Christiaan Cronje

Lab 3 Option A

COSC 2347 8AM

Due: April 7, 2017

Submitted: April 7, 2017

**Makefile**

FILENAME=EngineeringFunctionTest

all: independent share link path

independent:

gcc -fPIC -c engeom.c

share:

gcc -shared -o libEngeom.so engeom.o -lm

link:

gcc EngineeringFunctions.c -L. -lEngeom -o $(FILENAME) -ldl

path:

export LD\_LIBRARY\_PATH=./

clean:

rm -rf \*.o \*.so $(FILENAME)

run: all

./$(FILENAME)

**EngineeringFunctions.c**

#include <dlfcn.h>

#include "engeom.h"

#include <stdio.h>

#include <stdlib.h>

#include <ctype.h>

int main()

{

void \*handle;

char \*err;

float srad, crad, cht;

FILE \*outFile;

char outFileName[21];

char response;

//Define DLL pointers

void (\*eng\_sphere\_d)(float, float\*, float\*);

float (\*eng\_volCylinder\_d)(float, float);

float (\*eng\_sumFloats\_d)(float\*, int);

double (\*eng\_sin\_d)(float);

//Get handle to dynamic library

handle = dlopen("./libEngeom.so", RTLD\_LAZY);

if (handle == (void \*)0)

{

fputs(dlerror(), stderr);

exit(-1);

}

//Check if pointers exist

eng\_sphere\_d = dlsym(handle, "eng\_sphere");

err = dlerror();

if (err != NULL)

{

fputs(err, stderr);

exit(-1);

}

eng\_volCylinder\_d = dlsym(handle, "eng\_volCylinder");

err = dlerror();

if (err != NULL)

{

fputs(err, stderr);

exit(-1);

}

eng\_sumFloats\_d = dlsym(handle, "eng\_sumFloats");

err = dlerror();

if (err != NULL)

{

fputs(err, stderr);

exit(-1);

}

eng\_sin\_d = dlsym(handle, "eng\_sin");

err = dlerror();

if (err != NULL)

{

fputs(err, stderr);

exit(-1);

}

do

{

printf("Enter output file name: ");

scanf("%s", outFileName);

//check if file already exists and asks for user authorizations

if((outFile = fopen(outFileName, "r")) != NULL)

{

printf("A file by that name already exists.\n");

do

{

printf("Do you wish to override? [y/n]: ");

scanf(" %c", &response);

}

while(toupper(response) != 'N' && toupper(response) != 'Y');

}

}

while(toupper(response) == 'N');

//creates output file if it doesnt exist, or overwrite

outFile = fopen(outFileName, "w");

if (outFile == NULL)

{

printf("File could not be created.\n");

printf("Exiting Program!\n");

exit(1);

}

//sphere radius

printf("Enter the radius of a sphere: ");

scanf("%f", &srad);

float s\_surface, s\_volume;

(\*eng\_sphere\_d)(srad, &s\_surface, &s\_volume);

printf("The surface area is: %f\n", s\_surface);

printf("The Volume is: %f\n", s\_volume);

fprintf(outFile, "The surface area of a %.3fcm radius sphere is: %fcm\n", srad, s\_surface);

fprintf(outFile, "The volume of a %.3fcm radius sphere is: %fcm\n\n", srad, s\_volume);

//cylinder radius

printf("Enter the radius of a cylinder: ");

scanf("%f", &crad);

printf("Enter the height of a cylinder: ");

scanf("%f", &cht);

float c\_volume = (\*eng\_volCylinder\_d)(crad, cht);

printf("The volume of a cylinder is: %f\n", c\_volume);

fprintf(outFile, "The volume of a %.2fft x %.2fft cylinder is: %fft\n\n", crad, cht, c\_volume);

//sum of array

int arrSize = 0;

printf("Enter the size of a float array: ");

scanf("%d", &arrSize);

float farray[arrSize];

for (int i = 0; i < arrSize; i++)

