National Institute of Technology Karnataka Surathkal,
Mangalore - 575025

DEPARTMENT OF INFORMATION TECHNOLOGY
DEPARTMENT OF CIVIL ENGINEERING

LAB ASSIGNMENT 1

Submitted for Parallel Computing (IT301) By

Gaurav Chaurasia 181CV155

To

Dr. Geetha V *Dept of IT, NITK Surathkal*

Code link
github

Problem - 1

Generating random values and setting up the temp array

```
69
70  int main() {
71      // vector<vector<vector<int>>> tem(1000, vector<vector<int>>(
72      // setting up the array with random data between 1 to 100;
73      for (int i = 0; i < CITY; i++) {
74          for (int j = 0; j < HOUR; j++) {
75              for (int k = 0; k < READINGS; k++) {
76                  tem[i][j][k] = 1 + (rand() % 100);
77              }
78          }
79      }
80
81      for (int i = 0; i < CITY; i++) {
82          for (int j = 0; j < HOUR; j++) {
83              for (int k = 0; k < READINGS; k++) {
84                  mean_tem[i][j] += tem[i][j][k];
85              }
86              mean_tem[i][j] /= READINGS;
87          }
88      }
89
90      min_temperature(7);
91      max_temperature(7);
92      avg_temperature(7);
93      return 0;
94  }
```

Function to find max temp

```
46
47  void max_temperature(int count=CITY) {
48      cout << "#####\tCITY NO-----MAX TEMPERATURE\t#####" << endl;
49      for (int i = 0; i < min(count, CITY); i++) {
50          int max_tem = INT_MIN;
51          for (int j = 0; j < HOUR; j++) {
52              max_tem = max(max_tem, mean_tem[i][j]);
53          }
54          cout << "[ 'CITY NO': " << i+1 << ", 'MAX TEMPERATURE': " << max_tem << " ]" << endl;
55      }
56  }
57
```

Function to find avg temp

```
57
58  void avg_temperature(int count=CITY) {
59      cout << "#####\tCITY NO-----AVG TEMPERATURE\t#####" << endl;
60      for (int i = 0; i < min(count, CITY); i++) {
61          float avg_tem = 0;
62          for (int j = 0; j < HOUR; j++) {
63              avg_tem += mean_tem[i][j];
64          }
65          avg_tem /= HOUR;
66          cout << "[ 'CITY NO': " << i+1 << ", 'AVG TEMPERATURE': " << avg_tem << " ]" << endl;
67      }
68  }
```

Function to find max temp

```
35
36  void min_temperature(int count=CITY) {
37      cout << "#####\tCITY NO-----MIN TEMPERATURE\t#####" << endl;
38      for (int i = 0; i < min(count, CITY); i++) {
39          int min_tem = INT_MAX;
40          for (int j = 0; j < HOUR; j++) {
41              min_tem = min(min_tem, mean_tem[i][j]);
42          }
43          cout << "[ 'CITY NO': " << i+1 << ", 'MIN TEMPERATURE': " << min_tem << " ]" << endl;
44      }
45  }
46
```

Function to print temp info

```
22  // prints the temperature information
23  void prind_temperature_info() {
24      for (int i = 0; i < CITY; i++) {
25          for (int j = 0; j < HOUR; j++) {
26              cout << "[ ";
27              for (int k = 0; k < READINGS; k++) {
28                  cout << tem[i][j][k] << " ";
29              }
30              cout << "]" << endl;
31          }
32          cout << endl;
33      }
34  }
```

Function to generate random nums

```
11
12  // generate count number of random numers between 1 to 100
13  void genrate_random_nums(int count) {
14      int result;
15      for (int i = 0; i < count; i++) {
16          result = 1 + (rand() % 100);
17          cout << result << " ";
18      }
19      cout << endl;
20  }
21
```

