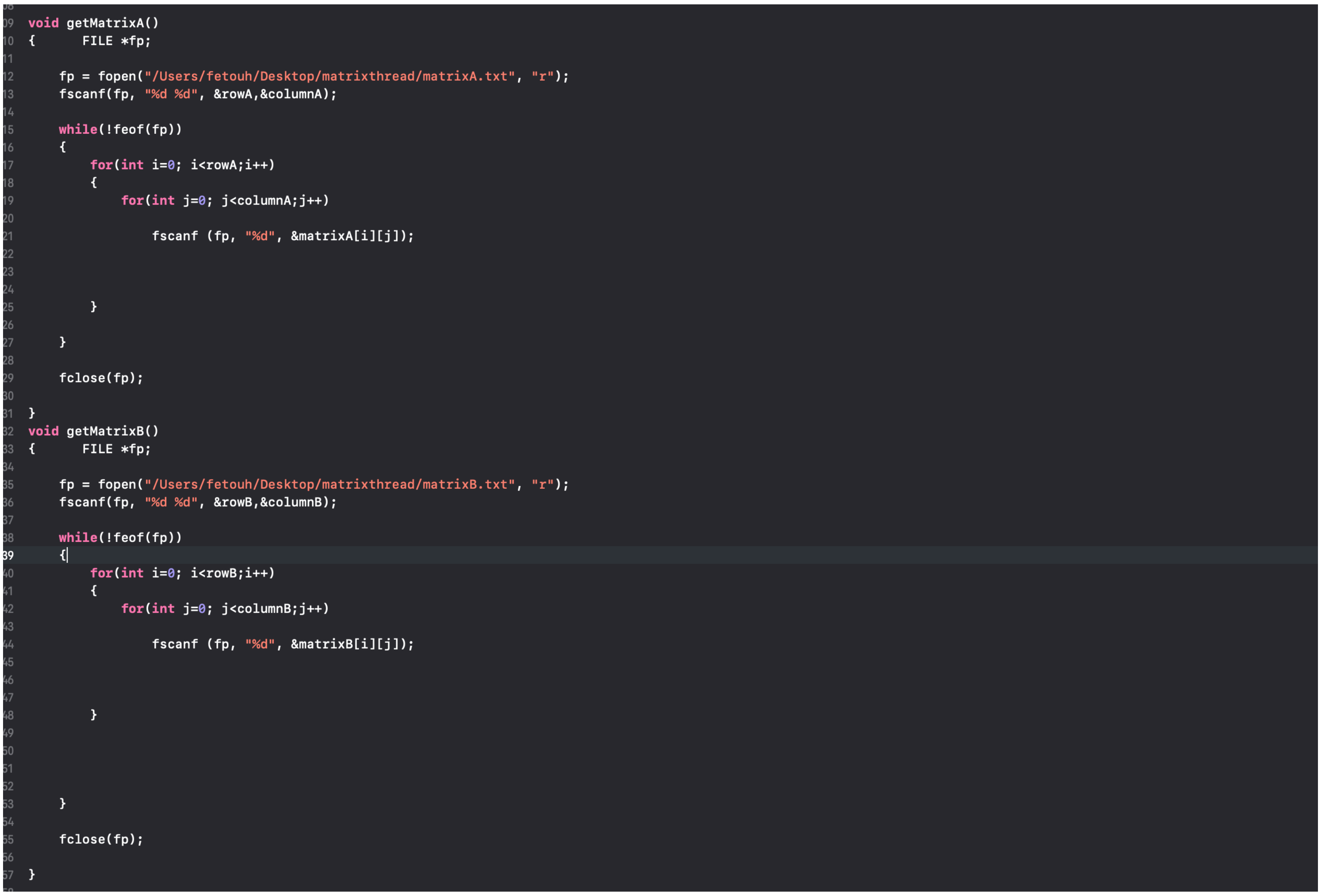
