Design Document of Assignment 3
Zihua Li
CSE 13S
Prof. Darrell Long

The purpose of this assignment is to implement Shell sort, Batcher Sort (Batcher's method), Heapsort and recursive Quicksort, using the pseudocode provided, and a test harness, statistical data about the performance. // Shell sort, pseudo code is followed.

Shell Sort in Python

```python
def shell_sort(arr):
    for gap in gaps:
        for i in range(gap, len(arr)):
            j = i
            temp = arr[i]
            while j >= gap and temp < arr[j - gap]:
                arr[j] = arr[j - gap]
                j -= gap
            arr[j] = temp
```

I used a for loop to switch between values in gaps, which the python program generated, and while loop to complete the sorting. Sorting the array given. Also refer to stats.c, move() and cmp() are used, also functions need to have Stats *stats or will not count the statistics. // Heapsort, psudo code is followed

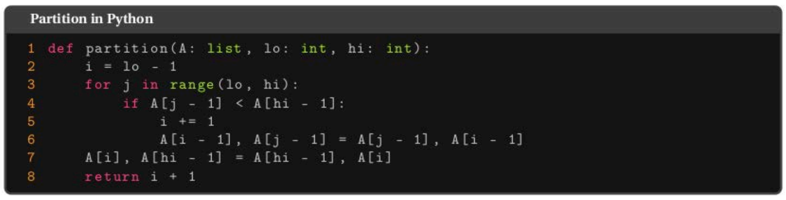
Heap maintenance in Python

```python
def max_child(A: list, first: int, last: int):
    left = 2 * first
    right = left + 1
    if right <= last and A[right - 1] > A[left - 1]:
        return right
    return left

def fix_heap(A: list, first: int, last: int):
    found = False
    mother = first
    great = max_child(A, mother, last)

    while mother <= last // 2 and not found:
        if A[mother - 1] < A[great - 1]:
            A[mother - 1], A[great - 1] = A[great - 1], A[mother - 1]
            mother = great
            great = max_child(A, mother, last)
        else:
            found = True
```

Heapsort in Python

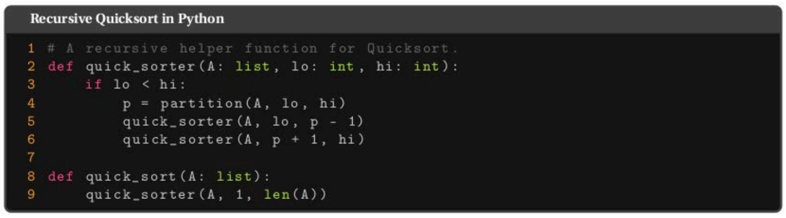
```python
def build_heap(A: list, first: int, last: int):
    for father in range(last // 2, first - 1, -1):
        fix_heap(A, father, last)

def heap_sort(A: list):
    first = 1
    last = len(A)
    build_heap(A, first, last)
    for leaf in range(last, first, -1):
        A[first - 1], A[leaf - 1] = A[leaf - 1], A[first - 1]
        fix_heap(A, first, leaf - 1)
```

I basically followed the pseudocode provided, it is nothing more than converting Python to C. Also refer to stats.c, cmp(), swap() are used, also functions need to have Stats *stats or will not count the statistics. // Quicksort, pseudo code is followed



Partition in Python

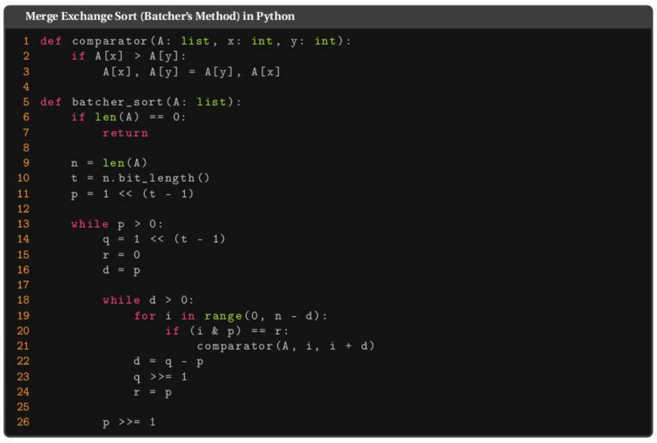
```python
def partition(A: list, lo: int, hi: int):
    i = lo - 1
    for j in range(lo, hi):
        if A[j - 1] < A[hi - 1]:
            i += 1
            A[i - 1], A[j - 1] = A[j - 1], A[i - 1]
    A[i], A[hi - 1] = A[hi - 1], A[i]
    return i + 1
```



Recursive Quicksort in Python

```python
# A recursive helper function for Quicksort.
def quick_sorter(A: list, lo: int, hi: int):
    if lo < hi:
        p = partition(A, lo, hi)
        quick_sorter(A, lo, p - 1)
        quick_sorter(A, p + 1, hi)

def quick_sort(A: list):
    quick_sorter(A, 1, len(A))
```

Following the pseudocode. Refer to stats.c, cmp(), swap() are used, also functions need to have Stats *stats or will not count the statistics. // Batcher sort, pseudo code is followed



Merge Exchange Sort (Batcher's Method) in Python

```python
def comparator(A: list, x: int, y: int):
    if A[x] > A[y]:
        A[x], A[y] = A[y], A[x]

def batcher_sort(A: list):
    if len(A) == 0:
        return

    n = len(A)
    t = n.bit_length()
    p = 1 << (t - 1)

    while p > 0:
        q = 1 << (t - 1)
        r = 0
        d = p

        while d > 0:
            for i in range(0, n - d):
                if (i & p) == r:
                    comparator(A, i, i + d)
            d = q - p
            q >>= 1
            r = p

        p >>= 1
```

Following the pseudocode. Refer to stats.c, cmp(), swap() are used, also functions need to have Stats *stats or will not count the statistics. Bitwise operations are used in the program. // For sorting.c, getopt() and switch case are used. When it detects a parameter is inputted, it will change the corresponding variable to 1, hence when we are out of the while loop, it will generate data and start printing correspondingly. // The test harness generally use the same methods like in the last assignment.

Work Cited:

Python pseudocode from the assignment instruction file