Printing temp info of first 7 cities

```
> g++ temperature.cpp
> ./a.out
#####   CITY NO-----MIN TEMPERATURE     #####
[ 'CITY NO': 1, 'MIN TEMPERATURE': 31 ]
[ 'CITY NO': 2, 'MIN TEMPERATURE': 31 ]
[ 'CITY NO': 3, 'MIN TEMPERATURE': 37 ]
[ 'CITY NO': 4, 'MIN TEMPERATURE': 28 ]
[ 'CITY NO': 5, 'MIN TEMPERATURE': 38 ]
[ 'CITY NO': 6, 'MIN TEMPERATURE': 27 ]
[ 'CITY NO': 7, 'MIN TEMPERATURE': 34 ]
#####   CITY NO-----MAX TEMPERATURE     #####
[ 'CITY NO': 1, 'MAX TEMPERATURE': 65 ]
[ 'CITY NO': 2, 'MAX TEMPERATURE': 74 ]
[ 'CITY NO': 3, 'MAX TEMPERATURE': 67 ]
[ 'CITY NO': 4, 'MAX TEMPERATURE': 74 ]
[ 'CITY NO': 5, 'MAX TEMPERATURE': 74 ]
[ 'CITY NO': 6, 'MAX TEMPERATURE': 66 ]
[ 'CITY NO': 7, 'MAX TEMPERATURE': 70 ]
#####   CITY NO-----AVG TEMPERATURE     #####
[ 'CITY NO': 1, 'AVG TEMPERATURE': 50.0417 ]
[ 'CITY NO': 2, 'AVG TEMPERATURE': 49.2083 ]
[ 'CITY NO': 3, 'AVG TEMPERATURE': 51.1667 ]
[ 'CITY NO': 4, 'AVG TEMPERATURE': 51.5 ]
[ 'CITY NO': 5, 'AVG TEMPERATURE': 53.5 ]
[ 'CITY NO': 6, 'AVG TEMPERATURE': 50.6667 ]
[ 'CITY NO': 7, 'AVG TEMPERATURE': 50.375 ]

/mnt/d/SEM/SEM 06/IT301-Parallel Computing/IT301-LAB/LAB-1                    11:03:18 AM
```

Code for Problem 1

```
#include <iostream>
#include <climits>
#include <vector>
using namespace std;

#define CITY 1000
#define HOUR 24
#define READINGS 10
int tem[CITY][HOUR][READINGS];
int mean_tem[CITY][HOUR];

// generate count number of random numers between 1 to 100
void genrate_random_nums(int count) {
    int result;
    for (int i = 0; i < count; i++) {
        result = 1 + (rand() % 100);
        cout << result << " ";
    }
    cout << endl;
}

// prints the temperature information
void prind_temperature_info() {
    for (int i = 0; i < CITY; i++) {
        for (int j = 0; j < HOUR; j++) {
            cout << "[ ";
            for (int k = 0; k < READINGS; k++) {
                cout << tem[i][j][k] << " ";
            }
            cout << "]" << endl;
        }
        cout << endl;
    }
}
```

```
void min_temperature(int count=CITY) {
    cout << "#####\tCITY NO-----MIN TEMPERATURE\t#####" <<
endl;
    for (int i = 0; i < min(count, CITY); i++) {
        int min_tem = INT_MAX;
        for (int j = 0; j < HOUR; j++) {
            min_tem = min(min_tem, mean_tem[i][j]);
        }
        cout << "[ 'CITY NO': " << i+1 << ", 'MIN
TEMPERATURE': " << min_tem << " ]" << endl;
    }
}

void max_temperature(int count=CITY) {
    cout << "#####\tCITY NO-----MAX TEMPERATURE\t#####" <<
endl;
    for (int i = 0; i < min(count, CITY); i++) {
        int max_tem = INT_MIN;
        for (int j = 0; j < HOUR; j++) {
            max_tem = max(max_tem, mean_tem[i][j]);
        }
        cout << "[ 'CITY NO': " << i+1 << ", 'MAX
TEMPERATURE': " << max_tem << " ]" << endl;
    }
}

void avg_temperature(int count=CITY) {
    cout << "#####\tCITY NO-----AVG TEMPERATURE\t#####" <<
endl;
    for (int i = 0; i < min(count, CITY); i++) {
        float avg_tem = 0;
        for (int j = 0; j < HOUR; j++) {
            avg_tem += mean_tem[i][j];
        }
        avg_tem /= HOUR;
        cout << "[ 'CITY NO': " << i+1 << ", 'AVG
TEMPERATURE': " << avg_tem << " ]" << endl;
```

```
    }
}

int main() {
    // vector<vector<vector<int>>> tem(1000,
vector<vector<int>>(24, vector<int>(10, 0)));
    // setting up the array with random data between 1 to
100;
    for (int i = 0; i < CITY; i++) {
        for (int j = 0; j < HOUR; j++) {
            for (int k = 0; k < READINGS; k++) {
                tem[i][j][k] = 1 + (rand() % 100);
            }
        }
    }

    for (int i = 0; i < CITY; i++) {
        for (int j = 0; j < HOUR; j++) {
            for (int k = 0; k < READINGS; k++) {
                mean_tem[i][j] += tem[i][j][k];
            }
            mean_tem[i][j] /= READINGS;
        }
    }

    min_temperature(7);
    max_temperature(7);
    avg_temperature(7);
    return 0;
}
```