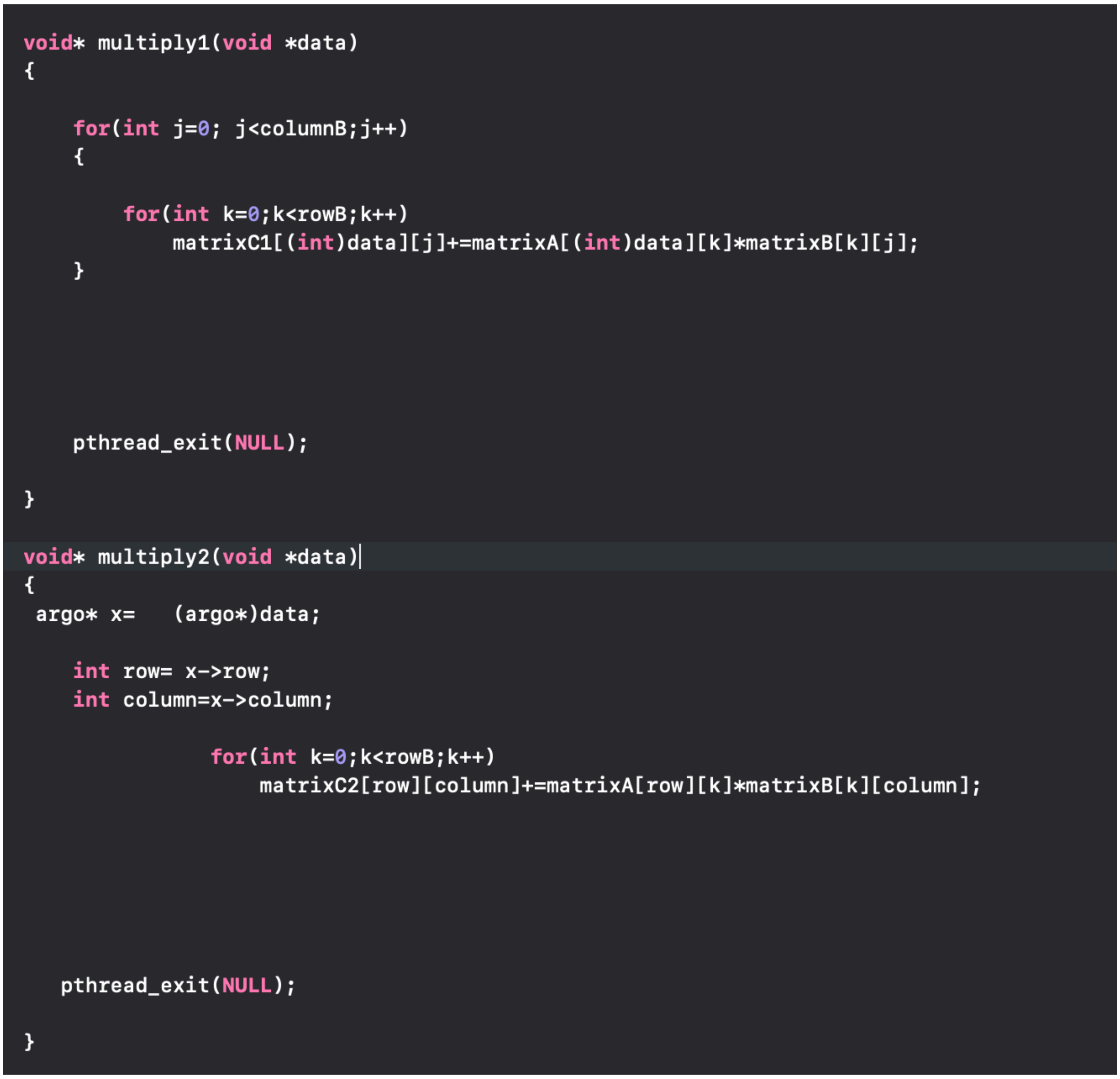
Matrix Multiplication



