COP 2220 Week 4 Assignments

Due 11:59 PM Monday September 24

By: David McCain

**Review Questions**

1. B
2. A
3. C
4. A
5. A
6. C
7. A
8. B
9. D
10. D

**Exercises**

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | food  rent  utilities  car payment | double food = 0.0;  double rent = 0.0;  double utilities = 0.0;  double carPayment = 0.0; |
| **Process** | none | none |
| **Output** | total expenses | double totalExpenses = 0.0; |
| **Algorithm** | 1. enter food, rent, utilities, and car payment 2. calculate total expenses by adding together food, rent, utilities, and car payment 3. display total expenses | cout << “Food: $“;  cin >> food;  cout <<”Rent: $“;  cin >> rent;  cout <<”Utilities: $“;  cin >> utilities;  cout <<”Car payment: $”;  cin >> carPayment;  totalExpenses = food + rent + utilities + carPayment;  cout << “Total expenses: $ “  cout << totalExpenses << endl; |

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | latex price  mylar price  latex purchased  mylar purchased  sales tax rate of 6% | double latexPrice = 0.0;  double mylarPrice = 0.0;  int latexPurchased = 0;  int mylarPurchased = 0;  const double TAX\_RATE = 0.06; |
| **Process** | total latex cost  total mylar cost  subtotal  sales tax | double totalLatexCost = 0.0;  double totalMylarCost = 0.0;  double subtotal = 0.0;  double salesTax = 0.0; |
| **Output** | total cost | double totalCost = 0.0; |
| **Algorithm** | 1. enter latex price, mylar price, latex purchased, and mylar purchased 2. calculate total latex cost by multiplying latex purchased with latex price 3. calculate total mylar cost by multiplying mylar purchased with mylar price 4. calculate subtotal by adding total latex cost with total mylar cost 5. calculate sales tax by multiplying subtotal with sales tax rate 6. calculate total cost by adding subtotal with sales tax 7. display total cost | cout << “Latex price: $“;  cin >> latexPrice;  cout <<”Mylar price: $“;  cin >> mylarPrice;  cout <<”Latex purchased: “;  cin >> latexPurchased;  cout <<”Mylar purchased: ”;  cin >> mylarPurchased;  totalLatexCost = latexPrice \* latexPurchased;  totalMylarCost = mylarPrice \* mylarPurchased;  subtotal = totalLatexCost + totalMylarCost;  salesTax = subtotal \* TAX\_RATE;  totalCost = subtotal + salesTax;  cout << “Total cost : $”;  cout << totalCost << endl; |

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | food  rent  utilities  car payment (always $253.75) | double food = 0.0;  double rent = 0.0;  double utilities = 0.0;  const double carPayment = 253.75; |
| **Process** | none | none |
| **Output** | total expenses | double totalExpenses = 0.0; |
| **Algorithm** | 1. enter food, rent, utilities, and display car payment 2. calculate total expenses by adding together food, rent, utilities, and car payment 3. display total expenses | cout << “Food: $“;  cin >> food;  cout <<”Rent: $“;  cin >> rent;  cout <<”Utilities: $“;  cin >> utilities;  cout <<”Car payment is: $”;  cout << carPayment;  totalExpenses = food + rent + utilities + carPayment;  cout << “Total expenses: $ “  cout << totalExpenses << endl; |

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | quantity  total cost | int quantity = 0;  double totalCost = 0.0; |
| **Process** | none |  |
| **Output** | cost per item | double itemCost = 0.0; |
| **Algorithm** | * 1. enter the quantity and total cost   2. calculate the cost per item by dividing the total cost by the quantity   3. display the cost per item | cout << “Quantity: “;  cin >> quantity;  cout << “Total cost: $“;  cin >> totalCost;  itemCost = totalCost / quantity;  cout <<”Cost per item: $”;  cout << itemCost << endl; |

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | quantity sold  item cost  item selling price | int quantity = 0;  double cost = 0.0;  double sellPrice = 0.0; |
| **Process** | price and cost difference | double difference = 0.0; |
| **Output** | profit | double profit = o.o; |
| **Algorithm** | * 1. enter the quantity sold, item cost, and item selling price   2. calculate the cost and price difference by subtracting the item cost from the item selling price   3. calculate the profit by multiplying the price and cost difference by the quantity sold   4. display the profit | cout << “Quantity sold: “;  cin >> quantity;  cout << “Item cost: $“;  cin >> cost;  cout << “Selling price: $“;  cin >> sellPrice;  difference = sellPrice - cost;  profit = difference \* quantity;  cout <<”Cost per item: $”;  cout << itemCost << endl; |

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | number of pets  number of owners | int pets = 0;  int owners = 0; |
| **Process** | none |  |
| **Output** | number of pets per owner | double petsPerOwner = 0; |
| **Algorithm** | * 1. enter number of pets and number of owners   2. calculate number of pets per owner by dividing number of pets by number of owners   3. display number of pets per owner | cout <<”Enter number of pets: “;  cin >> pets;  cout <<”Enter number of owners: “;  cin >> owners;  petsPerOwner = pets / owners;  cout <<”The number of pets per owner is: “;  cout << petsPerOwner << endl; |

