Assignment 1 Documentation

Scott Ross Klein 19279288

Solution Logic Pseudo Code

Main function

```
main
    List head
    number input = -1
    while input is not 0
        display input options
        get user input
        if input 1   //Build list option
            delete old list
            head = new list
        end if

        else if input 2
            sort list
        end if

        else if input 3
            print list
        end if

        else
            print "Input not recognised"
        end if
    end while
    print "Terminating"
end main
```

BuildList

```
buildList
    top = new block
    current = top

    repeat 10 times
        current.link = new block
        current = current.link
    end repeat

    return top
end buildList
```

DisplayList

```
displayList
    while head is not NULL
        print head.filler
        head = head.link
    end while
end displayList
```

SortList

```
sortList (head, priority())
    List current
```

```
number start = 0

number size = count(current)


repeat size times

    number smallestIndex

    number index

    number smallest = 50000

    flag   swapflag = false


    while current is not NULL

        if smallest > priority(current.base_pri) AND index >= start

            smallest = priority(current.base_pri)

            smallestIndex = index

            swapflag = true

        end if


        current = current.link

        index = index +1

    end while


    if swapflag is true

        head = swap(head, start, smallestIndex)

    end if


    start = start + 1

end repeat
```

```
    return head

end sortList
```

Swap

```
swap(top, a, b)

    aPrevious

    aNode

    bPrevious

    bNode

    aNext

    bNext


    number traversal = max(a+1, b+1)


    repeat traversal times

        if loop is a - 1

            aPrevious = current

        if loop is a

            aNode = current

        if loop is b - 1

            bPrevious = current

        if loop is b
```

```
            bNode = current
        current = current.link
    end repeat

    bNext = bNode.link
    aNext = aNode.link

    //perform the actual swap
    aPrevious.link = bNode
    bPrevious.link = aNode

    aBlock.link = bNext
    bBlock.link = aNext

end swap
```