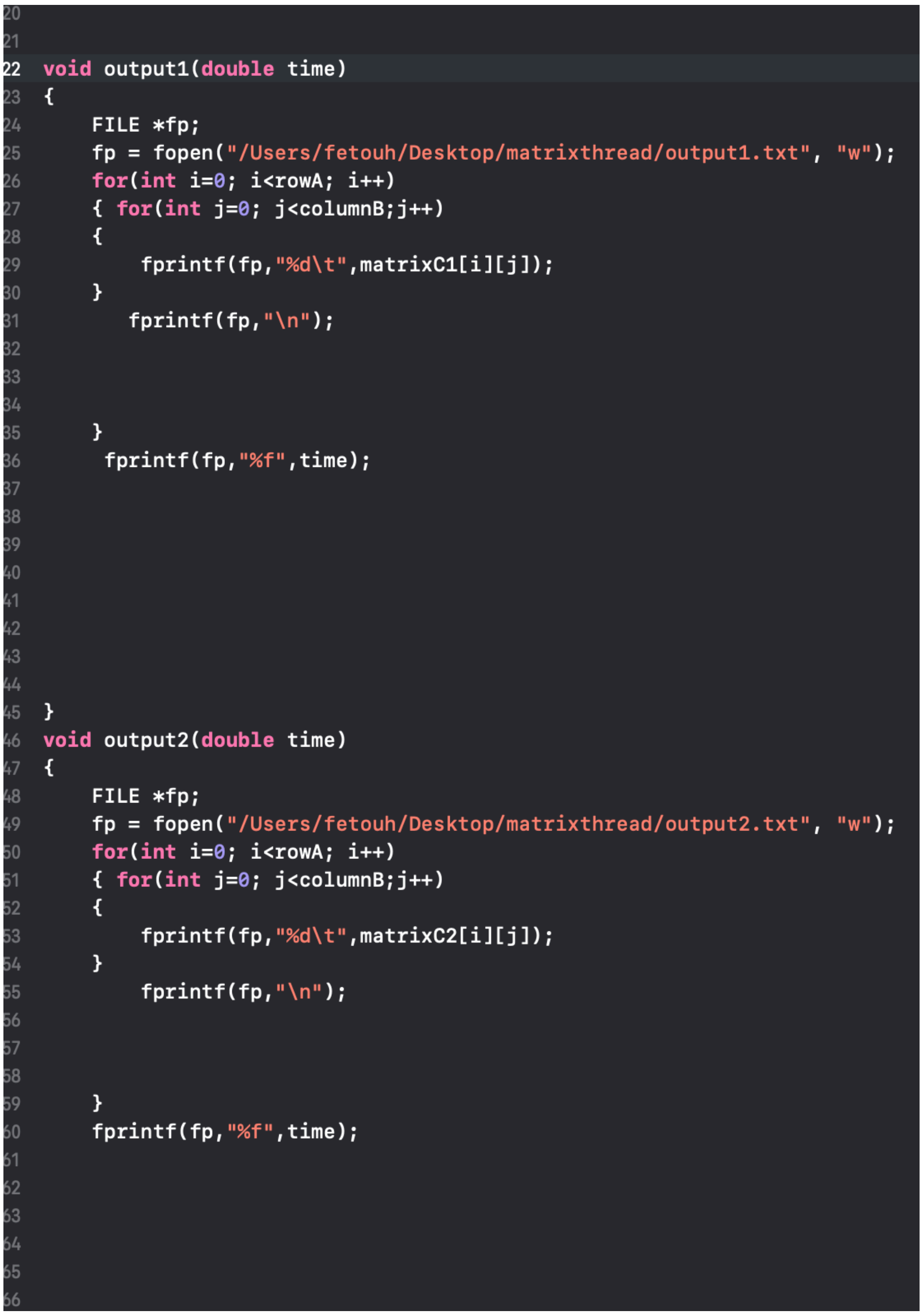
```
void getMatrixA()
{      FILE *fp;

    fp = fopen("/Users/fetouh/Desktop/matrixthread/matrixA.txt", "r");
    fscanf(fp, "%d %d", &rowA,&columnA);

    while(!feof(fp))
    {
        for(int i=0; i<rowA;i++)
        {
            for(int j=0; j<columnA;j++)

                fscanf (fp, "%d", &matrixA[i][j]);



        }

    }

    fclose(fp);

}
void getMatrixB()
{      FILE *fp;

    fp = fopen("/Users/fetouh/Desktop/matrixthread/matrixB.txt", "r");
    fscanf(fp, "%d %d", &rowB,&columnB);

    while(!feof(fp))
    {
        for(int i=0; i<rowB;i++)
        {
            for(int j=0; j<columnB;j++)

                fscanf (fp, "%d", &matrixB[i][j]);



        }




    }

    fclose(fp);

}
```

