# Practical 1 : DFS ( Sequential and Parallel Algorithms )

```
#include <iostream>
#include <vector>
#include <stack>
#include <omp.h>
#include <cstdlib>

using namespace std;

const int MAX = 100000;
vector<int> graph[MAX];
bool visited[MAX];

void dfs(int node) {
  stack<int> s;
  s.push(node);

  while (!s.empty()) {
    int curr_node = s.top();
    s.pop();

    if (!visited[curr_node]) {
      visited[curr_node] = true;
      cout << curr_node << " ";

      #pragma omp parallel for
      for (int i = 0; i < graph[curr_node].size(); i++) {
        int adj_node = graph[curr_node][i];
        if (!visited[adj_node]) {
          s.push(adj_node);
        }
      }
    }
  }
}

int main() {
  cout << "This is Atharva Pingale's code";
  cout << "\nPractical 1 : DFS ( Sequential and Parallel algorithms )";
  int n, m, start_node;
  double start_time, end_time;
  cout << "\n\nEnter number of nodes : ";
  cin >> n;
  cout << "Enter number of edges : ";
  cin >> m;
  cout << "Enter the starting node of the graph : ";
  cin >> start_node;

  for (int i = 0; i < m; i++) {
    int random_u = rand() % 999999;
    int random_v = rand() % 999999;

    graph[random_u].push_back(random_v);
    graph[random_v].push_back(random_u);
  }

  // Sequential Algorithm
  for (int i = 0; i < n; i++) {
    visited[i] = false;
  }

  start_time = omp_get_wtime();
  dfs(start_node);
  end_time = omp_get_wtime();
  double seq_time = end_time - start_time;

  // Parallel Algorithm
  for (int i = 0; i < n; i++) {
    visited[i] = false;
```

```
    }

    start_time = omp_get_wtime();
    dfs(start_node);
    end_time = omp_get_wtime();
    double parallel_time = end_time - start_time;
    cout << "\n\nSequential Algorithm Time: " << seq_time << " seconds\n";
    cout << "\nParallel Algorithm Time: " << parallel_time << " seconds\n";

    return 0;
}
```

# Output :

```
athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals (main)
$ g++ -fopenmp DFS.cpp -o DFS

athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals (main)
$ ./DFS
This is Atharva Pingale's code
Practical 1 : DFS ( Sequential and Parallel algorithms )

Enter number of nodes : 9999
Enter number of edges : 9999
Enter the starting node of the graph : 0
0 0

Sequential Algorithm Time: 0.0150001 seconds

Parallel Algorithm Time: 0 seconds

athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals (main)
$
```

Sequential Algo Time : 0.0150001 seconds
Parallel Algo Time : 0 seconds

```
athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals/Codes (main)
$ g++ -fopenmp DFS.cpp -o DFS

athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals/Codes (main)
$ ./DFS
This is Atharva Pingale's code
Practical 1 : DFS ( Sequential and Parallel algorithms )

Enter number of nodes : 10500
Enter number of edges : 15643
Enter the starting node of the graph : 0
0 20920 19463 5249 20600 20036 14817 4956 3095 18259 5927 20851 4940 1005 16051 18715
615 11222 2776 23657 16835 3573 5553 18277 29577 14474 80 14571 14802 2715 9087 17688
27494 32617 5678 7576 5234 30556 30974 22413 32399 30046 16386 23254 10700 2138 6204 6

Sequential Algorithm Time: 0.0340002 seconds

Parallel Algorithm Time: 0 seconds
```