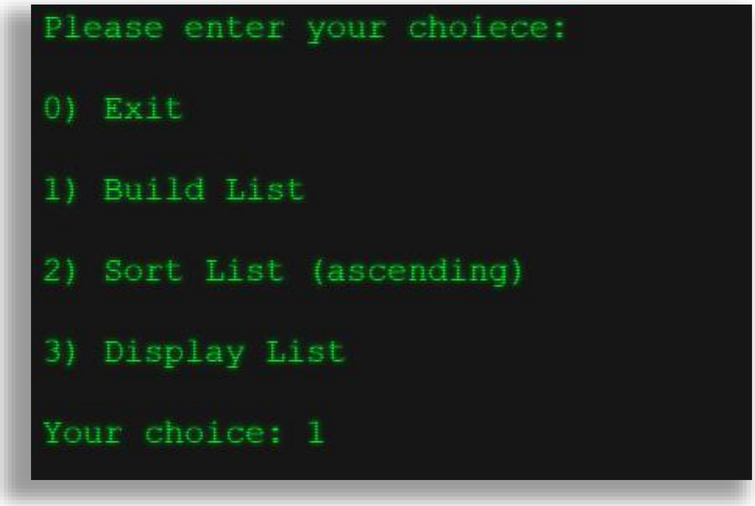
Test plan and limitations

Compilation

```
cc main.c -o main.o -std=c99
```

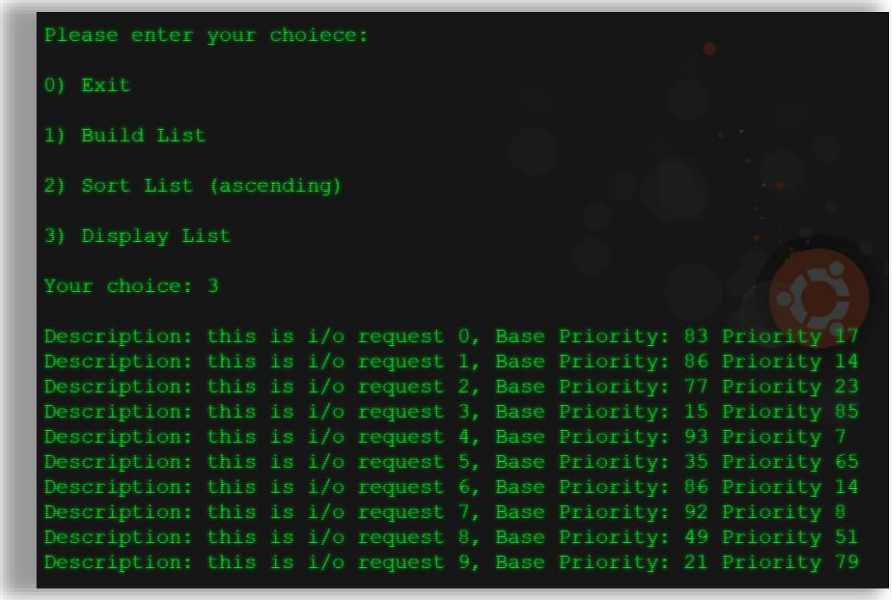
Running

Output should be as follows



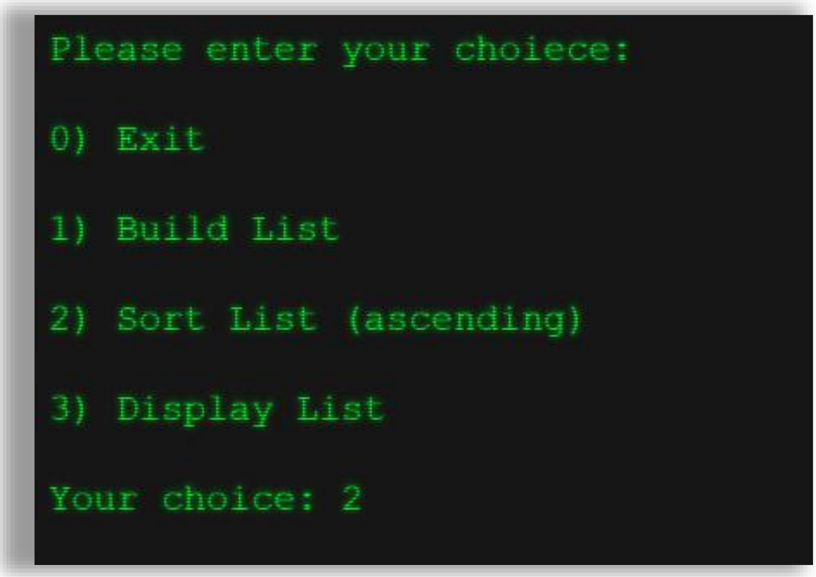
```
Please enter your choiece:

0) Exit

1) Build List

2) Sort List (ascending)

3) Display List

Your choice: 1
```

Input choice 1 to build an initial list



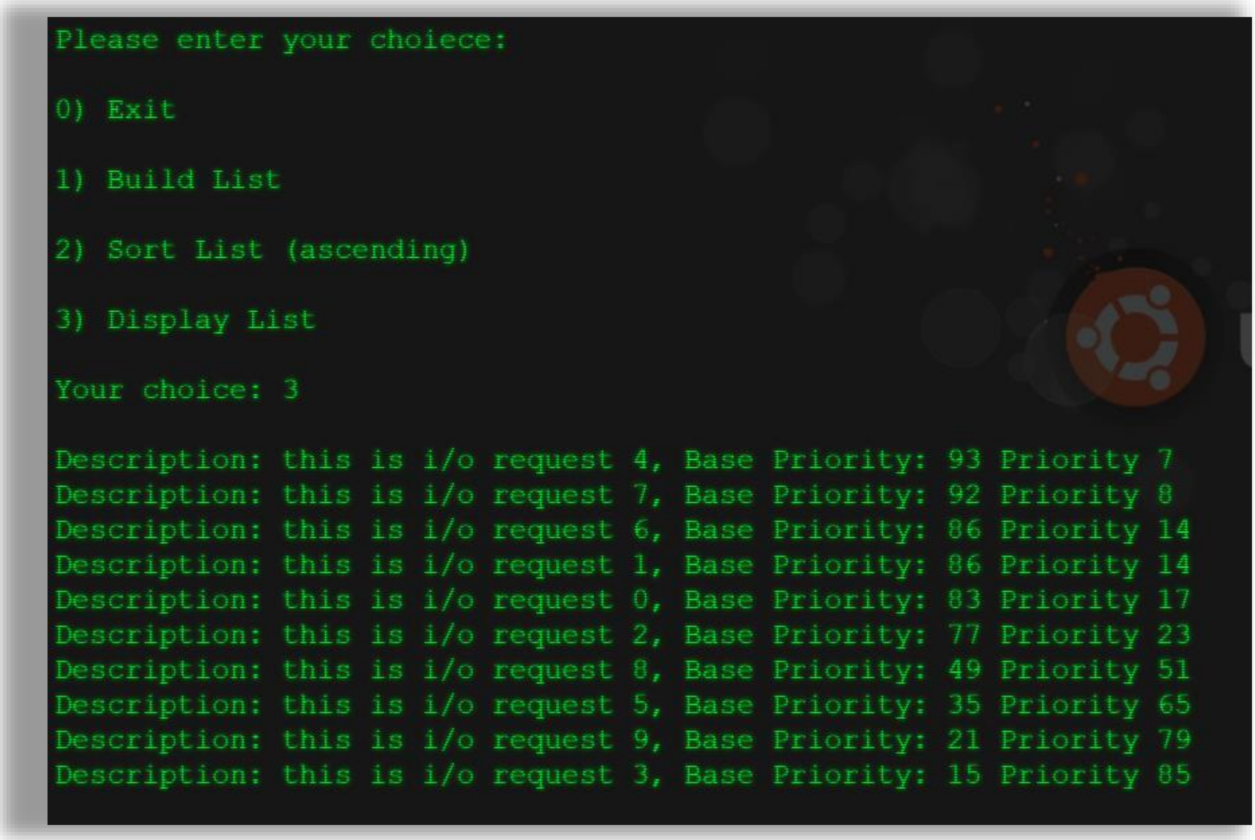
```
Please enter your choiece:

0) Exit

1) Build List

2) Sort List (ascending)

3) Display List

Your choice: 3

Description: this is i/o request 0, Base Priority: 83 Priority 17
Description: this is i/o request 1, Base Priority: 86 Priority 14
Description: this is i/o request 2, Base Priority: 77 Priority 23
Description: this is i/o request 3, Base Priority: 15 Priority 85
Description: this is i/o request 4, Base Priority: 93 Priority 7
Description: this is i/o request 5, Base Priority: 35 Priority 65
Description: this is i/o request 6, Base Priority: 86 Priority 14
Description: this is i/o request 7, Base Priority: 92 Priority 8
Description: this is i/o request 8, Base Priority: 49 Priority 51
Description: this is i/o request 9, Base Priority: 21 Priority 79
```

