

OOP

Assignment#2



Name: Talib Husain

Roll# 21F-9070

Task-1

// Making of a TIC TAC TOE Game

#include<iostream>

using namespace std;

void boardfilling(char\*\* array);

void boardprinting(char\*\* array);

bool winningcond(char\*\* array);

void playersturn(char\*\* array);

// Driver COde

int main() {

char \*\*array = new char\* [3];

for (int i = 0; i < 3; i++) {

\*(array + i) = new char[3];

}

//calling a fun to start a game

playersturn(array);

}

//wrtting a fun to fill the board with corresponding number 1-9

void boardfilling(char\*\* array) {

int x = 49;

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

\*(\*(array + i) + j) = x;

x++;

}

}

}

//function to print the board every time

void boardprinting(char\*\* array) {

for (int i = 0; i < 3; i++) {

for (int j = 0; j < 3; j++) {

cout << \*(\*(array + i) + j) << " ";

}

cout << "\n";

}

}

//function to check who win the game

bool winningcond(char\*\* array) {

char p1 = 'X', p2 = 'O';

//For checking player-1

if (

array[0][0] == p1 && array[0][0] == array[0][1] && array[0][1] == array[0][2] ||

array[1][0] == p1 && array[1][0] == array[1][1] && array[1][1] == array[1][2] ||

array[2][0] == p1 && array[2][0] == array[2][1] && array[2][1] == array[2][2] ||

array[0][0] == p1 && array[0][0] == array[1][0] && array[1][0] == array[2][0] ||

array[0][1] == p1 && array[0][1] == array[1][1] && array[1][1] == array[2][1] ||

array[0][2] == p1 && array[0][2] == array[1][2] && array[1][2] == array[2][2] ||

array[0][0] == p1 && array[0][0] == array[1][1] && array[1][1] == array[2][2] ||

array[0][2] == p1 && array[0][2] == array[1][1] && array[1][1] == array[2][0]

)

return 1; //returns '1' if Player-1 Won

//For checking player-2

else if (

array[0][0] == p2 && array[0][0] == array[0][1] && array[0][1] == array[0][2] ||

array[1][0] == p2 && array[1][0] == array[1][1] && array[1][1] == array[1][2] ||

array[2][0] == p2 && array[2][0] == array[2][1] && array[2][1] == array[2][2] ||

array[0][0] == p2 && array[0][0] == array[1][0] && array[1][0] == array[2][0] ||

array[0][1] == p2 && array[0][1] == array[1][1] && array[1][1] == array[2][1] ||

array[0][2] == p2 && array[0][2] == array[1][2] && array[1][2] == array[2][2] ||

array[0][0] == p2 && array[0][0] == array[1][1] && array[1][1] == array[2][2] ||

array[0][2] == p2 && array[0][2] == array[1][1] && array[1][1] == array[2][0]

)

return 2; //returns '2' if Player-2 Won

else

return 0; //returns '0' if GAME DRAW

}

void playersturn(char\*\* array) {

int row, col;

char s;

//calling to fill the board

boardfilling(array);

for (int i = 0; i < 9; i++) {

system("cls");

//calling to print the board

boardprinting(array);

//switching players turn

if (i % 2 == 0) {

s = 'X';

cout << "Player-1 Turn and Symbol is 'X'\n";

}

else {

s = 'O';

cout << "Player-2 Turn and Symbol is 'O'\n";

}

//getting the dimensions of board to writhe with 'O' or 'X'

cout << "Row: ";

cin >> row;

cout << "Col: ";

cin >> col;

\*(\*(array + row - 1) + col - 1) = s;

//calling fun to check the winnig possibilities

bool iswin = winningcond(array);

if (iswin == 1) {

cout << "\nPLayer-1 Wins the Game";

break;

}

else if (iswin == 2) {

cout << "\nPlayer-2 Wins the Game";

}

else if (i == 8) {

cout << "\nGame is Draw\n";

cout << "PLay Again (1/0): ";

int again;

cin >> again;

// To play again

if (again == 1) {

boardfilling(array);

playersturn(array);

}

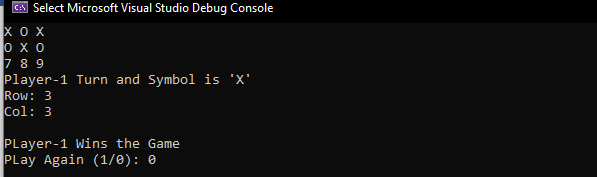
else

break;

