Compiler Design(18CSC304J)

Experiment 14

CONSTRUCTION OF DAG IN CFG

Harsh Goel
RA1811003010185

Aim: Implement and verify construction of dag in cfg.

Language: C

Procedure:

1. Start.
2. Create a file or select the file for performing the operations on.
3. Start any python IDE and type the necessary code.
4. Run the code and perform the operations required.
5. Note the output and document it.
6. End.

Algorithm:

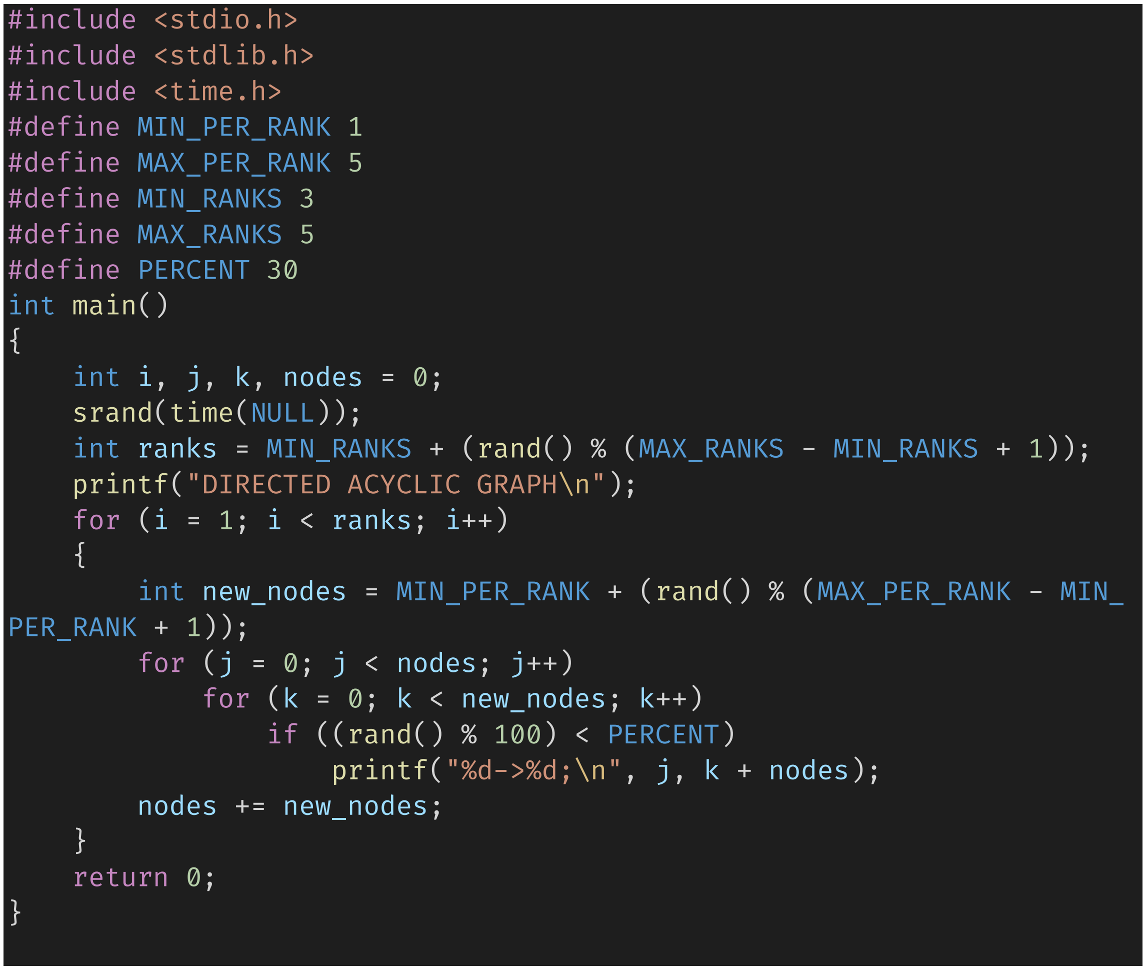
1. Each node contains a label. For leaves, the label is an identifier.Each node contains a list of attached identifiers to hold the computed values.

   It can have cases as:

   a. x:= y OP z
   b. x:= OP y
   c. x:= y

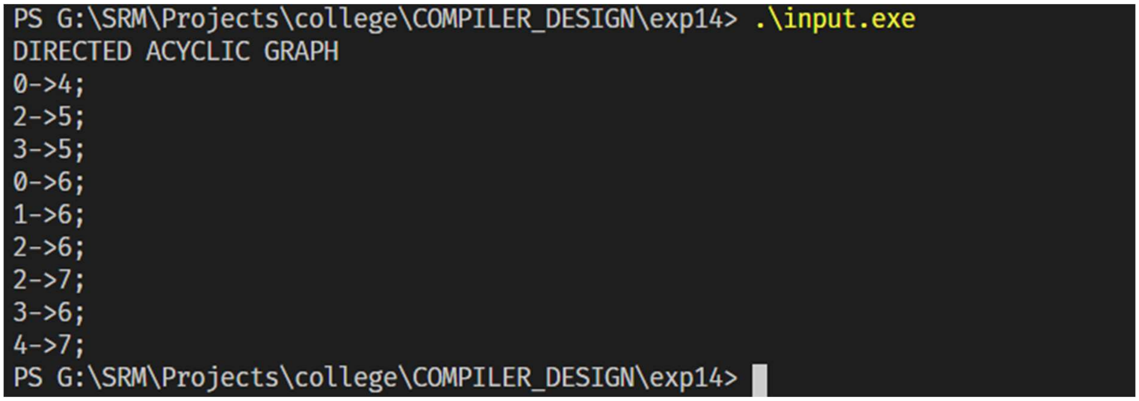
   Method:
2. Step 1:
   a. If y operand is undefined then create node(y).
   b. If z operand is undefined then for case(i) create node(z).
3. Step 2:
   a. For case(a),
      i. create node(OP) whose right child is node(z) and left child is node(y).
   b. For case(b)
      i. check whether there is node(OP) with one child node(y).
   c. For case(c)
      i. node n will be node(y).

Code Snippet:



```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define MIN_PER_RANK 1
#define MAX_PER_RANK 5
#define MIN_RANKS 3
#define MAX_RANKS 5
#define PERCENT 30
int main()
{
    int i, j, k, nodes = 0;
    srand(time(NULL));
    int ranks = MIN_RANKS + (rand() % (MAX_RANKS - MIN_RANKS + 1));
    printf("DIRECTED ACYCLIC GRAPH\n");
    for (i = 1; i < ranks; i++)
    {
        int new_nodes = MIN_PER_RANK + (rand() % (MAX_PER_RANK - MIN_PER_RANK + 1));
        for (j = 0; j < nodes; j++)
            for (k = 0; k < new_nodes; k++)
                if ((rand() % 100) < PERCENT)
                    printf("%d->%d;\n", j, k + nodes);
        nodes += new_nodes;
    }
    return 0;
}
```

Output Screenshots:



```
PS G:\SRM\Projects\college\COMPILER_DESIGN\exp14> .\input.exe
DIRECTED ACYCLIC GRAPH
0->4;
2->5;
3->5;
0->6;
1->6;
2->6;
2->7;
3->6;
4->7;
PS G:\SRM\Projects\college\COMPILER_DESIGN\exp14>
```

Result:

The code was successfully implemented and output was recorded.