input both matrices using getMatrixA and getMatrixB

Multiply functions used to multiply matrices in both ways

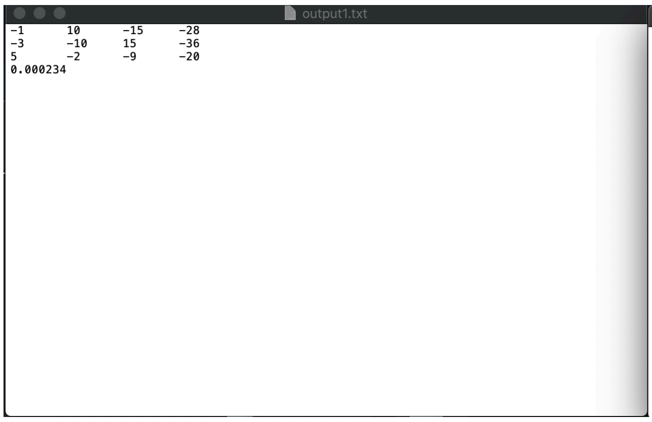


```c
void* multiply1(void *data)
{

    for(int j=0; j<columnB;j++)
    {

        for(int k=0;k<rowB;k++)
            matrixC1[(int)data][j]+=matrixA[(int)data][k]*matrixB[k][j];
    }




    pthread_exit(NULL);

}

void* multiply2(void *data)
{
 argo* x=   (argo*)data;

    int row= x->row;
    int column=x->column;

               for(int k=0;k<rowB;k++)
                   matrixC2[row][column]+=matrixA[row][k]*matrixB[k][column];





   pthread_exit(NULL);

}
```

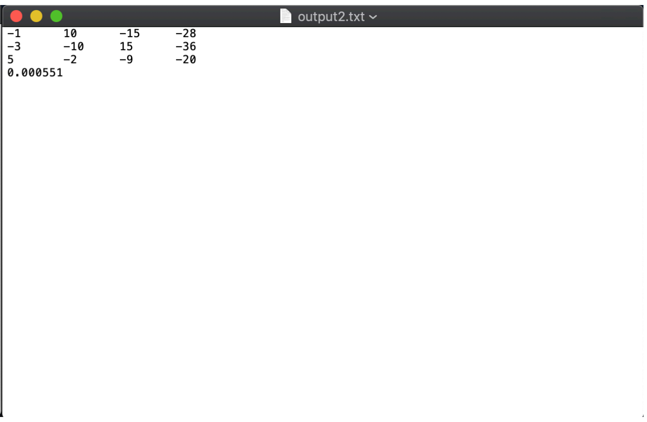


```c


void output1(double time)
{
    FILE *fp;
    fp = fopen("/Users/fetouh/Desktop/matrixthread/output1.txt", "w");
    for(int i=0; i<rowA; i++)
    { for(int j=0; j<columnB;j++)
    {
        fprintf(fp,"%d\t",matrixC1[i][j]);
    }
       fprintf(fp,"\n");



    }
     fprintf(fp,"%f",time);









}
void output2(double time)
{
    FILE *fp;
    fp = fopen("/Users/fetouh/Desktop/matrixthread/output2.txt", "w");
    for(int i=0; i<rowA; i++)
    { for(int j=0; j<columnB;j++)
    {
        fprintf(fp,"%d\t",matrixC2[i][j]);
    }
        fprintf(fp,"\n");



    }
    fprintf(fp,"%f",time);
```

print both matrices and time taken in a text file using output functions and print time taken time taken using threads for each row is less than thread less than each element .