}

}

}



Task-2

#include<iostream>

using namespace std;

int main() {

int row, \*\* array;

int\* col;

cout << "Enter a Number of Rows: ";

cin >> row;

//making a array of coloums for jagged array

col = new int[row];

array = new int\* [row];

//dynamacally making 2-d array

for (int i = 0; i < row; i++) {

cout << "Enter a Col-"<<i+1<<" Size\n";

cin >> col[i];

\*(array + i) = new int[col[i]];

}

// Getting elements from user

cout << "Enter Elements\n";

for (int i = 0; i < row; i++) {

for (int j = 0; j < col[i]; j++) {

cout << "Enter col-"<<i+1<<" Elements\n";

cin >> \*(\*(array + i) + j);

}

}

//printing elements of an array

cout << "Elements Of array are\n";

for (int i = 0; i < row; i++) {

for (int j = 0; j < col[i]; j++) {

cout << \*(\*(array + i) + j) << " ";

}

cout << "\n";

}

//deletion of array

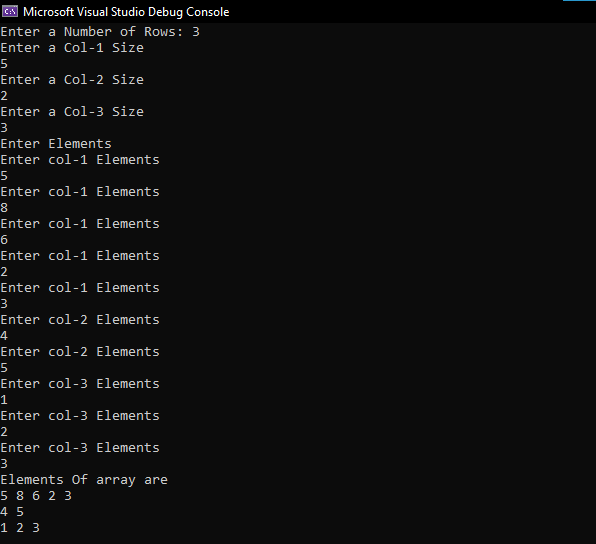
for (int i = 0; i < row; i++) {

delete\* (array + i);

}

delete[]array;

}



Task-3

#include<iostream>

#include<string>

using namespace std;

//funtuon prototye to reverse string

void reverse(string str);

//making a global variables

int len, lastlen;

//Driver Code

int main() {

string str;

cout << "Enter a string: ";

getline(cin, str);

lastlen = str.length() - 1;

len = lastlen;

cout << "The reversed string\n";

reverse(str);

}

//writting of a function

void reverse(string str) {

//base condition

if (len < 0)

return;

else {

if (str[len] == ' ' || len == 0)

{

if (len == 0)

len = -1;

for (int i = len + 1; i <= lastlen; i++)

cout << str[i];

lastlen = len - 1;

cout << ' ';

}

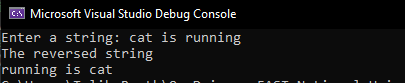
len--;

//recurring a funcion

reverse(str);

}

}



Task-4

#include<iostream>

#include<string>

using namespace std;

// making a struct

struct book {

string author{}, title{};

string bookid{};

int quantity;

};

//function proto types

void addBook(book\* b, int n);

void search(book\* b);

void display(book\* b, int n);

void update(book\* b);

void issue(book\* b);

book\* resizeArray(book\* b, int n);

//Driver code

int main() {

book\* rec = new book[3];

int ch;

int n = 0;

do {

cout << "1-Add a Book\n2-Search a Book\n3-Dislpay Info of all Books\n";

cout << "4-Update Data\n5-Issue a Book\n6-Exit\nChoose an Option: ";

cin >> ch;

cin.ignore();

switch (ch) {

case 1:

if (n >= 3) {

rec = resizeArray(rec, n);

}

addBook(rec,n);

n++;

break;

case 2:

search(rec);

break;

case 3:

display(rec, n);

break;

case 4:

update(rec);

break;

case 5:

issue(rec);

break;

case 6:

exit(1);

break;

}

cout << "\n";

system("pause");

system("cls");

} while ( ch!=6);

}

//for the adding of a data

void addBook(book\* b, int n) {

cout << "Enter a Book Title: ";

getline(cin, b[n].title);

cout << "Enter a Author Name: ";

getline(cin, b[n].author);

cout << "Enter a Book ID: ";

getline(cin, b[n].bookid);

cout << "Enter a Quantiy of Book: ";

cin >> b[n].quantity;

cin.ignore();