Sequential Algo Time : 0.0340002 seconds
Parallel Algo Time : 0 seconds

```
athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals/Codes (main)
$ g++ -fopenmp DFS.cpp -o DFS

athar@LAPTOP-U0997R48 MINGW64 /d/GitHub/BE-8th-Semester/hpc_practicals/Codes (main)
$ ./DFS
This is Atharva Pingale's code
Practical 1 : DFS ( Sequential and Parallel algorithms )

Enter number of nodes : 13546
Enter number of edges : 17865
Enter the starting node of the graph : 0
0 20920 19463 5249 20600 20036 14817 4956 3095 18155 18259 5927 20851 4940 1005 16051
3 9473 19286 23668 9359 13685 29011 30771 10639 7009 20930 18863 20057 16759 23022 223
20911 16941 21724 31981 11087 3080 25306 32171 11869 27976 2431 23503 7777 20076 21821
0383 5385 28402 19704 10804 1037 13865 3333 27452 29936 25925 18455 5725 6032 4901 277
 18343 32713 22709 14564 13813 15397 21929 8724 31149 14619 30102 15967 21206 15203 19
45 522 20669 30642 29995 25389 1449 31028 4466 988 23586 17327 3709 8799 13670 7957 22
2 14252 10361 22074 32028 28168 22199 12233 10477 7487 28297 29465 19854 9990 6732 210
9958 24137 22852 11560 12537 18551 29087 28666 12070 15830 22013 24453 4056 5124 26415
4260 22359 15890 6729 13459 11150 7855 19634 485 13240 10180 14017 22202 3528 13069 17
6 32146 15002 1568 23820 31374 26461 21792 15953 22347 13470 3447 3854 31171 30026 302
8 16602 20629 15823 8420 13887 19437 2780 6851 5398 31688 16543 32373 5663 9597 19091
 630 10560 16832 3600 24668 13694 140 23998 1917 8031 9494 29694 17940 10791 26828 204
0 2623 16337 15457 12287 9396 22596 10210 5642 12796 10312 14834 20620 1334 28091 2458
520 23428 22003 26038 13964 12273 27268 18444 30999 4317 12623 19629 15600 19330 13048
3637 16649 1307 4232 15767 2353 19876 16587 14377 23412 23753 22249 2307 25518 31080 5
8373 26289 13707 9025 31112 23900 10959 9539 7388 18390 19550 13624 15658 23239 23725
8 6178 15784 1219 24290 29059 16157 10516 18055 5015 2452 21783 13767 23030 28279 139
 4220 30234 1748 11507 8582 10202 3625 15522 6700 30451 23136 252125 30574 15713 30935
025 31112 23900 10959 9539 7388 18390 19550 13624 15658 23239 23725 10619 10268 28041
24290 29059 16157 10516 18055 5015 2452 21783 13767 23030 28279 139 7898 14008 7262 70
1507 8582 10202 3625 15522 6700 30451 23136 25212 13296 7698 4094 315 19911 15238 1652
 20366 20227 15393 18219 1352 1185 26258 15857 26865 16833 13404 17662 2158 13864 2192
674 26545 8168 17399 18464 26245 23849 24075 13985 17192 7332 25920 28366 2885 24792 2
9 1077 26852 20583 17950 27046 7146 14328 2028 3455 6633 31595 19004 9801 19263 7520 9
357 24477 11473 6797 22519 18366 15758 50 2734 25624 4432 7985 6985 30872 19866 12949
0 13793 2955 30574 15713 30935 4248 26533 16465 22453 13726 21015 1131 10640 4970 1524
5 30574 15713 30935 4248 26533 16465 22453 13726 21015 1131 10640 4970 15246 28617 237
 10619 10268 28041 5076 2068 14379 24846 12529 16687 22974 16290 31015 27886 25308 232
 7898 14008 7262 7033 21218 26371 23705 3339 25052 3316 1724 22821 19587 30098 26658 1
15 19911 15238 16526 31232 7178 6352 5491 24597 2516 27791 29524 20421 90 1164 25396 3
662 2158 13864 21922 437 32134 23251 10724 16962 12053 17305 11325 12646 22710 30389 2
 28366 2885 24792 25533 6695 25219 20800 22894 30608 13216 10957 8519 13194 26807 2294
4 9801 19263 7520 9607 17535 22216 25956 7079 685 13589 2081 14355 711 20830 24593 145
 30872 19866 12949 5055 10168 9480 26818 12673 28296 23816 24830 3079 23179 26187 2330
 31101 15006 18286 2739 31289 25699 10087 31400 5829 6270 6188 30227 9576 6838 17059 1
7866 9383 4467 12694 4060 18331 8821 21091 20597 4844 31068 21991 12138 4343 9578 1551
4 28 21543 29692 6929 25603 9190 30461 17349 15827 5372 22425 17752 4399 10719 11831 1
6 29108 23486 6602 30807 1804 29865 14612 8310 1866 20605 22001 30411 28103 6358 19069
 7389 30231 24115 12135 7740 8546 20906 4908 22451 17383 4151 20779 8368 22935 26639 2
245 23088 24966 12088 32120 31057 4388 19695 9573 2439 11659 9609 30405 4897 11734 629
13 28796 9096 1006 6367 30776 3648 11883 19723 22990 2282 21987 6209 14820 30432 6741
1453 13071 13773 32072 32129 2877 13398 24011 28746 1498 26033 22619 19008 2880 17242
57 25484 23851 1699 19035 24442 5707 21496 9308 29439 13334 19826 11939 8561 16634 163
42 564 24961 14199 23432 12674 9764 10437 16803 22576 13462 27981 4047 29091 2697 2752
1 19384 5785 13233 194 31208 3247 23733 10294 4931 22197 13509 30300 24557 19686 13138
69 7937 9033 14672 27460 21763 30188 18960 16044 17753 6776 3493 14141 28026 20124 239
 9864 10484 25310 8689 8419 10901 22432 19225 24630 19326 4948 19206 32273 11810 25772
65 14781 10778 4439 7903 7377 12181 25264 8891 25538 193 23195 15628 8696 25652 872 15
 1694 27931 21060 14855 16001 32670 7849 14979 23973 9938 763 31321 10522 1576 30538 2
774 7877 18849 2700 4826 31942 27519 17225 28873 26684 14059 17657 4953 1932 12602 885
50 23452 17560 9365 5771 1052 6995 4694 20633 3241 7357 3164 20136 4680 19248 26076 86
900 32591 26841 22015 16088 8879 11858 8252 15145 25990 2403 32677 4410 21115 8911 326
2 1008 31736 27978 17372 19651 803 6250 10024 16562 23873 31719 10892 27242 19886 1619
386 23254 10700 2138 6204 2425 635 16612 12316 3035 26067 22187 21155 26720 26291 1492

Sequential Algorithm Time: 0.889 seconds

Parallel Algorithm Time: 0 seconds
```

Number of Nodes : 13546
Number of Edges : 17865
Start node : 0
Sequential Algo Time : 0.889 seconds
Parallel Algo Time : 0 seconds