Choose 3 to display the initial list in its created order with random priority.



```
Please enter your choiece:

0) Exit

1) Build List

2) Sort List (ascending)

3) Display List

Your choice: 2
```

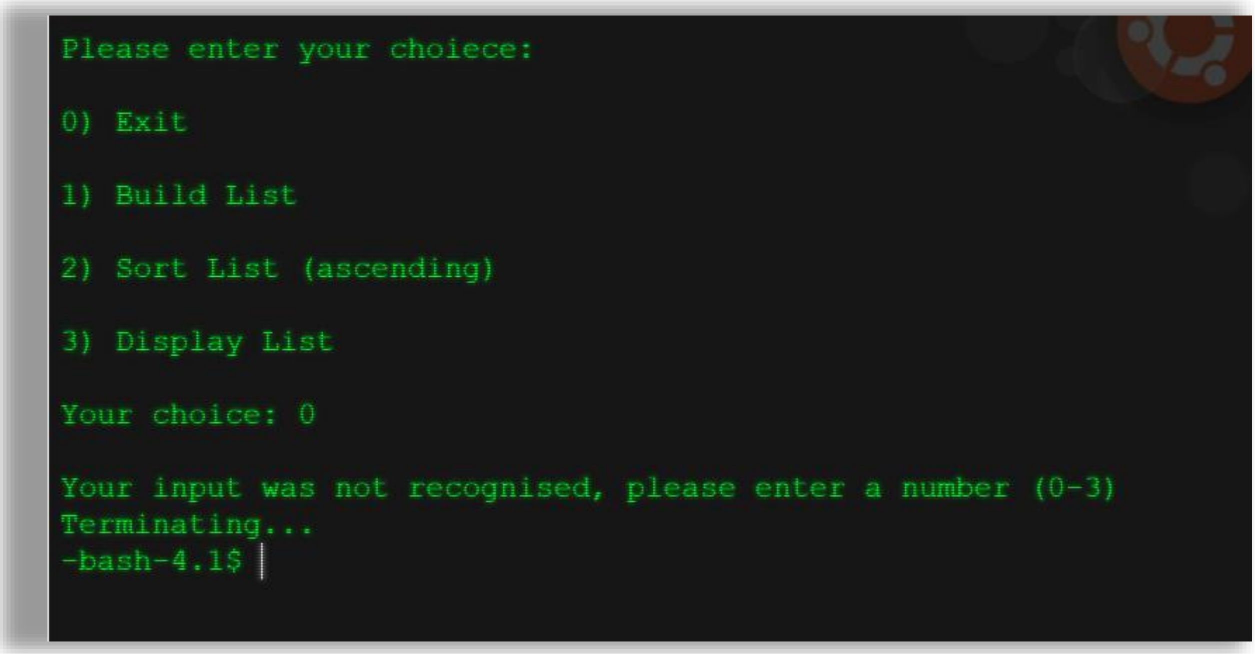
Select choice 2, this will order the list using the specified sorting algorithm.



```
Please enter your choiece:

0) Exit

1) Build List

2) Sort List (ascending)

3) Display List

Your choice: 3

Description: this is i/o request 4, Base Priority: 93 Priority 7
Description: this is i/o request 7, Base Priority: 92 Priority 8
Description: this is i/o request 6, Base Priority: 86 Priority 14
Description: this is i/o request 1, Base Priority: 86 Priority 14
Description: this is i/o request 0, Base Priority: 83 Priority 17
Description: this is i/o request 2, Base Priority: 77 Priority 23
Description: this is i/o request 8, Base Priority: 49 Priority 51
Description: this is i/o request 5, Base Priority: 35 Priority 65
Description: this is i/o request 9, Base Priority: 21 Priority 79
Description: this is i/o request 3, Base Priority: 15 Priority 85
```

The order of the list is now sorted based upon the priority of each IORB

Build list may be run again, the old list nodes will be freed with free() and a new list generated automatically. There are no memory leaks.



```
Please enter your choiece:

0) Exit

1) Build List

2) Sort List (ascending)

3) Display List

Your choice: 0

Your input was not recognised, please enter a number (0-3)
Terminating...
-bash-4.1$ 
```

Finally the program terminates Correctly.

Limitations

There are no known limitations. List size is fixed to 10 to match the example.