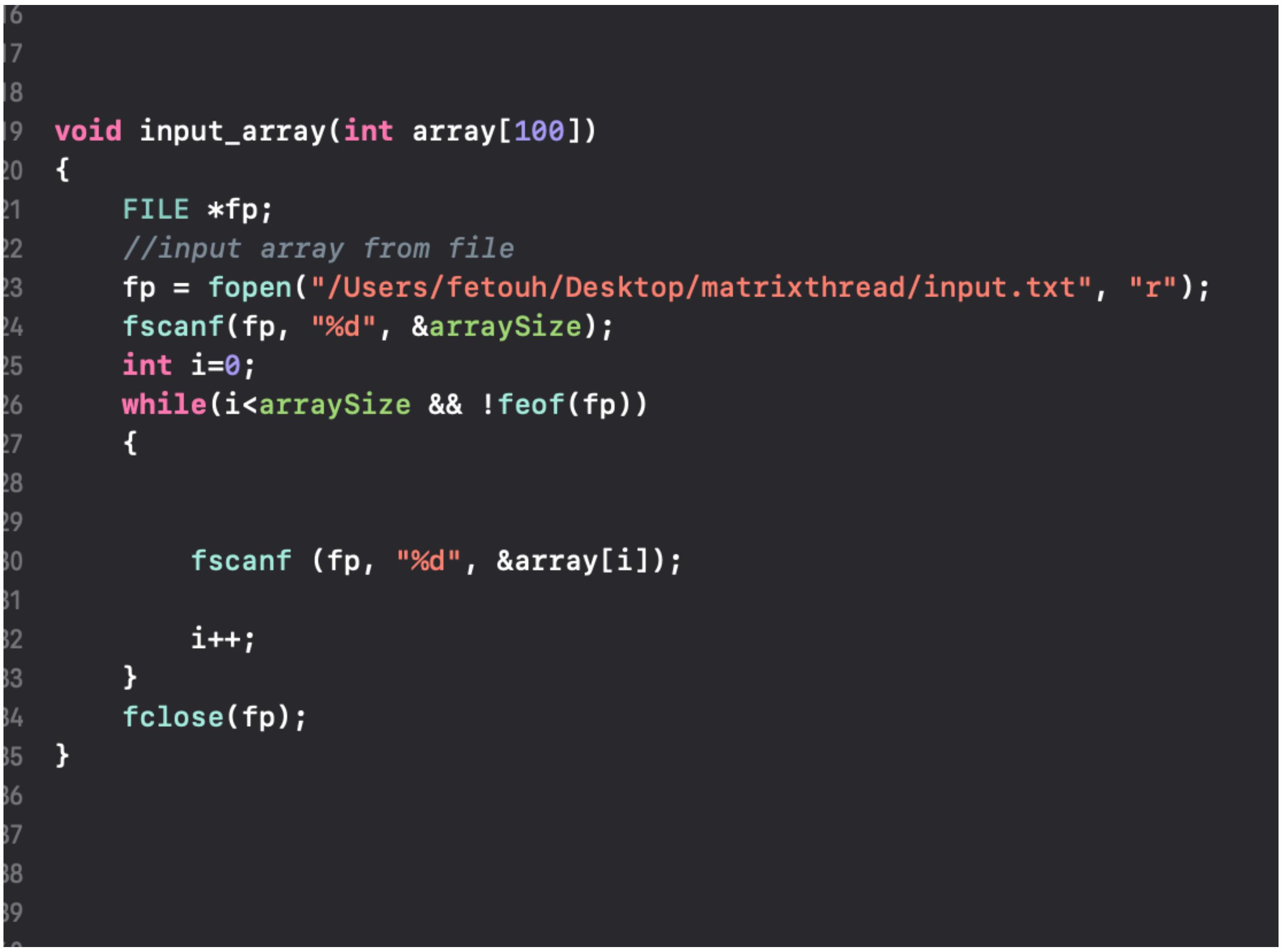


```
-1	10	-15	-28
-3	-10	15	-36
5	-2	-9	-20
0.000234
```



```
-1	10	-15	-28
-3	-10	15	-36
5	-2	-9	-20
0.000551
```

