**DATA STRUCTURES AND ALGORITHM**

**CSE2003**

**TOPIC: TYPING TUTOR**

**Abhishek Mukherjee (19BCE0598)**

**Shagun Mishra (19BCE0565)**

**Abstract**

Typing Tutor is specially designed to improve the user’s typing skills in speed and accuracy. This system works on the concept of storing and retrieving the user’s typing data whenever required. The whole project is designed in ‘C++’ language and different variables and data structures (eg. Linked List using structure ‘LL’) , have been used for the development of this project. It’s easy to operate and to be understood by users. The user interface is very simple and user will have no trouble to navigate.

**Features:**

1. Records user’s speed and accuracy of every attempt
2. Proper allocation of memory for new users
3. Levels allocated for user’s interest
4. Easy to operate

**Introduction**

These days, typing is a very important skill for it’ s use in our day to day lives. Typing is still one of the most important computer skills you can learn. Learning to type fast and accurately helps us in many ways in life, and it should be considered an essential skill for anyone working on a computer.

Few of the reasons why typing fast is beneficial

1. Saves Time
2. Makes More Productive
3. Improves Your Posture

By this Project, we are facilitating the effective use of programming in improving typing skills.

**Motivation**

As college students, we spend a lot of time working on computers and most of the time we are in a hurry as time is of the essence. Having good typing skills helps in a big way. This led our group into making this Typing Tutor program for benefitting the students into improving their typing skills.

**Modules:**

In FileCreator.cpp:

Void Create() : To create files to store user’s attempt

In LLops.cpp:

LL \*Delete() : To delete from linked list

LL \*Insert() : To insert in linked list

LL \*search() : To search a record in linked list

In Driver.cpp:

LL \*whereItHappens() : calculating speed and accuracy

int SubMenu() : displaying sub menu

LL \*SearchLeast()

Void mainProgram() : coordinating all functions for use

Void newUser()

Void oldUser()

Void viewRecords()

Int menu()

Int main()

In Prerequisites.h() :

Struct LL() :

Other main program definitions.

**Algorithm:**

Start

Displays this menu:

1. New user

2. Old user

3. Read records

4. Exit

User enters username

User selects an option from the main menu

If user selects 1.new user, new record created for the username

If user selects 2.old user, the user’s records are only accessed for the username

So after user types name and presses enter

New menu opens

1. Beginner

2. Intermediate

3. Expert

4. Go Back

User types 1 and presses enter

Control then opens Beginner.txt, copies 1 line, and displays it

User writes that line and presses enter

Control then calculates time, accuracy and stores it in the linked list, then goes back to this menu

When user presses 4, control runs through the LL, finds the smallest time for a particular group, say beginner, and stores it in Previous Games file

**Code:**

**Driver.cpp:**

#include<fstream>

#include<string>

#include"Prerequisites.h"

LL whereItHappens(string line)

{

int c = 0;

LL node;

cout<<"Start writing\n\n";

cout<<line<<endl;

clock\_t start = clock();

string input;

cin.ignore();

getline(cin,input);

clock\_t en = clock();

for(string::iterator i = line.begin(), j = input.begin(); j != input.end(); i++, j++)

{

if(\*i != \*j)

c++;

}

node.acc = 1.0 - ((double)c)/line.length();

node.time = ((double)(en - start))/CLOCKS\_PER\_SEC;

cout<<"Time : "<<node.time<<endl;

cout<<"Accuracy : "<<node.acc<<endl;

Sleep(5000);

return node;

}

int SubMenu()

{

int ch;

cout<<"Enter\n1. Beginner\n2. Intermediate\n3. Expert\n4. Go Back\n\nChoice: ";

cin>>ch;

return ch;

}

LL \* SearchLeast(LL \* head, int menu)

{

LL \* p = head;

LL \* minnode = NULL;

double minimum = 100000.0;

while(p != NULL)

{

if(p->menu == menu)

{

if(p->time < minimum)

{

minimum = p->time;

minnode = p;

}

}

p = p->next;

}

return minnode;

}

void mainProgram(LL\* head, string Name)

{

int ch;

ifstream b,i,e;

b.open("Beginner.txt");

i.open("Intermediate.txt");

e.open("Expert.txt");

do

{

LL node;

system("CLS");

ch = SubMenu();

switch(ch)

{

case 1:

{

string line;

getline(b,line);

node = whereItHappens(line);

break;

}

case 2:

{

string line;

getline(i,line);

node = whereItHappens(line);

break;

}

case 3:

{

string line;

getline(e,line);

node = whereItHappens(line);

break;

}

case 4: break;

