**BIL 105E – Introduction to Scientific and Engineering Computing (C)**

**Spring 2015-2016**

**Homework 4**

**CRN:21834**

**Yunus Güngör**

**No:150150701**

**Date:09.05.2016**

**Introduction**

This project’s main aim is to understand file processes in C language by simulating a hospital’s patient database on Body Mass Index. A database in height\_weight.txt which has the patients height and weight used to calculate and create a database in output.txt on body mass index of patients.

**Development Environment**

This program has only 1 source code file written in C:

150150701.c

This program tested and compiled in following system:

gcc 4.8.5 20150623 on Red Hat 4.8.5-4 (ITU SSH Server)

gcc compiler has been used to compile the program by the command:

gcc 150150701.c -o hw4

**Important Variables and Functions**

**person struct:** has data of patients such as height, weight and BMI

**readfile:** This function creates a person array named people and saves data in it from height\_weight.txt. It returns the address of people.

**getPatients:** This function creates the output.txt file by desired specifications.

**sort:** this functions sorts a nx2 matrix by second column. Usually first column used for the ids of people and second column used for the data to sort.

**N:** Total number of people. This value is variable, so you have to allocate enough space at

run time.

**M:** Number of people whose BMI values are the farthest from the threshold.

**threshold:** Normal value of the BMI value.

**People:** An array of person structure

**thresDif:** A 2 column matrix for personID and the difference between BMI and threshold by each person.

**patients: :** A 2 column matrix for personID and BMI by each person.

**Program Flow**

Pseudo code of the program:

struct person {int personID; double height; double weight; double BMI;}

int main(N,M,threshold) {

initialize people as person array;

people=readFile("./height\_weight.txt",N);

if (people==NULL)

{

print("Error\n");

return -1;

}

getPatients(people,N,M,threshold,"./output.txt");

return 0;

}

person\* readFile(file,N)

{

Allocate N\*sizeof(person) bytes in memory for people;

open file as height\_weight;

Allocate 34\*sizeof(char) bytes in memory for firstline;

get first 34 characters from height\_weight to firstline;

free the space of firstLine from memory;

for (i=0;i<N;i++)

{

Get next character from height\_weight as tmp;

personID=0;

while('0'>=tmp AND tmp>'9' AND tmp!=EOF)

Get next character from height\_weight as tmp;

while('0'<=tmp AND tmp<='9')

{

Add tmp as new digit to personID;

Get next character from height\_weight as tmp;

}

height=0;

Get next character from height\_weight as tmp;

while('0'<=tmp AND tmp<='9')

{

Add tmp as new digit to height;

Get next character from height\_weight as tmp;

}

weight=0;

Get next character from height\_weight as tmp;

while('0'<=tmp AND tmp<='9')

{

Add tmp as new digit to weight;

Get next character from height\_weight as tmp;

}

people[personID-1].personID=personID;

people[personID-1].weight=weight;

people[personID-1].height=height;

people[personID-1].BMI=weight/(height in meter)^2;

Get next character from height\_weight;

}

close height\_weight;

return people;

}

void getPatients(people,N,M,threshold,file)

{

for(i=0;i<N;i++)

{

thresDif[i][0]=people[i].personID;

t=threshold-people[i].BMI;

t=absolute of t;

thresDif[i][1]=t;

}

sort(thresDif,N);

for (i=0;i<M;i++)

{

personID=thresDif[i][0];

patients[i][0]=personID;

patients[i][1]=people[personID-1].BMI;

}

sort(patients,M);

open file as output;

print to output("Person\_id\tHeight(cm)\tWeight(kg)\tBMI\n");

for (i=0;i<M;i++)

{

j=(int)patients[i][0]-1;//get indicator for people

print to output( "%d\t%3.f\t%2.f\t%2.2f\n",people[j].personID,people[j].height,people[j].weight,people[j].BMI);

}

Close output;

}

}

void sort(array[][2], n)

{

for (i=0;i<(n-1);i++)

{

for(j=0;j<n-i-1;j++)

{

if (array[j][1]<array[j+1][1])

{

tmp=array[j][1];

array[j][1]=array[j+1][1];

array[j+1][1]=tmp;

tmp=array[j][0];

array[j][0]=array[j+1][0];

array[j+1][0]=tmp;

}

}

}

}

**Conclusion**

This project helped me to understand file processes and basic databases. Besides that, I also used subjects like memory allegation, sorting algorithms and functions for better performance and creating better structure for the project