HPC
Experiment 3
Bubble Sort (Serial and Parallel)

Hrushikesh Pandit
63 TYCSE
Panel F

Code:

Bubble:

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <time.h>

void swap(int32_t *, int32_t *);
int32_t bubble_sort(int32_t *, uint32_t);

int32_t main() {
  int32_t *arr;
  uint32_t i = 0, N = 0;

  srand(time(NULL));
  do {
    printf("What should the size of array be?\nSize: ");
    scanf("%d", &N);
  } while(N <= 0);

  arr = (int32_t *) malloc(N * sizeof(int32_t));

  while(i < N) {
    arr[i++] = rand();
  }

  bubble_sort(arr, N);

  for(i = 0; i < N; i ++)
    printf("%d ", arr[i]);

  return 0;
}

void swap(int32_t *a, int32_t *b) {
 *a = *a ^ *b;
 *b = *a ^ *b;
```

```
 *a = *a ^ *b;

 return;
}

int32_t bubble_sort(int32_t *arr, uint32_t N) {
  if(arr == NULL) {
    printf("Reference to null.");
    return 1;
  }

  for (int32_t i = 0; i < N; i ++) {
    for(int32_t j = i+1; j < N; j ++) {
      if(*(arr+i) > *(arr+j)) {
        swap(arr+i, arr+j);
      }
    }
  }

  return 0;
}
```

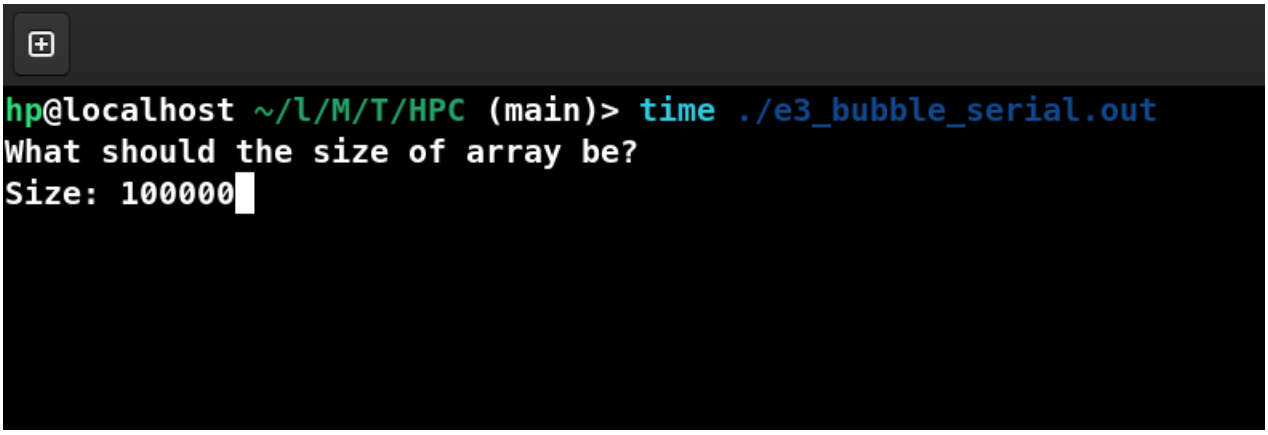
Brick sort:

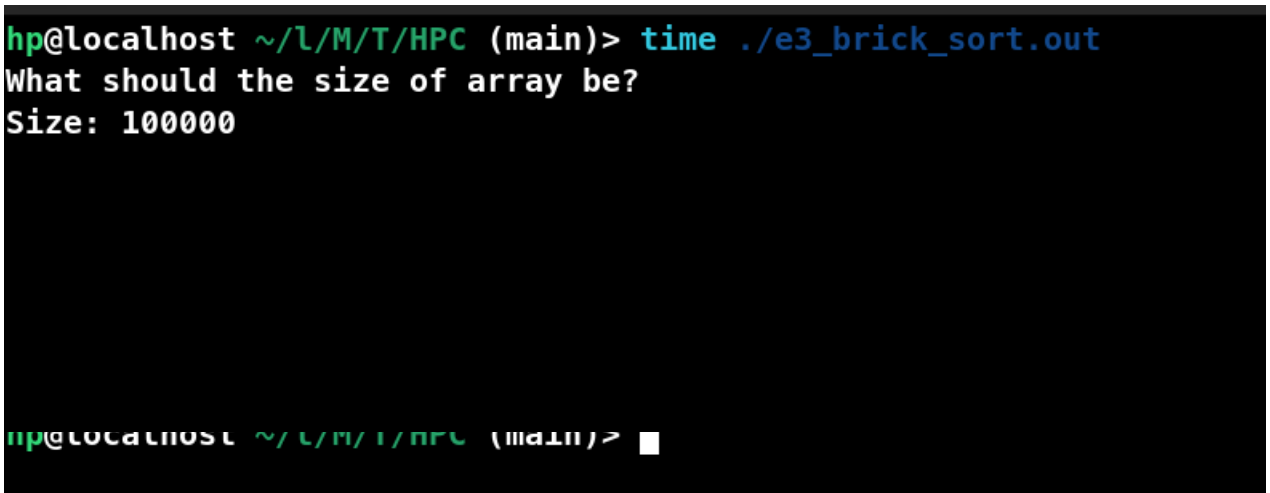
```
#include <stdio.h>
#include <omp.h>
#include <stdint.h>
#include <stdlib.h>
#include <time.h>
#include <stdbool.h>

int32_t brick_sort(int32_t *, uint32_t);
int32_t print_arr(int32_t *, uint32_t);
uint32_t swap(int32_t *, int32_t *);

uint32_t main(void) {
    uint32_t N = 0;
    uint32_t i = 0;
    int32_t *arr = NULL;
    srand(time(NULL));

    do {
        printf("What should the size of array be?\nSize: ");
        scanf("%d", &N);
    }while(N <= 0);
```

```
    arr = (int32_t *) malloc(N * sizeof(int32_t));

    while(i < N) {
        arr[i++] = rand();
    }
    brick_sort(arr, N);
    print_arr(arr, N);

    return 0;
}

int32_t print_arr(int32_t *arr, uint32_t N) {
    uint32_t i = 0;
    if(arr == NULL) {
        return 1;
    }

    while(i < N)
        printf("%d ", *(arr+i++));

    return 0;
}

uint32_t swap(int32_t *a, int32_t *b) {
    *a = *a ^ *b;
    *b = *a ^ *b;
    *a = *a ^ *b;
}

int32_t brick_sort(int32_t *arr, uint32_t N) {
    bool sorted = false;
    uint32_t i = 0;

    if(arr == NULL) {
        return 1;
    }

    while(!sorted) {
        sorted = true;
        #pragma omp for
            for (i = 1; i < N; i ++) {
                if(arr[i] > arr[i + 1]) {
                    swap(&arr[i], &arr[i+1]);
                    sorted = false;
```

```
        }
      }

    #pragma omp for
      for (i = 0; i < N - 1; i ++) {
        if(arr[i] > arr[i + 1]) {
          swap(&arr[i], &arr[i+1]);
          sorted = false;
        }
      }
  }

  return 0;
}
```

Output:





```
583804 2146586192 2146595846 2146610759 2146626821 2146677133 2146
64 2146971727 2147003352 2147009654 2147034525 2147044468 21470455
147398023 2147398778 2147399262 2147447880 2147458069 2147470459

_____________________________________________________________________
Executed in   35.98 secs    fish           external
   usr time   34.17 secs  315.00 micros   34.17 secs
   sys time    0.01 secs  181.00 micros    0.01 secs

hp@localhost ~/l/M/T/HPC (main)>
```