}

if(ch != 4) head = Insert(head,Name,node.acc,node.time,ch);

}while(ch != 4);

b.close();

i.close();

e.close();

for(int j = 1; j <= 3; j++)

{

LL \* temp = SearchLeast(head, j);

ofstream y;

if(j == 1) y.open("BegPrevGames.txt", ios:: app);

else if(j == 2) y.open("IntPrevGames.txt", ios:: app);

else if(j == 3) y.open("AdvPrevGames.txt", ios:: app);

if(temp)

{

y<< temp->name<< "#"

<< temp->acc<< " "

<< temp->time<< "\n" ;

}

y.close();

}

}

void newUser()

{

LL \* head;

head = NULL;

string name;

cout<<"Enter Username: ";

cin.ignore();

getline(cin,name);

mainProgram(head,name);

}

void oldUser()

{

Create();

LL \* head;

head = NULL;

string name;

cout<<"Enter Username: ";

cin.ignore();

getline(cin, name);

ifstream f;

f.open("BegPrevGames.txt");

while(!f.eof())

{

string NameFromFile;

f>> NameFromFile;

if(NameFromFile == name)

{

cout<< "You have a save in Beginner"<<endl;

break;

}

getline(f,NameFromFile);

}

f.close();

f.open("IntPrevGames.txt");

while(!f.eof())

{

string NameFromFile;

f>> NameFromFile;

if(NameFromFile == name)

{

cout<< "You have a save in Intermediate"<<endl;

break;

}

getline(f,NameFromFile);

}

f.close();

f.open("AdvPrevGames.txt");

while(!f.eof())

{

string NameFromFile;

f>> NameFromFile;

if(NameFromFile == name)

{

cout<< "You have a save in Expert"<<endl;

break;

}

getline(f,NameFromFile);

}

f.close();

string NameFromFile; double AccFromFile; double TimeFromFile;

point1:

int ch = SubMenu();

switch(ch)

{

case 1: {f.open("BegPrevGames.txt");break;}

case 2: {f.open("IntPrevGames.txt");break;}

case 3: {f.open("AdvPrevGames.txt");break;}

default: {cout<< "Invalid Input. Please Enter Correct Input"; goto point1;}

}

f>> NameFromFile;

f>> AccFromFile;

f>> TimeFromFile;

head = Insert(head, NameFromFile, AccFromFile, TimeFromFile, ch);

mainProgram(head, name);

}

void viewRecords()

{

LL \* head, \* Min; int i = 1;

head = NULL;

string NameFromFile; double AccFromFile; double TimeFromFile;

ifstream f;

point2:

system("CLS");

int ch = SubMenu();

switch(ch)

{

case 1: {f.open("BegPrevGames.txt"); break; }

case 2: {f.open("IntPrevGames.txt"); break; }

case 3: {f.open("AdvPrevGames.txt"); break; }

default: {cout<< "Invalid Input. Please Enter Correct Input."; goto point2; }

}

while(true)

{

getline(f,NameFromFile,'#');

if(f.eof()) break;

f>> AccFromFile;

f>> TimeFromFile;

head = Insert(head, NameFromFile, AccFromFile, TimeFromFile, ch);

}

cout<< "Sl. No.\t\tName\t\tTime\t\tAccuracy\n";

Min = head;

while(Min)

{

cout<< i << "\t\t" << Min->name << "\t\t" << Min->time << "\t\t" << Min->acc << endl;

Min = Min->next;

i++;

}

cout<< "Press any key to return: ";

string anyKey;

cin>> anyKey;

}

int Menu()

{

int ch;

cout<<"Welcome to Typing Tutor\n\n\n1. New User\n2. Existing User\n3. View Records\n4. Exit\n\n CHOICE: ";

cin>>ch;

return ch;

}

int main()

{

while(true)

{

system("CLS");

switch(Menu())

{

case 1:

newUser();

break;

case 2:

oldUser();

break;

case 3:

viewRecords();

break;

case 4:

exit(0);

default: cout<<"Input Entered is invalid, please enter a Valid Input"<<endl;

}

}

}

**LLops.cpp:**

#include<stdlib.h>

#include"Prerequisites.h"

LL\* Insert(LL\* head, string N, double A, double T, int M)

{

LL \* node = new LL;

node->name = N;

node->acc = A;

node->time = T;

node->menu = M;

node->next = NULL;

if(head == NULL) node->next = NULL;

else

{

node->next = head;

}

head = node;

return head;

}

LL \* Delete(LL\* head, string N)

{

LL \* node = new LL;

node = Search(head, N);

if(node == NULL)