Problem 2

```
    }
}
int main() {
    generate_hex();

    // setting up the array with random data between 0 to 255;
    for (int i = 0; i < GRID; i++) {
        for (int j = 0; j < GRID; j++) {
            pixle[i][j] = (rand() % 256);
        }
    }
    prind_pixle_grid(20, 20);
    inc_pixle(20);
    return 0;
}
```

```

void inc_pixle(int count) {
    for (int i = 0; i < GRID; i++) {
        for (int j = 0; j < GRID; j++) {
            pixle[i][j] = min(255, pixle[i][j] + count);
        }
    }
}
```

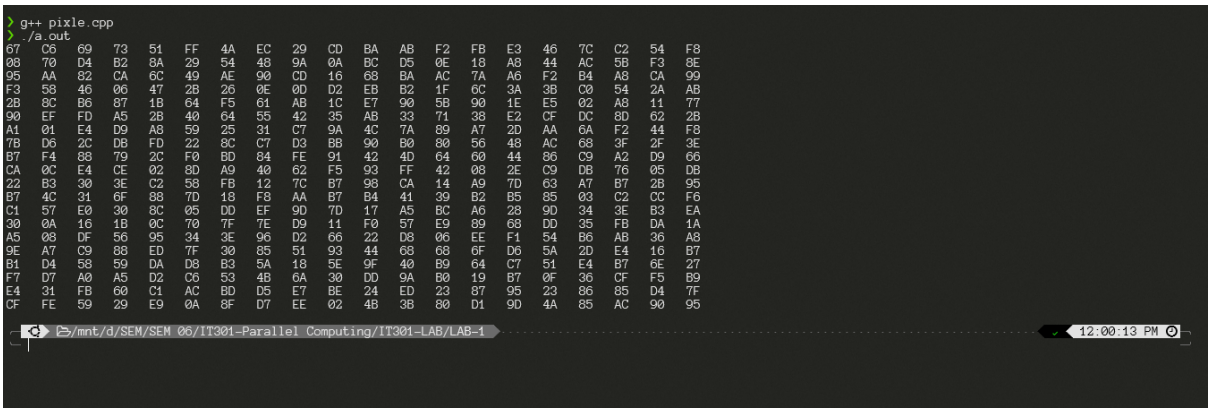
```

// prints the temperature information
void prind_pixle_grid(int max_row=GRID, int max_col=GRID) {
    for (int i = 0; i < min(max_row, GRID); i++) {
        for (int j = 0; j < min(max_col, GRID); j++) {

void generate_hex() {
    char hex[16] = { '0', '1', '2', '3', '4', '5', '6', '7', '8', '9', 'A', 'B', 'C', 'D', 'E' , 'F' };
    for (int i = 0; i < 16; i++) {
        for (int j = 0; j < 16; j++) {
            string str = "\0";
            str.push_back(hex[i]);
            str.push_back(hex[j]);
            pixle_hex.push_back(str);
        }
    }
    sort(pixle_hex.begin(), pixle_hex.end());
}
```

Printing one part of 256X256 GRID

First 25X25 GRID



Code for Problem 2

```
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;

#define GRID 256
int pixle[GRID][GRID];
vector<string> pixle_hex; // for storing hex conversion
// 00, 01, 02, .... , FD, FE, FF

void generate_hex() {
    char hex[16] = { '0', '1', '2', '3', '4', '5', '6',
'7', '8', '9', 'A', 'B', 'C', 'D', 'E' , 'F' };
    for (int i = 0; i < 16; i++) {
        for (int j = 0; j < 16; j++) {
            string str = "\0";
            str.push_back(hex[i]);
            str.push_back(hex[j]);
            pixle_hex.push_back(str);
```

```
        }
    }
    sort(pixle_hex.begin(), pixle_hex.end());
}

// prints the temperature information
void prind_pixle_grid(int max_row=GRID, int max_col=GRID) {
    for (int i = 0; i < min(max_row, GRID); i++) {
        for (int j = 0; j < min(max_col, GRID); j++) {
            cout << pixle_hex[pixle[i][j]] << "   ";
        }
        cout << endl;
    }
}

void inc_pixle(int count) {
    for (int i = 0; i < GRID; i++) {
        for (int j = 0; j < GRID; j++) {
            pixle[i][j] = min(255, pixle[i][j] + count);
        }
    }
}
int main() {
    generate_hex();

    // setting up the array with random data between 0 to
255;
    for (int i = 0; i < GRID; i++) {
        for (int j = 0; j < GRID; j++) {
            pixle[i][j] = (rand() % 256);
        }
    }
    prind_pixle_grid(20, 20);
    inc_pixle(20);
    return 0;
}
```