Assignment #2

Name:

Kabeer Ahmad

Section:

BSCB 2A

Submitted to:

Noor Ullah Khan

(EXERCISE Part A)

Question 1:

a. False, members of a struct can be of the same or different types.

b. False, a struct is both a definition and a declaration.

c. False, struct variables can be declared before or after the struct definition.

d. False, the -> operator is used to access struct members.

e. False, other operations like passing to functions, returning from functions, etc. are allowed.

f. True

g. True, operations like cin/cout are allowed on struct variables.

h. True

i. False, a struct can be returned from a function.

j. True

k. True

Question 2:

struct computerType

{

string manufacturer;

string modelType;

string processorType;

int ram;

int hardDriveSize;

int year;

double price;

};

Question 4:

a.

computerType computer;

computer.manufacturer = "Computer Corporation";

computer.modelType = "Desk Top";

computer.processorType = "Core I 7";

computer.ram = 12;

computer.hardDriveSize = 500;

computer.year = 2016;

computer.price = 875.00;

b.

oldHouse.style = "Two-story";

oldHouse.bedrooms = 5;

oldHouse.bathrooms = 3;

oldHouse.garageSpaces = 4;

oldHouse.yearBuilt = 1975;

oldHouse.finishedSqFt = 3500;

oldHouse.price = 675000;

oldHouse.tax = 12500;

c.

oldHouse.style = "Two-story";

oldHouse.bedrooms = 5;

oldHouse.bathrooms = 3;

oldHouse.garageSpaces = 4;

oldHouse.yearBuilt = 1975;

oldHouse.finishedSqFt = 3500;

oldHouse.price = 675000;

oldHouse.tax = 12500;

Question 5:

if (firstHouse.style == secondHouse.style &&

firstHouse.price == secondHouse.price)

cout << "true";

else

cout << "false";

Question 6:

struct fruitType

{

string fruitName;

string color;

int fat;

int sugar;

int carbohydrate;

};

Question 7:

fruitType fruit = {"banana", "yellow", 1, 15, 22};

Question 8:

void printFruitInfo(fruitType f)

{

cout << "Fruit Name: " << f.fruitName << endl;

cout << "Color: " << f.color << endl;

// print other data

}

Question 9:

Assignment operation like = is allowed on struct variables but not on arrays.

Question 10:

a. Valid

b. Valid

c. Invalid, cannot assign a studentType to a nameType

d. Valid

e. Valid

f. Invalid, cannot assign a courseType to a studentType

g. Invalid, cannot use << on a struct variable with cin/cout

h. Valid

i. Invalid, classList is an array, no member access with .

j. Invalid, studentType does not have a course member

Question 11:

a.

classList[0].name.first = "Jessica"; classList[0].name.last = "Miller"; classList[0].gpa = 3.8; classList[0].course.name = "Data Structure"; classList[0].course.callNum = 8340; classList[0].course.credits = 3; classList[0].course.grade = 'B';

b.

student = classList[0];

Question 12:

course.name = "Programming I";

course.callNum = 13452;

course.credits = 3;

course.grade = "";

b. for (int i=0; i<100; i++) classList[i].gpa = 0.0;

c. student = classList[30];

d. classList[9].gpa += 0.75;

Question 13:

a. Invalid, cannot assign a string literal to a nameType

b. Valid

c. Valid

d. Valid

e. Invalid, employees is an array, no . member access

Question 14:

a.

newEmployee.name.first = "Mickey"; newEmployee.name.last = "Doe";

newEmployee.pID = 111111111; newEmployee.performanceRating = 2; newEmployee.dept = "ACCT";

newEmployee.salary = 34567.78;

b.

for (int i=0; i<100; i++) employees[i].performanceRating = 0;

c. newEmployee = employees[19];

d. employees[49].salary += 5735.87;

Question 15:

sportsType soccer[20];

Question 16:

1. for (int i=0; i<20; i++)

