CSc 3320: Systems Programming

Spring 2021

Midterm 1: Total points = 100

Assigned: 26th Feb 2021: 12.01 PM

Submission Deadline: 2nd Mar 2021: 12.01 PM

(No extensions. If your submission is not received by this time then it will NOT be accepted.)

Submission instructions:

1. Create a Google doc for your submission.
2. Start your responses from page 2 of the document and copy these instructions on page 1.
3. Fill in your name, campus ID and panther # in the fields provided. If this information is missing TWO POINTS WILL BE DEDUCTED.
4. Keep this page 1 intact. If this *submissions instructions* page is missing in your submission TWO POINTS WILL BE DEDUCTED.
5. Start your responses to each QUESTION on a new page.
6. If you are being asked to write code copy the code into a separate txt file and submit that as well. The code should be executable. E.g. if asked for a C script then provide myfile.c so that we can execute that script. In your answer to the specific question, provide the steps on how to execute your file (like a ReadMe).
7. If you are being asked to test code or run specific commands or scripts, provide the evidence of your outputs through a screenshot and/or screen video-recordings and copy the same into the document.
8. Upon completion, download a .PDF version of the google doc document and submit the same along with all the supplementary files (videos, pictures, scripts etc).

Full Name: Michael Anderson

Campus ID: manderson113

Panther #: 002485269

Questions 1-5 are 20pts each

Question 1

```
#!/bin/bash
# README: enter the command in lowercase
#         (F)orward, (B)ack, (Q)uit
indexes=`egrep -n "^[A-Z]+\(1\)" mandatabase.txt | sed -n
"s/\([0-9]\+\):\([A-Z]\+\).*/\2 \1/p" | tr '[A-Z]' '[a-z]'`
read -p "Type in a command to see it's manual: " command
pattern=".*[0-9]*\s*$command\s([0-9]+)\s*[a-z]*\s*([0-9]*)"
if [[ $indexes =~ $pattern ]]
then
  firstLine=${BASH_REMATCH[1]}
  lastLine=${BASH_REMATCH[2]}
  ((lastLine--))
  head -n$lastLine mandatabase.txt | tail -n+$firstLine |
less
else
  echo "sorry, I cannot help you"
fi
```

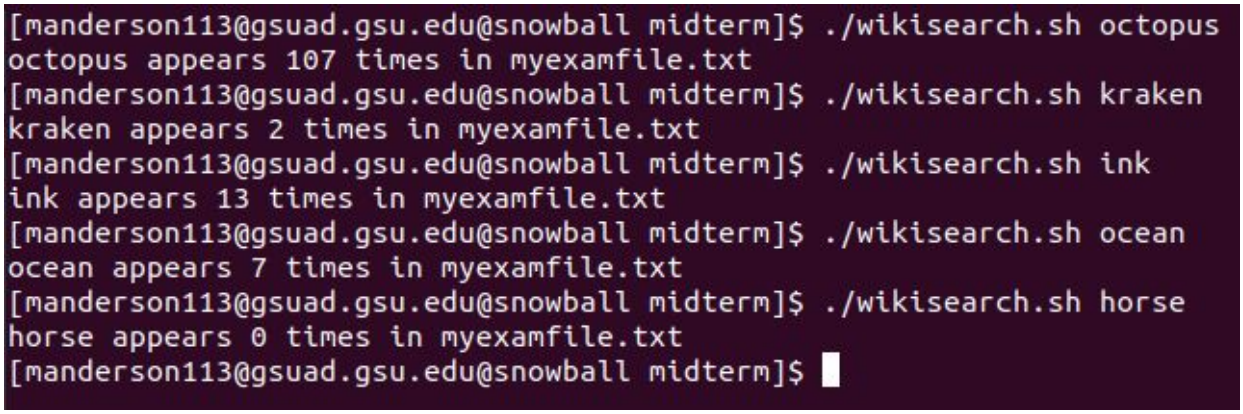
```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./helpme.sh
Type in a command to see it's manual: ls
```


```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./helpme.sh
Type in a command to see it's manual: ls
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./helpme.sh
Type in a command to see it's manual: ez
sorry, I cannot help you
[manderson113@gsuad.gsu.edu@snowball midterm]$
```

Question 2

```
#!/bin/sh
count=`grep -iow $1 myexamfile.txt | wc -l`
printf "%s appears %d times in myexamfile.txt\n" $1 $count
```

The first argument is the word you wish to search for.



