OS Lab Task 2

Omar Harb 900201063
Mohamed Shaalan 900201539
Sara Mohamed 900203032

Roles

Search Algorithm Code (Linear Search), done by Mohamed Shaalan

Sorting Algorithm Code (Selection Sort), done by Sara Mohamed

Statistics Code, done by Omar Harb

*Collaborated equally on researching and modifying xv6 files to fit requirements

Linear Search - Description & Pseudo Code

```
function isValidNum(num)
    for each character in num
        if character is not a digit
            return false
    return true
```

```
function myatoi(char* str)

            return sign (based on first index char) * result (converted char* number to int)
```

- The myatoi function is required due to a difference in passing command line arguments between regular C programs and in xv6 for negative numbers

```
function main(argc, argv)
    for each argument in argv[1:] # start from second argument
        if not isValidNum(argument)
            print "Invalid input. Exiting..."
            exit()

    target = atoi(argv[argc-1])

    for i = 1 to argc
        current = myatoi(argv[i])
        if current == target
            print "Element found at index ", i-1
            exit()

    print "Element not in array"
    exit()
```

Selection Sort - Description & Pseudo Code

```
function isValidNum(num)
    for each character in num
        if character is not a digit
            return false
    return true
```

- Same isValidNum function used in all added user programs to ensure all elements in argv are numeric (positive and negative) i.e valid for these types of programs

```
function main(argc, argv)
    for i = 1 to argc-1
        if not isValidNum(argv[i])
            print "Invalid input. Exiting...\n"
            exit()

    for i = 1 to argc-1
        for j = i+1 to argc-1
            if myatoi(argv[i]) > myatoi(argv[j])
                temp = argv[i]
                argv[i] = argv[j]
                argv[j] = temp

    for i = 1 to argc-1
        print myatoi(argv[i]), " "

    exit()
```

Stats Program- Description & Pseudo Code

```
function root(num)
    low = 0, high = n, mid = 0
    mid = (low+high)/2
    while (mid*mid != n)
        If (mid*mid > n)
            high = mid
        Else
            low = mid
        mid = (low + high)/2
    return mid
```

```
function isValidNum(num)
    for each character in num
        if character is not a digit
            return false
    return true
```

```
function main(argc, argv)
    validate argv as in previous programs
    sort using algorithm used in previous program

    sum = 0, sd = 0
    min = argv[1]
    max = argv[argc-1]

    for i = 1 to argc-1
        sum+= myatoi(argv[i])

    average = sum/(argc-1)
    median = argv[argc/2]

  for i = 1 to argc-1
        sd += (myatoi(argv[i])-avg)*(myatoi(argv[i])-avg))

  sd = root(sd)
  print min, max, sd, and median      # using appropriate functions
   exit()
```

- Since the <math.h> library is not in xv6, the built-in sqrt() function was unavailable, so a new root function was implemented using binary search

Files Added/Changed

- Changed the Makefile to add our programs under both the UPROGS_ and EXTRA sections
  - the Makefile acts as a guideline to the kernel as to which programs in the xv6 file to compile, which is done by adding the file name under UPROGS.
- Added the 3 .c files we wrote for the 3 user programs into the xv6 folder, alongside the other xv6 files
- Added printfloat() function in “printf.c” file, as well as its function declaration in “user.h” file, so it could be used to print statistics as floats (average and standard deviation)

Changes in .c files for xv6

- had to call exit() function
  - not necessary in regular .c programs
  - however, not calling it in this case may cause memory leaks and undefined behavior; as exit() terminates process to free any resources it was utilizing
- Included extra parameter in printf function
  - file descriptor, adds info about printing functionality
    - when =1, writes to the console in standard output stream
    - when =2, refers to standard error output, used for error messages etc.
- Include “types.h”, “stat.h”, “user.h” libraries
  - “stat.h” contains necessary info about a file’s **stat**us, like permission, type, and size
  - “user.h” contains necessary functions regarding environment of **user** like exit(), wait(), and sleep()
  - “types.h” defines data **types** used in system source code, defines collection of necessary structures and symbols