#include <iostream>

using namespace std;

int main()

{

int quantity = 0;

cout << “Enter the quantity ordered: “;

cin >> quantity;

cout <<”You entered “ << quantity << endl;

return 0;

} //end of main function

11)

Program with user input:

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | beginning balance  total deposits  total withdraws | double beginBalance = 0.0;  double deposits = 0.0;  double withdraws = 0.0; |
| **Process** | none |  |
| **Output** | savings balance at end of the month | double endBalance = 0.0; |
| **Algorithm** | * 1. enter the beginning balance, total deposits, and total withdraws   2. calculate the end balance by subtracting withdraws and adding deposits to beginning balance   3. display the end balance | cout << “Enter current balance: $“;  cin >> beginBalance;  cout <<”Enter amount of deposits: $“;  cin >> deposits;  cout <<”Enter amount of withdraws: $“;  cin >> withdraws;  endBalance = beginBalance – withdraws + deposits;  cout << “The new balance: $ “  cout << endBalance << endl; |

Desk check #1

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | beginning balance  total deposits  total withdraws | double beginBalance = 2545.75;  double deposits = 409.43;  double withdraws = 210.65; |
| **Process** | none |  |
| **Output** | savings balance at end of the month | double endBalance = 2744.53; |
| **Algorithm** | 1. calculate the end balance by subtracting withdraws and adding deposits to beginning balance 2. display the end balance | endBalance = beginBalance – withdraws + deposits;  cout << “The new balance: $ “  cout << endBalance << endl; |

Desk check #2

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | beginning balance  total deposits  total withdraws | double beginBalance = 1125.33;  double deposits = 23;  double withdraws = 800.94; |
| **Process** | none |  |
| **Output** | savings balance at end of the month | double endBalance = 347.39; |
| **Algorithm** | 1. calculate the end balance by subtracting withdraws and adding deposits to beginning balance 2. display the end balance | endBalance = beginBalance – withdraws + deposits;  cout << “The new balance: $ “  cout << endBalance << endl; |

12)

Desk check #1

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | length  width  height | double length = 20.5;  double width = 10.5;  double height = 12.5; |
| **Process** | convert from cubic inches to gallons by dividing by 231 | int conversion = 231; |
| **Output** | number of gallons of water | double gallons = 11.6477; |
| **Algorithm** | 1. calculate how many gallons are in the aquarium by multiplying length by width by height and dividing by the conversion 2. display the number of gallons | gallons = length \* width \* height / conversion;  cout << “The aquarium can hold “ << gallons << “ gallons of water” << endl; |

Desk check #2

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | length  width  height | double length = 30;  double width = 9;  double height = 14; |
| **Process** | convert from cubic inches to gallons by dividing by 231 | int conversion = 231; |
| **Output** | number of gallons of water | double gallons = 16.3636; |
| **Algorithm** | 1. calculate how many gallons are in the aquarium by multiplying length by width by height and dividing by the conversion 2. display the number of gallons | gallons = length \* width \* height / conversion;  cout << “The aquarium can hold “ << gallons << “ gallons of water” << endl; |

13)

Desk check #1

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | length (in feet)  width (in feet)  price of tile (per square foot) | double length = 10.0;  double width = 5.0;  double pricePerSqrFt = 3.25; |
| **Process** | none |  |
| **Output** | area of floor  total cost | double area = 50.0;  double totalCost = 162.50; |
| **Algorithm** | 1. calculate the area of the floor by … 2. calculate the total cost by multiplying the area of the floor by the price of tile per square foot 3. display the area of the floor and the total cost | area = length \* width;  totalCost = area \* pricePerSqrFt;  cout << “The area of the floor is “ << area << “ square feet and the total cost for tile is $” << totalCost << endl; |

Desk check #2

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | length (in feet)  width (in feet)  price of tile (per square foot) | double length = 17.5;  double width = 12.0;  double pricePerSqrFt = 4.75; |
| **Process** | none |  |
| **Output** | area of floor  total cost | double area = 210.0;  double totalCost = 997.50; |
| **Algorithm** | 1. calculate the area of the floor by … 2. calculate the total cost by multiplying the area of the floor by the price of tile per square foot 3. display the area of the floor and the total cost | area = length \* width;  totalCost = area \* pricePerSqrFt;  cout << “The area of the floor is “ << area << “ square feet and the total cost for tile is $” << totalCost << endl; |

14)

