Din Bostandzic
CSE 13S

Assignment 5 - Sorting Program Design

Pre-Lab Answers

Part 1

1. 4 rounds of swapping will be required to sort the numbers 8,22,7,9,31,5,13 using bubble sort in ascending order, with a total of 10 comparisons being made.
2. There will be n(n-1)/2 comparisons made in the worst case scenario of bubble sort.

Part 2

1. The worst case time complexity of shell sort depends on the gap size because the bigger the gap, the more passes through the elements are required, thus resulting in more comparisons and swaps taking place. To improve the time complexity of this sort by changing the gap size, it is achieved by decreasing the gap by adjusting the equation for gaps, which will result in less inputs to be checked.
2. To improve the runtime of this sort without changing the gap size, use pre-existing gap values that will optimize the sort or include a binary search to quickly locate where to place the elements.

Part 3

1. The worst case time complexity of O(n^2) for Quicksort only occurs when a pivot point of the starting index or very last index of an array is chosen, which results in having to go through each value. Quicksort is not doomed by this issue because usually the pivot is chosen to be the middle index in the array to get optimized.

Part 4

1. When the binary search algorithm is combined with insertion sort algorithm, the effect is a reduced time complexity, as the items can get their locations of placement more quickly.

Part 5

1. Since each sort will reside in its own header file, to keep track of the number of moves and comparisons, an extern variable will be used in the main, accessing the counts from headers.

```
void setopt(opt* option, opt enum_opt)
{
        *option |= (1<<enum_opt);

}

bool checkopt(opt* option, opt enum_opt)
{
        Return *option & (1<<enum_opt)
}
```

main module data design

Define OPTIONS as constant string “Absqip:r:n:”
Define opts as enumerated type {A, b, s, q, i, p, r, n}
Declare enum opts option
Declare A, b, s, q, i, p, r, n as bool initialized each to 0
Declare default_arraysize as integer initialize to 100
Declare default_print as integer initialized to 100
Declare default_seed as integer initialize to 8222022
Declare max_num = (1 << 30)-1

Declare input_num as string initialized to NULL
Declare start_enum = 0;

main module design

```
Begin main(int arc, char **argv)
        Begin While ((c = getopt(argc, argv,OPTIONS)) != -1)
                Switch(c)
                        Case ‘A’
                                Assign value of true to A
                                setopt(option, start_enum)
                                Break
                        Case ‘b’
                                Assign value of true to b
                                setopt(option, start_enum++)
                                Break
                        Case ‘s’
                                Assign value of true to s
                                setopt(option, start_enum++)
```

```
                Break
        Case ‘q’
                Assign value of true to q
                setopt(option, start_enum++)
                Break
        Case ‘i’
                Assign value of true to i
                setopt(option, start_enum++)
                Break
        Case ‘p’
                Assign value of true to p
                setopt(option, start_enum++)
                Assign to input_num value of optarg
                Assign to default_print value of input_num as integer
                Break
        Case ‘r’
                Assign value of true to r
                setopt(option, start_enum++)
                Assign to input_num value of optarg
                Assign to default_seed value of input_num as integer
                Break
        Case ‘n’
                Assign value of true to n
                setopt(option, start_enum++)
                Assign to input_num value of optarg
                Assign to default_arraysize value of input_num as integer
                Break
        Default case
                Display “Character not found in string”
                Return exit status fail
    End Switch
End while
Begin if (argc == 1)
    Display “No arguments supplied!”
    Return exit status fail
End if
srand(default_seed );
Dynamically create array of integer type, size default_arraysize
Begin For(int i =0; i <default_arraysize; i++)
    Array[i] = rand() % max_num
```

```
        End for
        Begin for(enum opt i = A, i<=n; i++)
                Begin if(checkopt(options, i))
                        Begin Switch(i)
                                Case A
                                        Call all sorts(pass in array, pass in default_arraysize)
                                        Break
                                Case b
                                        Call bubblesort(pass in array, pass in default_arraysize)
                                        Break
                                Case s
                                        Call shellsort(pass in array, pass in default_arraysize)
                                        Break
                                Case q
                                        Call quicksort(pass in array, pass in default_arraysize)
                                        Break
                                Case i
                                        Call binary insertion sort(pass in array, pass in default_arraysize)
                                        Break
                                Case p
                                        Call print(pass in array, pass in default_arraysize, default_print)
                                        Break
                        End Switch
                End if
        End for
End main module
```