{

printf("Enter the size of %dst float array: ", i+1);

scanf("%f", &farray[i]);

}

//display entered data

printf("Unsorted Data:\n");

fprintf(outFile, "Unsorted Data:\n");

for (int i = 0; i < arrSize; i++)

{

printf("%f\n", farray[i]);

fprintf(outFile, "%f\n", farray[i]);

}

//sort, add, and display data

float fsum = (\*eng\_sumFloats\_d)(farray, arrSize);

printf("Sorted Data:\n");

fprintf(outFile, "\nSorted Data:\n");

for (int i = 0; i < arrSize; i++)

{

printf("%f\n", farray[i]);

fprintf(outFile, "%f\n", farray[i]);

}

printf("The sum of the float array is: %f\n\n", fsum);

fprintf(outFile, "The sum of the float array is: %f\n\n", fsum);

//get the sine of some angles

int sinCount = 0;

printf("Enter the number of angles: ");

scanf("%d", &sinCount);

double sinResult;

float sinOf;

for (int i = 0; i < sinCount; i++)

{

printf("Enter the size of %dst angle: ", i+1);

scanf("%f", &sinOf);

sinResult = (\*eng\_sin\_d)(sinOf);

printf("sin(%f) = %lf\n", sinOf, sinResult);

fprintf(outFile, "sin(%f) = %lf\n", sinOf, sinResult);

}

dlclose(handle);

return 0;

}

**engeom.h**

#ifndef \_\_ENGEOM\_H

#define \_\_ENGEOM\_H

extern void eng\_sphere(float radius, float \*surface, float \*volume);

extern float eng\_volCylinder(float radius, float height);

extern float eng\_sumFloats(float x[], int numFloats);

extern double eng\_sin(float angle);

#endif

**engeom.c**

#include <stdio.h>

#define PI 3.14159265358979323846

void eng\_sphere(float radius, float \*surface, float \*volume)

{

\*surface = 4.0 \* PI \* radius \* radius;

\*volume = (4.0 / 3.0) \* PI \* radius \* radius \* radius;

return;

}

float eng\_volCylinder(float radius, float height)

{

float volume = PI \* radius \* radius \* height;

return volume;

}

float eng\_sumFloats(float x[], int numFloats)

{

int min;

float tmp;

for (int i = 0; i < numFloats - 1; i++)

{

min = i;

for (int j = i + 1; j < numFloats; j++)

{

if (x[j] < x[min])

{

min = j;

}

}

if (min != i)

{

tmp = x[i];

x[i] = x[min];

x[min] = tmp;

}

}

float sum = 0.0;

for (int i = 0; i < numFloats; i++)

{

sum += x[i];

}

return sum;

}

int factorial(int n);

double powr(double base, int exponent);

double eng\_sin(float angle)

{

double EPSILON = 0.0000001;

int i = 1, sign = 1;

double result, rad;

rad = angle / 180.0 \* PI;

while((powr(rad, i) / factorial(i)) > EPSILON)

{

result += sign \* (powr(rad, i) / factorial(i));

i += 2;

sign \*= -1;

}

return result;

}

int factorial(int n)

{

int fact = 1;

for (int i = n; i > 1; i--)

{

fact \*= i;

}

return fact;

}

double powr(double base, int exponent)

{

double total = 1;

for (int i = 0; i < exponent; i++)

{

total \*= base;

}

return total;

}

The surface area of a 17.583cm radius sphere is: 3885.042969cm

The volume of a 17.583cm radius sphere is: 22770.236328cm

The volume of a 13.35ft x 3.71ft cylinder is: 2077.238281ft

Unsorted Data:

0.347679

100400.796875

0.000010

1.343567

78.345673

1678.567017

997999.625000

0.000001

Sorted Data:

0.000001

0.000010

0.347679

1.343567

78.345673

1678.567017

100400.796875

997999.625000

The sum of the float array is: 1100159.000000

sin(45.000000) = 0.707107

sin(36.000000) = 0.587785

sin(0.345000) = 0.006021