}

//functiojn to search a specific data

void search(book\* b) {

string str;

cout << "Enter a Book Title/Id/Author Name\n";

getline(cin, str);

for (int i = 0; i < 3; i++) {

//finding the specific searched book

if (str == b[i].author || str == b[i].bookid || str == b[i].title) {

cout << "Book Title: " << b[i].title << "\nAuthor Name: " << b[i].author << "\n";

cout << "Book ID: " << b[i].bookid << "\nQuantiy: " << b[i].quantity;

continue;

}

}

}

//displaying all data

void display(book\* b, int n) {

for (int i = 0; i < n; i++) {

cout << "Book-" << i + 1 << endl;

cout << "Book Title: " << b[i].title << "\nAuthor Name: " << b[i].author << "\n";

cout << "Book ID: " << b[i].bookid << "\nQuantiy: " << b[i].quantity << endl;

}

}

// function to update the data

void update(book\* b) {

string id;

cout << "Enter a Book ID: ";

getline(cin, id);

for (int i = 0; i < 3; i++) {

//prinitng the matched books

if (id == b[i].bookid) {

cout << "The Book Found in Libraray\n";

addBook(b, i);

}

}

}

// issuing a book

void issue(book\* b) {

string id;

cout << "Enter a Book ID: ";

getline(cin, id);

for (int i = 0; i < 3; i++) {

//finding a book

if (id == b[i].bookid) {

if (b[i].quantity == 0) {

cout << "This Book is out of Stock\n";

break;

}

b[i].quantity--;

}

}

}

// Function for resising of an array

book\* resizeArray(book\* b, int n) {

book\* temp = new book[n + 1];

//assigning a values to resized array

for (int i = 0; i < n; i++) {

temp[i].author = b[i].author;

temp[i].title = b[i].title;

temp[i].bookid = b[i].bookid;

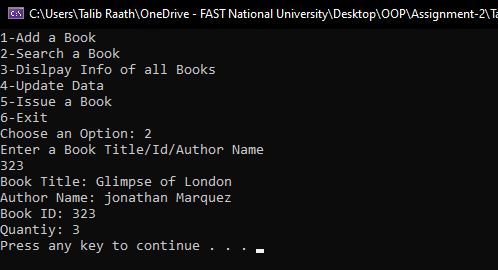
temp[i].quantity = b[i].quantity;

}

b = temp;

return temp;

}



Task-5

#include<iostream>

#include<string>

using namespace std;

//making a class of students

class student {

public:

string name{}, address{};

int id{};

long int CNIC{};

int n{};

string regcourses[5];

void stu(int id);

void regcourse(string c, int n);

void drop();

void search();

};

//Witting of functions

void student::stu(int id) {

this->id = id;

cout << "Enter a Name: ";

getline(cin, name);

//cin>>name;

cout << "Enetr a CNIC: ";

cin >> CNIC;

cin.ignore();

cout << "Enter a Address: ";

getline(cin, address);

//cin >> address;

}

void student::regcourse(string c, int n) {

this->n = n;

static int i = 0;

this->regcourses[i] = c;

i++;

}

void student::drop() {

int choice;

cout << "List of Registered Courses\n";

for (int i = 0; i < n; i++) {

cout << i << "-" << regcourses[i] << "\n";

}

cout << "CHoose a course to Drop: ";

cin >> choice;

regcourses[choice] = "Already Dropped";

}

void student::search() {

int s;

cout << "Enter a Student Id: ";

cin >> s;

cout << this->name << "\n" << id << "\n" << CNIC << "\n" << this->address << "\nCourses\n";

for (int i = 0; i < this->n; i++) {

cout << regcourses[i] << "\n";

}

}

class cours {

public:

int n;

string cour[5]{};

void set();

void get();

};

void cours::set() {

cout << "Enetr a no of courses : ";

cin >> n;

cin.ignore();

for (int i = 0; i < n; i++) {

cout << "Enter a Course" << i + 1 << " Name: ";

getline(cin, cour[i]);

//cin >> cour[i];

}

}

void cours::get() {

cout << "LIst of COurses Offered\n";

for (int i = 0; i < n; i++) {

cout << i + 1 << "-" << cour[i] << endl;

