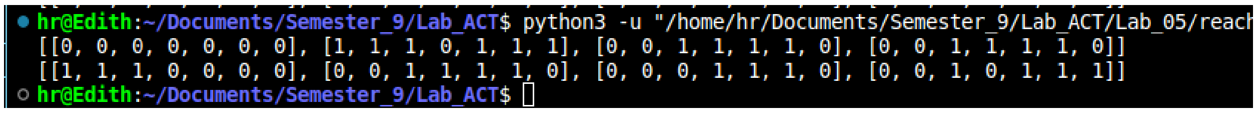
```python
def generate_gen_Set(number_of_blocks, number_of_variables,
generate_matrix):
   gen_set = []
   for i in range(number_of_blocks):
       temp = []
       for j in range(number_of_variables):
           if(generate_matrix[i][j]==1):
               temp.append(j+1)
       gen_set.append(temp)
   print(f'Gen Set : {gen_set}')

def generate_kill_Set(number_of_blocks, number_of_variables, kill_matrix):
   kill_set = []
   for i in range(number_of_blocks):
       temp = []
       for j in range(number_of_variables):
           if(kill_matrix[i][j]==1):
               temp.append(j+1)
```

```python
        kill_set.append(temp)
    print(f'Kill Set : {kill_set}')

def generate_predecessor_Set(number_of_blocks, predecessors_matrix):
    predecessors_set = []
    for i in range(number_of_blocks):
        temp = []
        for j in range(number_of_blocks):
            if(predecessors_matrix[i][j]==1):
                temp.append(j+1)
        predecessors_set.append(temp)
    return predecessors_set

def take_array_input(number_of_blocks, number_of_variables, text):
    print(text)
    arr = []
    for i in range(number_of_blocks):
        a = []
        print(f'Enter {i} row : ')
        for _ in range(number_of_variables):
            a.append(int(input()))
        arr.append(a)
    print(f'{text} : {arr}')
    return arr

def find_union(list_of_lists, number_of_variables):
    result = []
    if(list_of_lists):
        for i in range(len(list_of_lists[0])):
            max_value = max([list_of_lists[j][i] for j in
range(len(list_of_lists))])
            result.append(max_value)
    else:
        result = [0]*number_of_variables
    return result

def find_difference(reach_in_list, kill_list):
    result = []
    for i in range(len(reach_in_list)):
        if(reach_in_list[i]==1):
            result.append(reach_in_list[i] - kill_list[i])
        else:
            result.append(0)
    return result

def main():
    # generate_matrix = take_array_input(number_of_blocks,
number_of_variables, "Generate Matrix : ")
    generate_matrix = [
        [1, 1, 1, 0, 0, 0, 0],
        [0, 0, 0, 1, 1, 0, 0],
        [0, 0, 0, 0, 0, 1, 0],
        [0, 0, 0, 0, 0, 0, 1],
    ]
    # kill_matrix = take_array_input(number_of_blocks, number_of_variables,
"Kill Metrix : ")
    kill_matrix = [
        [0, 0, 0, 1, 1, 1, 1],
        [1, 1, 0, 0, 0, 0, 1],
        [0, 0, 1, 0, 0, 0, 0],
        [1, 0, 0, 1, 0, 0, 0],
    ]

    # predecessors_matrix = take_array_input(number_of_blocks,
number_of_blocks, "Predecessor Matrix : ")
    predecessors_matrix = [
        [0, 0, 0, 0],
        [1, 0, 0, 1],
        [0, 1, 0, 0],
        [0, 1, 1, 0],
    ]
    # number_of_blocks = int(input("Enter number of blocks : "))
    number_of_blocks = 4

    # number_of_variables = int(input("Enter number of definitions : "))
    number_of_variables = 7

    reach_in_set = [[0] * number_of_variables] * number_of_blocks
    reach_out_set = [[0] * number_of_variables] * number_of_blocks
    reach_in_set_previous = [[0] * number_of_variables] * number_of_blocks
```

```
   reach_out_set_previous = [[0] * number_of_variables] * number_of_blocks
   predecessors_set = generate_predecessor_Set(number_of_blocks,
predecessors_matrix)

   while(True):
       for i in range(number_of_blocks):
           temp_reachin = []
           for j in range(len(predecessors_set[i])):

temp_reachin.append(reach_out_set[predecessors_set[i][j]-1])
           reach_in_set[i] = find_union(temp_reachin, number_of_variables)

           temp_reachout = []
           temp_reachout.append(generate_matrix[i])
           temp_reachout.append(find_difference(reach_in_set[i],
kill_matrix[i]))
           reach_out_set[i] = find_union(temp_reachout,
number_of_variables)

       if((reach_in_set_previous == reach_in_set) and
(reach_out_set_previous == reach_out_set)):
           print(reach_in_set)
           print(reach_out_set)
           exit(0)
       else:
           reach_in_set_previous = reach_in_set
           reach_out_set_previous = reach_out_set

if __name__=="__main__":
   main()
```

Sample Output :

- This output is for given hardbinded input, But we can uncomment and take input from the user.



```
hr@Edith:~/Documents/Semester_9/Lab_ACT$ python3 -u "/home/hr/Documents/Semester_9/Lab_ACT/Lab_05/reach
[[0, 0, 0, 0, 0, 0, 0], [1, 1, 1, 0, 1, 1, 1], [0, 0, 1, 1, 1, 1, 0], [0, 0, 1, 1, 1, 1, 0]]
[[1, 1, 1, 0, 0, 0, 0], [0, 0, 1, 1, 1, 1, 0], [0, 0, 0, 1, 1, 1, 0], [0, 0, 1, 0, 1, 1, 1]]
hr@Edith:~/Documents/Semester_9/Lab_ACT$
```