CS2263 Lab 6

Stephen Cole

3553803

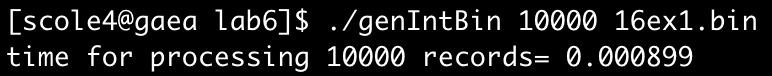
Exercise One

Compile

A close up of a sign

Description automatically generated

Execution (16ex1.bin included in src)



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;

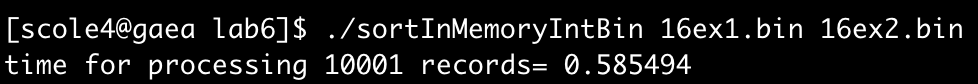
}

Exercise Two

Compile



Execute



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;

}

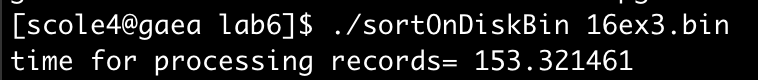
Exercise Three

Compile

A screenshot of a cell phone screen with text

Description automatically generated

Run



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

A screenshot of a cell phone

Description automatically generated