{

soccer[i].sportName = "";

soccer[i].teamName = ""; soccer[i].numberOfPlayers = 0; soccer[i].teamPayroll = 0.0;

soccer[i].coachSalary = 0.0;

}

b.

int length;

cout << "Enter number of teams: ";

cin >> length;

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

{

cout << "Enter sport name: ";

cin >> soccer[i].sportName;

for (int i=0; i<20; i++)

{

if (soccer[i].teamPayroll >= 10000000)

cout << soccer[i].teamName << endl;

}

}

Question 17:

a.

void getSportsData(sportsType &s)

{

cout << "Enter sport name: ";

cin >> s.sportName;

}

sportsType team;

getSportsData(team);

b.

void printSportsData(sportsType s)

{

cout << "Sport Name: " << s.sportName << endl;

}

for (int i=0; i<20; i++)

printSportsData(soccer[i]);

Question 18:

a. tourType destination;

b.

destination.cityName = "Chicago"; destination.distance = 550;

destination.travelTime.hr = 9; destination.travelTime.min = 30; destination.travelTime.sec = 0;

c.

void printTourInfo(tourType t)

{

cout << "City: " << t.cityName << endl;

cout << "Distance: " << t.distance << " miles" << endl;

cout << "Travel Time: " << t.travelTime.hr << ":"

<< t.travelTime.min << ":" << t.travelTime.sec << endl;

}

d.

tourType inputTourInfo()

{

tourType temp; cout << "Enter city name: "; cin >> temp.cityName;

}

e.

void getTourInfo(tourType &t)

{

cout << "Enter city name: ";

cin >> t.cityName;

}

Programing Exercises

Question 1:

#include <iostream>

using namespace std;

struct computerType {

string manufacturer;

string modelType;

string processorType;

int ramGB;

int hardDriveGB;

int yearBuilt;

double price;

};

int main() {

computerType myComputer;

cout << "Enter manufacturer of the computer: ";

getline(cin, myComputer.manufacturer);

cout << "Enter model type of the computer: ";

getline(cin, myComputer.modelType);

cout << "Enter processor type of the computer: ";

getline(cin, myComputer.processorType);

cout << "Enter RAM size of the computer in GB: ";

cin >> myComputer.ramGB;

cout << "Enter hard drive size of the computer in GB: ";

cin >> myComputer.hardDriveGB;

cout << "Enter year when the computer was built: ";

cin >> myComputer.yearBuilt;

cout << "Enter price of the computer: ";

cin >> myComputer.price;

cout << "\nComputer Information:\n";

cout << "Manufacturer: " << myComputer.manufacturer << endl;

cout << "Model Type: " << myComputer.modelType << endl;

cout << "Processor Type: " << myComputer.processorType << endl;

cout << "RAM (GB): " << myComputer.ramGB << endl;

cout << "Hard Drive (GB): " << myComputer.hardDriveGB << endl;

cout << "Year Built: " << myComputer.yearBuilt << endl;

cout << "Price: $" << myComputer.price << endl;

return 0;

}

Question 2:

#include <iostream>

#include <string>

using namespace std;

struct studentType {

string studentFName;

string studentLName;

int testScore;

char grade;

};

void readStudentData(studentType students[], int numStudents) {

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

cout << "Enter student " << i+1 << "'s first name: ";

cin >> students[i].studentFName;

cout << "Enter student " << i+1 << "'s last name: ";

cin >> students[i].studentLName;

cout << "Enter student " << i+1 << "'s test score: ";

cin >> students[i].testScore;

}

}

void assignGrades(studentType students[], int numStudents) {

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

if (students[i].testScore >= 90)

students[i].grade = 'A';

else if (students[i].testScore >= 80)

students[i].grade = 'B';

else if (students[i].testScore >= 70)

students[i].grade = 'C';

else if (students[i].testScore >= 60)

students[i].grade = 'D';

else

students[i].grade = 'F';

