Câu 1.

Cài đặt OpenMP và thêm các câu lệnh để thực thi chương trình sau song song:

```
int main() {

     // Do this part in parallel

     printf( "Hello, World!\n" );
     return 0;
}
```

** Tham khảo slides*

Câu 2.

Viết chương trình song song với OpenMP, sử dụng `omp_get_num_threads` và `omp_get_thread_num` để in ra số lượng active threads cũng như `id` của chúng.

Câu 3.

Cho một mảng số nguyên gồm N phần tử. Viết chương trình song song với OpenMP để tính tổng các phần tử của mảng.

Câu 4.

Cho một mảng số nguyên gồm N phần tử. Viết chương trình song song với OpenMP để tính giá trị lớn nhất/ giá trị nhỏ nhất của mảng.

Câu 5.

Cho 2 mảng số nguyên A, B cùng số lượng phần tử là N. Viết chương trình song song với OpenMP tính mảng C biết C[i]=A[i]+B[i].

Câu 6.

Cho 2 mảng số nguyên A, B cùng số lượng phần tử là N. Viết chương trình song song với OpenMP tính tổng

$$sum = \sum_{i=0}^{N-1} (A[i] * B[i])$$

Câu 7.

Cho chương trình tính số PI tuần tự như sau:

```
#include <stdio.h>
static long num_rects=1000000;
void main()
{
     int i;
     double mid, height, width, sum = 0.0;
     double area;
     width = 1.0/(double) num_rects;
     for (i = 0; i < num_rects; i++){
         mid = (i + 0.5) * width;
         height = 4.0/(1.0 + mid*mid);
         sum += height;
     }
     area = width * sum;
     printf("Computed pi = %f\n",area);
}
```

Viết chương trình song song với OpenMP để tính số PI. Thực thi chương trình theo số thread tương ứng và đánh giá kết quả theo bảng sau:

# Threads	Runtime[sec]	Speedup	Efficiency
1			
2			
3			
4			
5			
6			
7			
8			

**Chú ý loại bỏ hết data race (2 threads cũng truy xuất và thay đổi 1 biến dùng chung mà không có đồng bộ)*

Câu 8.

Viết chương trình song song với OpenMP nhân 2 ma trận A với kích thước ma trận A[M,N] và ma trận B có kích thước B[N, 1]. Thực thi chương trình với số thread tương ứng và đánh giá kết quả theo bảng sau:

# Threads	Runtime[sec]	Speedup	Efficiency
1			
2			
4			
8			

**Xét các kích thước ma trận khác nhau, có thể tham khảo xét các cặp* (M, N) *tương ứng với* $(8000000,8)$*,* $(8000,8000)$ *và* $(8,8000000)$*.*