}

}

int main() {

int id, choice = 0;

cours c;

student obj1;

for (; choice != 6;) {

cout << "1-Register Stuent\n2-Offer Courses\n3-Drop Courses\n4-Update Record\n5-Search Student\n6-Exit\nEnter Your Choice: ";

cin >> choice;

cin.ignore();

if (choice == 2) {

c.set();

}

else if (choice == 1) {

cout << "Enter a Student Id: ";

cin >> id;

cin.ignore();

obj1.stu(id);

int n = 0;

cout << "Enetr a no of courses : ";

cin >> n;

cin.ignore();

c.get();

int temp;

for (int i = 0; i < n; i++) {

cout << "Enter Course: ";

cin >> temp;

string str;

str = c.cour[temp - 1];

obj1.regcourse(str, n);

}

}

else if (choice == 3) {

obj1.drop();

}

else if (choice == 4) {

cout << "Enter a Student Id: ";

cin >> id;

obj1.stu(id);

}

else if (choice == 5) {

cin.ignore();

obj1.search();

}

else if (choice == 6) {

exit(1);

}

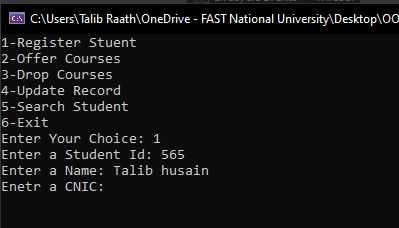
cout << "\n";

system("pause");

system("cls");

}

}



Task-6

//Employee Classes

#ifndef employee\_h

#define employee\_h

class employee {

private:

const int id;

char name[20];

int salaryperhour, monthlyworkedhours, taxpaid;

int yearofjoining;

int monthlysalary;

public:

employee(int id);

void updatehours();

void changesalary();

void cal\_monthly\_sal();

void taxdeduction();

void healthinsurnce();

void senioremployee();

void generatereceipt();

};

#endif;

// Employee classes Definition

#include"employee.h"

#include<iostream>

#include<string>

using namespace std;

employee::employee(int id) : id(id) {

cout << "ENter a Name: ";

cin.get(name, 20);

cin.ignore();

cout << "ENter a Salary Hour: ";

cin >> salaryperhour;

cout << "ENter a Monthly Worked Hour: ";

cin >> monthlyworkedhours;

cout << "ENter Year of joining: ";

cin >> yearofjoining;

taxpaid = 0;

}

void employee::updatehours() {

cout << "Enter Hours";

cin >> monthlyworkedhours;

}

void employee::cal\_monthly\_sal() {

this->monthlysalary = monthlyworkedhours \* salaryperhour;

}

void employee::changesalary() {

cout << "Enter Salary";

cin >> salaryperhour;

}

void employee::taxdeduction() {

this->taxpaid = (0.1) \* this->monthlysalary;

this->monthlysalary -= taxpaid;

}

void employee::healthinsurnce() {

this->monthlysalary -= 1000;

}

void employee::senioremployee() {

if (2022 - yearofjoining >= 8) {

this->monthlysalary += (0.1) \* monthlysalary;

}

}

void employee::generatereceipt() {

cout << "Name: " << this->name << "\nID: " << this->id << "\nSalary: " << monthlysalary;

cout << "\nTaxPaid: " << taxpaid;

}

//Main Function

#include<iostream>

#include<string>

#include "employee.h"

using namespace std;

int main() {

int choice = 0;

int id;

cout << "Enter an ID: ";

cin >> id;

cin.ignore();

employee obj1(id);

for (; choice != 8; ) {

cout << "0-Calculate Salary\n1-Enter Data\n2-Update Hours\n3-Change Salary\n4-Tax Deduction\n5-Health Insurance\n6-IsSenior\n7-Genreate Slip\n8-Exit\n";

cout << "ENter YOur Choice: ";

cin >> choice;

if (choice == 1) {

int id;

cout << "ENter an ID: ";

cin >> id;

cin.ignore();

employee obj1(id);

}

else if (choice == 2) {

obj1.updatehours();

}

else if (choice == 3) {

obj1.changesalary();

}

else if (choice == 4) {

obj1.taxdeduction();

}

else if (choice == 5) {

obj1.healthinsurnce();

}

else if (choice == 6) {

obj1.senioremployee();

}

else if (choice == 0) {

obj1.cal\_monthly\_sal();

}

else if (choice == 7) {

obj1.generatereceipt();

}

else if (choice == 8) {

exit(1);

}

cout << "\n";

system("pause");

system("cls");

}

return 0;

}