}

}

int findHighestScore(studentType students[], int numStudents) {

int highestScore = students[0].testScore;

for (int i = 1; i < numStudents; ++i) {

if (students[i].testScore > highestScore)

highestScore = students[i].testScore;

}

return highestScore;

}

void printStudentsWithHighestScore(studentType students[], int numStudents, int highestScore) {

cout << "Students with the highest test score (" << highestScore << "):\n";

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

if (students[i].testScore == highestScore)

cout << students[i].studentLName << ", " << students[i].studentFName << endl;

}

}

void outputStudentInfo(studentType students[], int numStudents) {

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

cout << students[i].studentLName << ", " << students[i].studentFName << ": ";

cout << "Test Score: " << students[i].testScore << ", Grade: " << students[i].grade << endl;

}

}

int main() {

const int numStudents = 20;

studentType students[numStudents];

readStudentData(students, numStudents);

assignGrades(students, numStudents);

outputStudentInfo(students, numStudents);

int highestScore = findHighestScore(students, numStudents);

printStudentsWithHighestScore(students, numStudents, highestScore);

return 0;

}

Question 3:

#include<iostream>

using namespace std;

struct menuItemType {

string menuItem;

double menuPrice;

};

int main(){

return 0;

}

Question 4:

#include <iostream>

#include <iomanip>

using namespace std;

struct menuItemType {

string menuItem;

double menuPrice;

};

void getData(menuItemType menuList[], int numItems) {

menuList[0] = {"Plain Egg", 1.45};

menuList[1] = {"Bacon and Egg", 2.45};

menuList[2] = {"Muffin", 0.99};

menuList[3] = {"French Toast", 1.99};

menuList[4] = {"Fruit Basket", 2.49};

menuList[5] = {"Cereal", 0.69};

menuList[6] = {"Coffee", 0.50};

menuList[7] = {"Tea", 0.75};

}

void showMenu(menuItemType menuList[], int numItems) {

cout << "Welcome to Johnny's Restaurant" << endl;

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

cout << left << setw(15) << menuList[i].menuItem << "$" << fixed << setprecision(2) << menuList[i].menuPrice << endl;

}

cout << endl;

cout << "To select an item, enter the number corresponding to the item:" << endl;

cout << "For example, enter 1 for Plain Egg, 2 for Bacon and Egg, and so on." << endl;

}

void printCheck(menuItemType menuList[], int selections[], int numSelections) {

double total = 0;

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

total += menuList[selections[i] - 1].menuPrice;

}

double tax = total \* 0.05;

double amountDue = total + tax;

cout << "Tax $" << fixed << setprecision(2) << tax << endl;

cout << "Amount Due $" << fixed << setprecision(2) << amountDue << endl;

}

int main() {

const int numItems = 8;

menuItemType menuList[numItems];

int selections[numItems];

int numSelections = 0;

getData(menuList, numItems);

showMenu(menuList, numItems);

int selection;

cout << "Enter the numbers corresponding to the items you want to order (enter 0 to finish):" << endl;

while (true) {

cin >> selection;

if (selection == 0)

break;

else if (selection < 1 || selection > numItems) {

cout << "Invalid selection. Please enter a number between 1 and " << numItems << "." << endl;

} else {

selections[numSelections++] = selection;

}

}

cout << endl;

printCheck(menuList, selections, numSelections);

return 0;

}

Question 5:

#include <iostream>

#include <iomanip>

using namespace std;

struct menuItemType {

string menuItem;

double menuPrice;

};

void getData(menuItemType menuList[], int numItems) {

menuList[0] = {"Plain Egg", 1.45};

menuList[1] = {"Bacon and Egg", 2.45};

menuList[2] = {"Muffin", 0.99};

menuList[3] = {"French Toast", 1.99};

menuList[4] = {"Fruit Basket", 2.49};

menuList[5] = {"Cereal", 0.69};

menuList[6] = {"Coffee", 0.50};

menuList[7] = {"Tea", 0.75};

}

void showMenu(menuItemType menuList[], int numItems) {

cout << "Welcome to Johnny's Restaurant" << endl;

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

cout << i + 1 << " " << menuList[i].menuItem << " $" << fixed << setprecision(2) << menuList[i].menuPrice << endl;

}

cout << endl;

cout << "To select an item, enter the number corresponding to the item followed by the quantity:" << endl;

cout << "For example, enter '1 2' to select 2 Bacon and Egg." << endl;

}

void printCheck(menuItemType menuList[], int selections[], int quantities[], int numSelections) {

double total = 0;

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

int index = selections[i] - 1;

total += menuList[index].menuPrice \* quantities[i];

cout << quantities[i] << " " << menuList[index].menuItem << " $" << fixed << setprecision(2) << menuList[index].menuPrice \* quantities[i] << endl;

}

double tax = total \* 0.05;

double amountDue = total + tax;

cout << "Tax $" << fixed << setprecision(2) << tax << endl;

cout << "Amount Due $" << fixed << setprecision(2) << amountDue << endl;

}

int main() {

const int numItems = 8;

menuItemType menuList[numItems];

int selections[numItems];

int quantities[numItems];

int numSelections = 0;

getData(menuList, numItems);

showMenu(menuList, numItems);

cout << "Enter the numbers corresponding to the items you want to order followed by the quantity (enter 0 to finish):" << endl;

int selection, quantity;

while (true) {

cin >> selection;

if (selection == 0)

break;

cin >> quantity;

selections[numSelections] = selection;

quantities[numSelections] = quantity;

numSelections++;

}

cout << endl;

printCheck(menuList, selections, quantities, numSelections);

return 0;

}

Question 6:

#include <iostream>

#include <fstream>

#include <iomanip>

using namespace std;

struct LetterCount {

char letter;

int capitalCount;

int smallCount;

};

void openFile(ifstream& inFile, ofstream& outFile);

void countLetters(ifstream& inFile, LetterCount counts[]);

void printResult(LetterCount counts[]);

int main() {

ifstream inFile;

ofstream outFile;

LetterCount counts[26];

openFile(inFile, outFile);

countLetters(inFile, counts);

printResult(counts);

inFile.close();

outFile.close();

return 0;

}

void openFile(ifstream& inFile, ofstream& outFile) {

string inputFileName, outputFileName;

cout << "Enter the name of the input file: ";

cin >> inputFileName;

inFile.open(inputFileName);

if (!inFile) {

cout << "Error: Unable to open input file." << endl;

exit(0);

}

}

void countLetters(ifstream& inFile, LetterCount counts[]) {

char ch;

while (inFile.get(ch)) {

if (isalpha(ch)) {

if (isupper(ch))

counts[ch - 'A'].capitalCount++;

else

counts[ch - 'a'].smallCount++;

}

}

}

void printResult(LetterCount counts[]) {

for (int i = 0; i < 26; ++i) {

char letter = 'A' + i;

int total = counts[i].capitalCount + counts[i].smallCount;

cout << letter << ": " << counts[i].capitalCount << " capital letters, " << counts[i].smallCount << " small letters, ";

if (total != 0) {

cout << fixed << setprecision(2) << (counts[i].capitalCount \* 100.0 / total) << "% capital, ";

cout << fixed << setprecision(2) << (counts[i].smallCount \* 100.0 / total) << "% small" << endl;

} else {

cout << "0% capital, 0% small" << endl;

}

}

}

Question 7:

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

// Structure to store football player data

struct FootballPlayer {

string playerName;

string playerPosition;

int touchdowns;

int catches;

int passingYards;

int receivingYards;

int rushingYards;

};

// Function prototypes

void inputData(FootballPlayer players[], int size);

void outputData(FootballPlayer players[], int size);

int searchPlayer(FootballPlayer players[], int size, string playerName);

void updatePlayerData(FootballPlayer players[], int size);

void saveDataToFile(FootballPlayer players[], int size);

int main() {

const int size = 10;

