OS Lab 2: Adding User Programs to XV6

Farah Kabesh - 900191706
Karim Sherif - 900191474

Role of Team Members

Farah

- Selection Sort Program
- Changes in Makefile
- Presentation

Karim

- Linear Search Program
- Bubble Sort Program
- Changes in Makefile
- Changes in user.h and ulib.c files

3 User Programs

Linear Search
Selection Sort
Bubble Sort

Linear Search

- Finds whether a desired element x is present in an array.
- Uses a sequential search algorithm.
- Starts from the first element in the array and compares the element we're searching for (x) with each element in the array until it is found or the array ends.
  - If x is found, it returns its index.
  - Otherwise, it will return -1.

Pseudo-code for Linear Search

```
LINEAR_SEARCH (array, size, value)
   for each element in the array
      if element == value
         return the element’s index
      else
         return -1
```

Testing the Program

```
SeaBIOS (version 1.13.0-1ubuntu1.1)


iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+1FF8CB00+1FECCB00 CA00



Booting from Hard Disk...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap star
t 58
init: starting sh
$ linearS 45 87 62 31 98 21 62
Item is present at index 3
$ _
```

Selection Sort

- Sorting algorithm which finds the smallest element in an array of unsorted elements and places it at the beginning of the array.
- Proceeds to do this repeatedly by iterating through the rest of the unsorted elements in the array till they are correctly placed and sorted in ascending order.
- Relies on the concept of swapping elements.

Pseudo-code for Selection Sort

```
selectionSort (array, size)
   for (size-1) times
   set first element as minimum
   for each element in unsorted array
         if element < minimum
              set element as new minimum
   swap new minimum with first element
```

Testing the Program

```
SeaBIOS (version 1.13.0-1ubuntu1.1)


iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+1FF8CB00+1FECCB00 CA00



Booting from Hard Disk...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap star
t 58
init: starting sh
$ selectionsort 45 21 -5 17 -2 87 61
Sorted array in Acsending Order using SelectionSort:
-5  -2  17  21  45  61  87
$
```

Bubble Sort

- Sorting algorithm which starts with the first 2 elements of an array.
- The elements are swapped if the first element is greater than the second.
- The second and third elements are then compared and are swapped if they’re in the wrong order.
- Iterates through all the elements by repeating the above process till the last unsorted element in the array.
- Array is sorted when all elements are in the correct positions in ascending order.
- Relies on the concept of swapping elements.

Pseudo-code for Bubble Sort

```
bubbleSort (array, size)
   for step ← 0 to size -1
     for i ← 1 to size - step -1
      if array[i] > array[i+1]
         temp = array[i]
         array[i] = array[i+1]
         array[i+1] = temp
```

Testing the Program

```
SeaBIOS (version 1.13.0-1ubuntu1.1)


iPXE (http://ipxe.org) 00:03.0 CA00 PCI2.10 PnP PMM+1FF8CB00+1FECCB00 CA00



Booting from Hard Disk...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap star
t 58
init: starting sh
$ bubblesort 45 87 50 25 71 17
Sorted Array in Ascending Order using BubbleSort:
17  25  45  50  71  87
$
```

Changes & Additions Made in XV6

- The 3 user programs for the sort and search algorithms were created, written in the C programming language and saved inside the XV6 OS folder as:
  - linearS.c
  - selectionsort.c
  - bubblesort.c

- The files containing the C programs had to contain the following header files:
  - include “types.h”
  - include “stat.h”
  - include “user.h”

Changes & Additions Made in XV6

- The makefile in the XV6 folder was edited in order to add them to the source code of XV6 and be able to compile the programs.
- The names of the files containing the C programs were added to the following 2 sections:

```
UPROGS=\
        _cat\
        _echo\
        _forktest\
        _grep\
        _init\
        _kill\
        _ln\
        _ls\
        _mkdir\
        _rm\
        _sh\
        _stressfs\
        _usertests\
        _wc\
        _zombie\
        _linearS\
        _selectionsort\
        _bubblesort\
```

```
EXTRA=\
        mkfs.c ulib.c user.h cat.c echo.c forktest.c grep.c kill.c\
        ln.c ls.c mkdir.c rm.c stressfs.c usertests.c wc.c zombie.c \
        linearS.c selectionsort.c bubblesort.c\
        README dot-bochsrc *.pl toc.* runoff runoff1 runoff.list\
        .gdbinit.tmpl gdbutil\
```

Changes & Additions Made in XV6

- Created a function *int atoi2(char *s)* and added it to the user.h and ulib.c files in order to handle negative numbers in the user programs.

Open *user.h ~/xv6

```c
struct stat;
struct rtcdate;

// system calls
int fork(void);
int exit(void) __attribute__((noreturn));
int wait(void);
int pipe(int*);
int write(int, const void*, int);
int read(int, void*, int);
int close(int);
int kill(int);
int exec(char*, char**);
int open(const char*, int);
int mknod(const char*, short, short);
int unlink(const char*);
int fstat(int fd, struct stat*);
int link(const char*, const char*);
int mkdir(const char*);
int chdir(const char*);
int dup(int);
int getpid(void);
char* sbrk(int);
int sleep(int);
int uptime(void);

// ulib.c
int stat(const char*, struct stat*);
char* strcpy(char*, const char*);
void *memmove(void*, const void*, int);
char* strchr(const char*, char c);
int strcmp(const char*, const char*);
void printf(int, const char*, ...);
char* gets(char*, int max);
uint strlen(const char*);
void* memset(void*, int, uint);
void* malloc(uint);
void free(void*);
int atoi(const char*);
int atoi2(char *s);
```

Open ulib.c ~/xv6

```c

  fd = open(n, O_RDONLY);
  if(fd < 0)
    return -1;
  r = fstat(fd, st);
  close(fd);
  return r;
}

int
atoi(const char *s)
{
  int n;

  n = 0;
  while('0' <= *s && *s <= '9')
    n = n*10 + *s++ - '0';
  return n;
}

void*
memmove(void *vdst, const void *vsrc, int n)
{
  char *dst;
  const char *src;

  dst = vdst;
  src = vsrc;
  while(n-- > 0)
    *dst++ = *src++;
  return vdst;
}

int
atoi2(char *s)
{
  int n;

  n = 0;
  while('0' <= *s && *s <= '9')
    n = n*10 + *s++ - '0';
  return n;
}
```