{

cout<<"Record not found";

return head;

}

LL \* temp1 = node->next;

temp1->next = node->next;

delete temp1;

return head;

}

LL \* Search(LL\* head, string N)

{

LL\* p = NULL;

p = head;

while(p != NULL)

{

if((p->next)->name == N)

return p;

p = p->next;

}

return NULL;

}

**FileCreator.cpp:**

#include<fstream>

using namespace std;

void Create()

{

std::fstream file;

file.open("BegPrevGames.txt", std::ios::app);

file.close();

file.open("IntPrevGames.txt", std::ios::app);

file.close();

file.open("AdvPrevGames.txt", std::ios::app);

file.close();

}

**Prerequisites.h:**

#include<iostream>

#include<time.h>

#include<windows.h>

using namespace std;

struct LL

{

int menu;

string name;

double time;

double acc;

LL \* next;

};

LL\* Insert(LL\*, string, double, double, int);

LL \* Delete(LL\*, string);

LL \* Search(LL\*, string);

void Create();

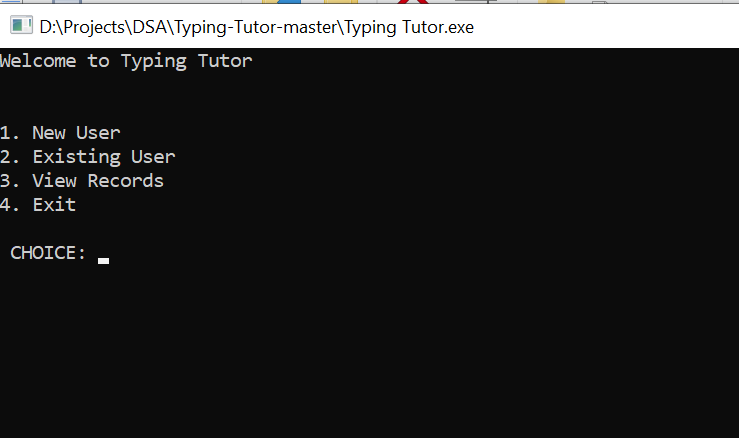
void newUser();

void oldUser();

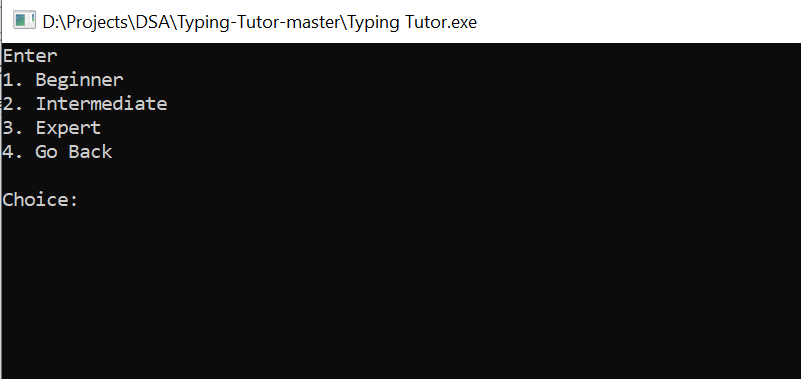
void viewRecords();

**Console output:**

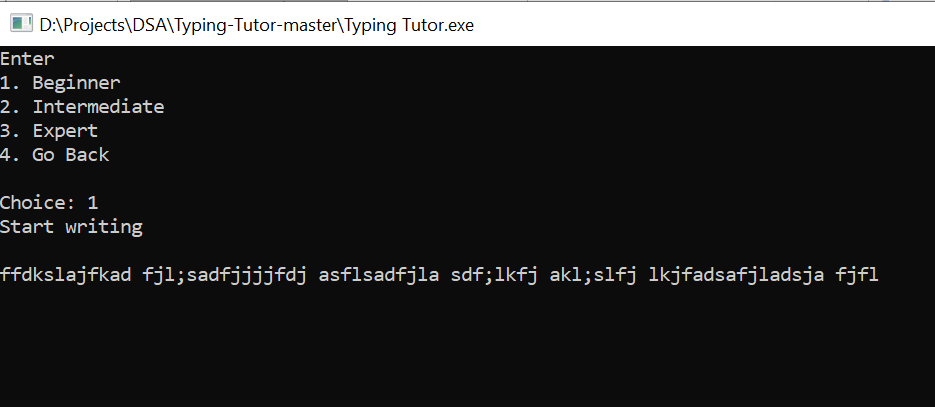
Main menu:



Sub menu:



Beginner level:



Displaying attempt results:

