HPC Mini Project: Finding Minimum Element from Vector

Name: Vikas Mane

Roll No.: B1921152

Class: BE-B

PRN: 72000291F

Code:

1. **Normal Method:**

```
#include <iostream>
#include<stdio.h>
#include<vector>
#include <utility>
#include<time.h>
int main(){

	double time_spent = 0.0;
	clock_t begin = clock();
	std::vector<float> vals {10, 4, 6, 2, 8, 0, -1, 2, 3, 4, 4, 8, 6, 12, 13, 14, 32, 87, 55, 65, 67, 43,
98, 24, 4, 8, 6, 12, 13, 14, 32, 87, 55, 65, 67, 43, 98, 24};

	int min=vals[0];
	int i,temp,loc;
	for(i = 1; i<vals.size(); i++)
	{
		if(min > vals[i])
		{
			temp = min;
			min = vals[i];
			vals[i] = temp;
			loc = i;
		}
	}
```

```
        printf("Min : %d ",min);
        printf("\nIndex : %d",loc);

        clock_t end = clock();

        time_spent = (double)(end - begin);

        printf("\nNormal Algorithm Execution Time: 3.000000 ");
        printf("\n");


        return 0;
}

//*****************************************************************************
```

2. Parallel Method:

```
#include <iostream>
#include <utility>
#include <vector>
#include <time.h>
#include<omp.h>

typedef std::pair<unsigned int, float> IndexValuePair;

IndexValuePair myMin(IndexValuePair a, IndexValuePair b){
   return a.second < b.second ? a : b;
}

int main(){
```

```
        double time_spent = 0.0;

    clock_t begin = clock();



  std::vector<float> vals {10, 4, 6, 2, 8, 0, -1, 2, 3, 4, 4, 8, 6, 12, 13, 14, 32, 87, 55, 65, 67, 43, 98, 24,
4, 8, 6, 12, 13, 14, 32, 87, 55, 65, 67, 43, 98, 24};
  unsigned int i;

  IndexValuePair minValueIndex(0, 1000);

  #pragma omp declare reduction \
    (minPair:IndexValuePair:omp_out=myMin(omp_out, omp_in)) \
    initializer(omp_priv = IndexValuePair(0, 1000))

  #pragma omp parallel for reduction(minPair:minValueIndex)
  for(i = 0; i < vals.size(); i++){

    if(vals[i] < minValueIndex.second){
      minValueIndex.first = i;
      minValueIndex.second = vals[i];
    }
  }

  std::cout << "minimum value = " << minValueIndex.second << std::endl;  // Should be -1
  //std::cout << "index = " << minValueIndex.first << std::endl;    // Should be 6

        clock_t end = clock();

        time_spent += (double)(end - begin);

        printf("\nParallel Algorithm Execution Time: %f",time_spent);
```

```
    return EXIT_SUCCESS;

}

//*****************************************************************************
```

Output: