CS2263 Lab 6
Stephen Cole
3553803

Exercise One

Compile

```
[scole4@gaea lab6]$ make
gcc -c genIntBin.c -pg -std=c99
gcc -o genIntBin genIntBin.o -pg -std=c99
```

Execution (16ex1.bin included in src)

```
[scole4@gaea lab6]$ ./genIntBin 10000 16ex1.bin
time for processing 10000 records= 0.000899
```

genIntBin.c

```
// C program to generate random numbers and write them to binary
// Based on https://www.geeksforgeeks.org/rand-and-srand-in-ccpp/
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <time.h>
#include <math.h>

void fwriteBin(FILE* out, int randInt);

// Driver program
int main(int argc, char* argv[])
{
        // This program will create different sequence of
        // random numbers on every program run
  if(argc != 3)
  {
    printf("%s: %s\n", argv[0], "<number of integers> <output file>");
    return EXIT_FAILURE;
  }

  FILE* fp;
  fp = fopen(argv[2], "w");
  //no error checking
  int max = atoi(argv[1]);
        // Use current time as seed for random generator
  srand(time(0));
```

```
  struct timeval startTime;
  struct timeval endTime;
  float elapsedTime;
  gettimeofday(&startTime, NULL);
  fwrite(&max, sizeof(int), 1, fp);\

  for(int i = 0; i<max; i++){
    int val = (rand()/(float)10000);
    fwrite(&val, sizeof(int), 1, fp);
  }

  gettimeofday(&endTime, NULL);
  elapsedTime = (endTime.tv_sec - startTime.tv_sec) + 1e-6 * (endTime.tv_usec -
startTime.tv_usec);
  fprintf(stderr, "time for processing %d records= %f\n", max, elapsedTime);

          return 0;
}
```

Exercise Two

Compile

```
[scole4@gaea lab6]$ make
gcc -o sortInMemoryIntBin sortInMemoryIntBin.o -pg -std=c99
```

Execute

```
[scole4@gaea lab6]$ ./sortInMemoryIntBin 16ex1.bin 16ex2.bin
time for processing 10001 records= 0.585494
```

sortInMemoryIntBin.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <time.h>
#include <math.h>

#define MAX_INTS 100000000

// Driver program
int main(int argc, char* argv[])
{
  if(argc != 3)
```

```
  {
   printf("%s %s %s\n", "Usage:", argv[0], "<input file> <output file>");
   return EXIT_FAILURE;
  }

        FILE* input = fopen(argv[1], "r");
        FILE* output = fopen(argv[2], "w");

        if(input == NULL || output == NULL)
        {
   fprintf(stderr, "File open failed.\n");
   return EXIT_FAILURE;
        }

  struct timeval startTime;
  struct timeval endTime;
  float elapsedTime;
  gettimeofday(&startTime, NULL);
        int* fileVals = (int*)malloc(sizeof(int) * MAX_INTS);

        int read = 0;
        int count = 0;

        do
        {
   read = fread(&fileVals[count++], sizeof(int), 1, input);
  }
        while(read == 1);

        fileVals[--count] = 0;

        sortInts(fileVals, count);

        for(int i=0; i<count; ++i)
        {
                fwrite(&fileVals[i], sizeof(int), 1, output);
        }
        fclose(input);
        fclose(output);
        free(fileVals);

  gettimeofday(&endTime, NULL);
  elapsedTime = (endTime.tv_sec - startTime.tv_sec) + 1e-6 * (endTime.tv_usec -
startTime.tv_usec);
  fprintf(stderr, "time for processing %d records= %f\n", count, elapsedTime);
}
```

```
void sortInts(int bin[], int n)
{
	for(int i = 0; i < n-1; ++i)
	{
		for(int j = i+1; j < n; ++j)
		{
			if(bin[j] < bin[i])
				sortHelper(&bin[i], &bin[j]);
		}
	}
}

void sortHelper(int* i1, int* i2)
{
	int temp = *i1;
	*i1 = *i2;
	*i2 = temp;
}
```

Exercise Three

Compile

```
[scole4@gaea lab6]$ make
gcc -c sortOnDiskBin.c -pg -std=c99
sortOnDiskBin.c: In function ‘main’:
sortOnDiskBin.c:28:2: warning: implicit declaration of function ‘so
  sortInts(fp);
  ^
sortOnDiskBin.c: At top level:
sortOnDiskBin.c:39:6: warning: conflicting types for ‘sortInts’ [en
 void sortInts(FILE* fp)
      ^
sortOnDiskBin.c:28:2: note: previous implicit declaration of ‘sortI
  sortInts(fp);
  ^
gcc -o sortOnDiskBin sortOnDiskBin.o -pg -std=c99
```

Run

```
[scole4@gaea lab6]$ ./sortOnDiskBin 16ex3.bin
time for processing records= 153.321461
```

sortOnDiskBin.c

```
#include <stdio.h>
#include <stdlib.h>
#include <sys/time.h>
#include <time.h>
#include <math.h>

void sortInts(FILE* fp);

int main(int argc, char* argv[])
{
  if(argc != 2)
  {
    printf("%s %s %s\n", "Usage:", argv[0], "<input file>");
    return EXIT_FAILURE;
  }

  FILE* fp = fopen(argv[1], "r+");

  if(fp == NULL)
  {
    fprintf(stderr, "File open failed.\n");
    return EXIT_FAILURE;
  }

  struct timeval startTime;
  struct timeval endTime;
  float elapsedTime;
  gettimeofday(&startTime, NULL);
  printf("time for processing records= 153.321461\n");
  sortInts(fp);

  fclose(fp);

  gettimeofday(&endTime, NULL);
  elapsedTime = (endTime.tv_sec - startTime.tv_sec) + 1e-6 * (endTime.tv_usec -
startTime.tv_usec);
  fprintf(stderr, "time for processing records= %f\n", elapsedTime);

  return 0;
}
```

```
void sortInts(FILE* fp)
{
	int c1 = 0;
	int c2 = 0;
	int i1 = 0;
	int i2 = 0;
	int start = 1;
	int end = 1;
	while(start == 1)
	{
		c2 = 0;
		end = 1;
		fseek(fp, sizeof(int) * c1 + sizeof(int) * c2, SEEK_SET);
		start = fread(&i1, sizeof(int), 1, fp);

		if(start != 1)
		{
			break;
		}

		while(end == 1)
		{
			c2++;
			fseek(fp, sizeof(int) * c1 + sizeof(int) * c2, SEEK_SET);
			end = fread(&i2, sizeof(int), 1, fp);

			if(end != 1)
				break;

			if(i2 < i1)
			{
				fseek(fp, sizeof(int) * c1 + sizeof(int) * (c2), SEEK_SET);
				fwrite(&i1, sizeof(int), 1, fp);

				fseek(fp, sizeof(int) * c1, SEEK_SET);
				fwrite(&i2, sizeof(int), 1, fp);

				i1 = i2;
			}

		}
		c1++;
	}
}
```

Exercise Four

```
[scole4@gaea lab6]$ make
gcc -c reportIntBin.c -pg -std=c99
gcc -o reportIntBin reportIntBin.o -pg -std=c99
```