FootballPlayer players[size];

int choice;

do {

cout << "\nMenu:\n";

cout << "1. Input player data\n";

cout << "2. Output player data\n";

cout << "3. Search for a player\n";

cout << "4. Update player data\n";

cout << "5. Save data to file\n";

cout << "6. Exit\n";

cout << "Enter your choice: ";

cin >> choice;

switch (choice) {

case 1:

inputData(players, size);

break;

case 2:

outputData(players, size);

break;

case 3: {

string playerName;

cout << "Enter player name to search: ";

cin >> playerName;

int index = searchPlayer(players, size, playerName);

if (index != -1)

cout << "Player found at index " << index << endl;

else

cout << "Player not found." << endl;

break;

}

case 4:

updatePlayerData(players, size);

break;

case 5:

saveDataToFile(players, size);

break;

case 6:

cout << "Exiting program.\n";

break;

default:

cout << "Invalid choice. Please try again.\n";

}

} while (choice != 6);

return 0;

}

// Function to input player data

void inputData(FootballPlayer players[], int size) {

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

cout << "Enter details for player " << i + 1 << ":\n";

cout << "Name: ";

cin >> players[i].playerName;

cout << "Position: ";

cin >> players[i].playerPosition;

cout << "Touchdowns: ";

cin >> players[i].touchdowns;

cout << "Catches: ";

cin >> players[i].catches;

cout << "Passing yards: ";

cin >> players[i].passingYards;

cout << "Receiving yards: ";

cin >> players[i].receivingYards;

cout << "Rushing yards: ";

cin >> players[i].rushingYards;

}

}

void outputData(FootballPlayer players[], int size) {

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

cout << "Player " << i + 1 << ":\n";

cout << "Name: " << players[i].playerName << endl;

cout << "Position: " << players[i].playerPosition << endl;

cout << "Touchdowns: " << players[i].touchdowns << endl;

cout << "Catches: " << players[i].catches << endl;

cout << "Passing yards: " << players[i].passingYards << endl;

cout << "Receiving yards: " << players[i].receivingYards << endl;

cout << "Rushing yards: " << players[i].rushingYards << endl;

}

}

void saveDataToFile(FootballPlayer players[], int size) {

ofstream outFile("player\_data.txt");

if (outFile.is\_open()) {

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

outFile << "Player " << i + 1 << ":\n";

outFile << "Name: " << players[i].playerName << endl;

outFile << "Position: " << players[i].playerPosition << endl;

outFile << "Touchdowns: " << players[i].touchdowns << endl;

outFile << "Catches: " << players[i].catches << endl;

outFile << "Passing yards: " << players[i].passingYards << endl;

outFile << "Receiving yards: " << players[i].receivingYards << endl;

outFile << "Rushing yards: " << players[i].rushingYards << endl;

}

cout << "Player data saved to player\_data.txt" << endl;

outFile.close();

} else {

cout << "Unable to open file." << endl;

}

}

int searchPlayer(FootballPlayer players[], int size, string playerName) {

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

if (players[i].playerName == playerName)

return i;

}

return -1; // Player not found

}

void updatePlayerData(FootballPlayer players[], int size) {

string playerName;

cout << "Enter player name to update: ";

cin >> playerName;

int index = searchPlayer(players, size, playerName);

if (index != -1) {

cout << "Enter new details for player ";

cout << "Name: ";

cin >> players[index].playerName;

cout << "Position: ";

cin >> players[index].playerPosition;

cout << "Touchdowns: ";

cin >> players[index].touchdowns;

cout << "Catches: ";

cin >> players[index].catches;

cout << "Passing yards: ";

cin >> players[index].passingYards;

cout << "Receiving yards: ";

cin >> players[index].receivingYards;

cout << "Rushing yards: ";

cin >> players[index].rushingYards;

}

else cout<<"Player not found";

}