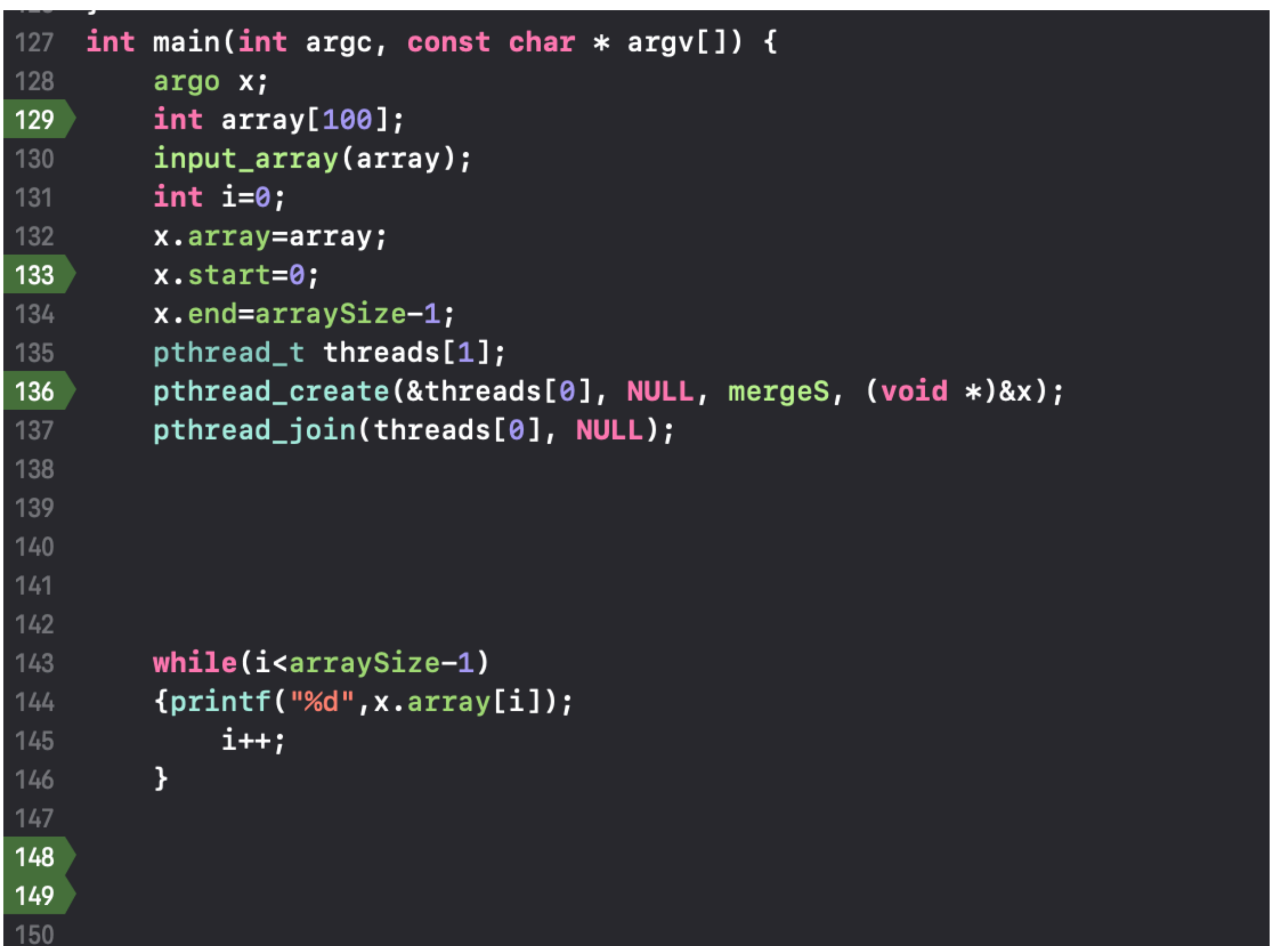
Merge Sort



```



void input_array(int array[100])
{
    FILE *fp;
    //input array from file
    fp = fopen("/Users/fetouh/Desktop/matrixthread/input.txt", "r");
    fscanf(fp, "%d", &arraySize);
    int i=0;
    while(i<arraySize && !feof(fp))
    {


        fscanf (fp, "%d", &array[i]);

        i++;
    }
    fclose(fp);
}
```

input array and array size from text file

Create new thread and call merge sort. Use pthread join to wait for all threads to finish.
Print array after mergesort



```
int main(int argc, const char * argv[]) {
    argo x;
    int array[100];
    input_array(array);
    int i=0;
    x.array=array;
    x.start=0;
    x.end=arraySize-1;
    pthread_t threads[1];
    pthread_create(&threads[0], NULL, mergeS, (void *)&x);
    pthread_join(threads[0], NULL);





    while(i<arraySize-1)
    {printf("%d",x.array[i]);
        i++;
    }



```