Question 3

```
#!/bin/sh
# README: compressor -r directory fullDaysSinceAccessed
#         -r is optional, makes it recursively apply
#         directory . = current directory
#         directory ~ = root directory
#         0 days means all files will be compressed
#         will not attempt to compress .gz files
#         will not compress files with this script's name

myName=`basename $0`

depth="-maxdepth 1"
if [[ $1 == '-r' ]]
then
  depth=""
  shift
fi

days=`expr $2 - 1`

if [[ -z $1 ]]
then
  echo "Please enter a valid directory"
  exit 128
fi

if [[ -z $2 || $2 -lt 0 ]]
then
  echo "Please enter a valid number of days"
  exit 128
fi

#fileList=`find $1 $depth -type f -atime +$days | grep -v '\.gz$'`
fileList=`find $1 $depth -type f -atime +$days | grep -v '\.gz$' |
grep -v $myName`

for file in $fileList
do
  echo "Compressing" $file
  gzip $file
done
```

```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./compressor.sh ./testing 0
Compressing ./testing/cake.txt
[manderson113@gsuad.gsu.edu@snowball midterm]$
```

Question 4

```
#!/bin/bash

# README: Command List
#         phoneNumber format: 999-999-9999, optional '-'
#         (a)dd firstName lastName phoneNumber address
#         (r)emove phoneNumber
#         (u)pdate phoneNumber
#         (f)ind searchText
#         (i)mport filename.csv
#         (h)elp

validName="^[A-Z][-'A-z]+$"
validNumber="^\(?([0-9]{3})\)?-?([0-9]{3})-?([0-9]{4})$"
validJustNumbers="^[0-9]{10}$"

validate() {
 if [[ !$firstName =~ $validName || !$lastName =~ $validName ||
!$phoneNumber =~ $validJustNumbers ]]
 then
  echo 'invalid input'
  return 1
 fi
 if !(checkDuplicate)
 then
  echo 'duplicate number, discarding input'
  return 1
 fi
 return 0
}

checkDuplicate() {
 duplicate=$(grep $phoneNumber phonebook)
 if [[ -n $duplicate ]]
 then
  return 1
 fi
 return 0
}

SORTED=true

action=$1; shift

if [[ $action =~ ^a ]]
then # add
 firstName=$1; shift
 lastName=$1; shift
 phoneNumber="${1//[!0-9]/}"; shift
 address="$*"
```

```
 if validate
 then
  printf 'adding:\n'
  printf "%s %s %s  %s\n" $firstName $lastName $phoneNumber "$address"
  printf "%-16s %-16s %s  %s\n" $firstName $lastName $phoneNumber
"$address" >> phonebook
 SORTED=false
 else
  echo 'add failed'
 fi
elif [[ $action =~ ^r ]]
then # remove
 phoneNumber="${1//[!0-9]/}"; shift
 if [[ $phoneNumber =~ $validNumber ]]
 then
  lineToRemove=$(grep -w $phoneNumber phonebook)
  if [[ -n $lineToRemove ]]
  then
   echo "Removing:"
   echo $lineToRemove
   sed -i "/$phoneNumber/d" phonebook
  else
   echo "Number not in phonebook"
  fi
 fi
elif [[ $action =~ ^u ]]
then # update
 phoneNumber="${1//[!0-9]/}"; shift
 if [[ !$phoneNumber =~ $validNumber ]]
 then
  echo "invalid phone number"
 else
  match=$(grep -w $phoneNumber phonebook)
  lineCount=$(echo "$match" | wc -l)
  if [[ $lineCount=="1" ]]
  then
   read -p "Enter their first name: " firstName
   read -p "Enter their last name: " lastName
   read -p "Enter their address: " address
   sed -i "/$phoneNumber/d" phonebook
   echo "updated:"
   printf "%s %s %s  %s\n" $firstName $lastName $phoneNumber "$address"
   printf "%-16s %-16s %s  %s\n" $firstName $lastName $phoneNumber
"$address" >> phonebook
   SORTED=false
  else
   echo "Number not in phonebook"
  fi
 fi
elif [[ $action =~ ^f ]]
then # find
 search="$*"
```

```
 if [[ $search =~ $validNumber ]]
 then
  search="${search//[!0-9]/}"
 else
  search="${search// / \{1,\}}"
 fi
 results=$(grep -i "$search" phonebook)
 if [[ -n $results ]]
 then
  echo "$results"
 else
  echo "No Results"
 fi
elif [[ $action =~ ^i ]]
then
 fileName=$1; shift
 cat $fileName | while read line
 do
  entryMatch="^(.*),(.*),([0-9]{3})-([0-9]{3})-([0-9]{4}),\"(.*)\"$"
  if [[ $line =~ $entryMatch ]]
  then
   firstName=${BASH_REMATCH[1]}
   lastName=${BASH_REMATCH[2]}
   pA=${BASH_REMATCH[3]}
   pB=${BASH_REMATCH[4]}
   pC=${BASH_REMATCH[5]}
   phoneNumber="$pA$pB$pC"
   address=${BASH_REMATCH[6]}
   exists=$(grep $phoneNumber phonebook)
   if [[ -z $exists ]]
   then
    printf "%-16s %-16s %s  " $firstName $lastName $phoneNumber >>
phonebook
    echo $address >> phonebook
   fi
  fi
 done
 SORTED=false
elif [[ $action == "" || $action =~ "^h" ]]
then # help
 echo "(a)dd firstName lastName 999-999-9999 address"
 echo "(r)emove 999-999-9999"
 echo "(u)pdate 999-999-9999"
 echo "(f)ind searchText"
 echo "(i)mport filename.csv"
else # invalid command
 echo "invalid command"
fi
if [[ $SORTED == false ]]
then
 sort -o phonebook -k2b,2 -k1,1 phonebook
fi
```

```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./bookmanager.sh
(a)dd firstName lastName 999-999-9999 address
(r)emove 999-999-9999
(u)pdate 999-999-9999
(f)ind searchText
(i)mport filename.csv
```

```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./bookmanager.sh a Harvey Dent 2
22-222-2222 Justice Ln
adding:
Harvey Dent 2222222222  Justice Ln
```

```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./bookmanager.sh f Harvey
Harvey          Dent            2222222222  Justice Ln
Helen           Harvey          2177540995  2426 Scenic Way
```

```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./bookmanager.sh u 222-222-2222
Enter their first name: Two-Face
Enter their last name: Dent
Enter their address: Injustice Ln
updated:
Two-Face Dent 2222222222  Injustice Ln
```

```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./bookmanager.sh r 222-222-2222
Removing:
Two-Face Dent 2222222222 Injustice Ln
```

```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ls -l phonebook
-rwx------. 1 manderson113@gsuad.gsu.edu manderson113@gsuad.gsu.edu 65432 Feb 2
8 17:12 phonebook
```

Question 5: A, Factorial

```
#include <stdio.h>
unsigned long factorial(int num)
{
  if (num <= 1)
  {
    return 1;
  }
  return num*factorial(num-1);
}
int main(int argc, char *argv[])
{
  int num = atoi(argv[1]);
  Num = (num < 0) ? 0 : num;
  unsigned long fac = factorial(num);
  printf("%d! = %ld\n", num, fac);
  return fac;
}
```

```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./factorial 0
0! = 1
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./factorial 5
5! = 120
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./factorial 11
11! = 39916800
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./factorial 42
42! = 7538058755741581312
```

Question 5: B, Shift & Complement

```
#include <stdio.h>
int main(int argc, char *argv[])
{
  int num = atoi(argv[1]);
  int result = (num<<3) + ~num;
  printf("%d<<3 = %d\n", num, num<<3);
  printf("~%d = %d\n", num, ~num);
  printf("(%d<<3)+~%d = %d\n", num, num, result);
  return result;
}
```

```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./shiftComplement 0
0<<3 = 0
~0 = -1
(0<<3)+~0 = -1
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./shiftComplement 5
5<<3 = 40
~5 = -6
(5<<3)+~5 = 34
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./shiftComplement -3
-3<<3 = -24
~-3 = 2
(-3<<3)+~-3 = -22
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./shiftComplement 42
42<<3 = 336
~42 = -43
(42<<3)+~42 = 293
```

Question 5: Bonus 1, Including Binary <-> Int

```
#include <stdio.h>
#include <math.h>

#define true 1
#define false 0

int main(int argc, char *argv[])
{
  char *digits = argv[1];
  int num = atoi(digits);
  int result = (num<<3) + ~num;
  printf("(%d<<3) + ~(%d) = %d\n", num, num, result);

  size_t size = strlen(digits);
  int isBinary = true;
  int negative = (num < 0);
  int binToDec=0;
  int power=0;
  int i;
  int start = negative ? 1 : 0;
  for (i=strlen(digits)-1; i>=start && isBinary; i--)
  {
    char digit = digits[i];
    if (digit == '1')
    {
      binToDec += (int) pow(2,power);
    }
    else if (digit != '0')
    {
      isBinary = false;
    }
    power++;
  }
```

```
  if (isBinary)
  {
    printf("%db = %s%dd\n",
      num,
      negative ? "-" : "",
      binToDec);
  }

  power=0;
  int* decToBin[32];
  for(i=31; i>=0; i--)
  {
    decToBin[31-i]=((num>>i) & 1);
  }
  i=0;
  while (decToBin[i]==0)
  {
    i++;
  }
  printf("%dd = ", num);
  while (i<32)
  {
    printf("%d",decToBin[i]);
    i++;
  }
  printf("b\n");

  return 0;
}
```

```
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./processNumber 101
(101<<3) + ~(101) = 706
101b = 5d
101d = 1100101b
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./processNumber -1101
(-1101<<3) + ~(-1101) = -7708
-1101b = -13d
-1101d = 11111111111111111111101110110011b
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./processNumber 23
(23<<3) + ~(23) = 160
23d = 10111b
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./processNumber -2147483648
(-2147483648<<3) + ~(-2147483648) = 2147483647
-2147483648d = 10000000000000000000000000000000b
[manderson113@gsuad.gsu.edu@snowball midterm]$ ./processNumber 2147483647
(2147483647<<3) + ~(2147483647) = 2147483640
2147483647d = 11111111111111111111111111111111b
```

Question 5: Bonus 2, Executing via Shell Script

```
#/bin/bash
read -p "Please enter a number: " num
./factorial $num
./processNumber $num
```