Desk check #1

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | number of small pizzas sold  number of medium pizzas sold  number of large pizzas sold  number of family pizzas sold | double smallPizzasSold = 25.0;  double mediumPizzasSold = 50.0;  double largePizzasSold = 50.0;  double familyPizzasSold = 75.0; |
| **Process** | none |  |
| **Output** | total pizzas sold  percentage of small pizzas sold  percentage of medium pizzas sold  percentage of large pizzas sold  percentage of family pizzas sold | int totalPizzasSold = 200;  double smallPizzaPercentage = 12.5;  double mediumPizzaPercentage = 25.0;  double largePizzaPercentage = 25.0;  double familyPizzaPercentage = 37.5; |
| **Algorithm** | 1. calculate total number of pizzas sold by adding together the number of small, medium, large, and family pizzas sold 2. calculate percentage of small pizzas sold by dividing the number of small pizzas sold by the total number of pizzas sold 3. calculate percentage of small pizzas sold by dividing the number of small pizzas sold by the total number of pizzas sold 4. calculate percentage of small pizzas sold by dividing the number of small pizzas sold by the total number of pizzas sold 5. calculate percentage of small pizzas sold by dividing the number of small pizzas sold by the total number of pizzas sold 6. display the total number of pizzas sold and the percentage of each of the different sized pizzas sold | totalPizzasSold = smallPizzasSold + mediumPizzasSold + largePizzasSold + familyPizzasSold;  smallPizzaPercentage = smallPizzasSold / totalPizzasSold;  mediumPizzaPercentage = mediumPizzasSold / totalPizzasSold;  largePizzaPercentage = largePizzasSold / totalPizzasSold;  familyPizzaPercentage = familyPizzasSold / totalPizzasSold;  cout << “The total number of pizzas sold was “ << totalPizzasSold << endl;  cout <<”The percentage of small pizzas sold was “ << smallPizzaPercentage << endl;  cout <<”The percentage of medium pizzas sold was “ << mediumPizzaPercentage << endl;  cout <<”The percentage of large pizzas sold was “ << largePizzaPercentage << endl;  cout <<”The percentage of family pizzas sold was “ << familyPizzaPercentage << endl; |

Desk check #2

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | number of small pizzas sold  number of medium pizzas sold  number of large pizzas sold  number of family pizzas sold | int smallPizzasSold = 30;  int mediumPizzasSold = 25;  int largePizzasSold = 85;  int familyPizzasSold = 73; |
| **Process** | none |  |
| **Output** | total pizzas sold  percentage of small pizzas sold  percentage of medium pizzas sold  percentage of large pizzas sold  percentage of family pizzas sold | int totalPizzasSold = 213;  double smallPizzaPercentage = 14.1;  double mediumPizzaPercentage = 11.7;  double largePizzaPercentage = 39.9;  double familyPizzaPercentage = 34.3; |
| **Algorithm** | 1. calculate total number of pizzas sold by adding together the number of small, medium, large, and family pizzas sold 2. calculate percentage of small pizzas sold by dividing the number of small pizzas sold by the total number of pizzas sold then multiplying by 100 percent 3. calculate percentage of small pizzas sold by dividing the number of small pizzas sold by the total number of pizzas sold then multiplying by 100 percent 4. calculate percentage of small pizzas sold by dividing the number of small pizzas sold by the total number of pizzas sold then multiplying by 100 percent 5. calculate percentage of small pizzas sold by dividing the number of small pizzas sold by the total number of pizzas sold then multiplying by 100 percent 6. display the total number of pizzas sold and the percentage of each of the different sized pizzas sold | totalPizzasSold = smallPizzasSold +  mediumPizzasSold + largePizzasSold + familyPizzasSold;  smallPizzaPercentage = smallPizzasSold / totalPizzasSold;  mediumPizzaPercentage = mediumPizzasSold / totalPizzasSold;  largePizzaPercentage = largePizzasSold / totalPizzasSold;  familyPizzaPercentage = familyPizzasSold / totalPizzasSold;  cout << “The total number of pizzas sold was “ << totalPizzasSold << endl;  cout <<”The percentage of small pizzas sold was “ << smallPizzaPercentage << endl;  cout <<”The percentage of medium pizzas sold was “ << mediumPizzaPercentage << endl;  cout <<”The percentage of large pizzas sold was “ << largePizzaPercentage << endl;  cout <<”The percentage of family pizzas sold was “ << familyPizzaPercentage << endl; |

15)

Desk check #1

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | number of individual members  number of dual members  number of family members  price of individual membership  price of dual membership  price of family membership | double numIndivMembers = 50.0;  double numDualMembers = 75.0;  double numFamMembers = 150.0;  const double indivMembPrice = 99.0;  const double dualMembPrice = 175.0;  const double famMembPrice = 225.0; |
| **Process** | individual membership revenue ($4,950)  dual membership revenue ($13,125)  family membership revenue ($33,750) | double indivMembRevenue = 0.0;  double dualMembRevenue = 0.0;  double famMembRevenue = 0.0; |
| **Output** | total membership revenue ($51,825)  percentage of revenue made from individual membership (9.6%)  percentage of revenue made from dual membership (25.3%)  percentage of revenue made from family membership (65.1%) | double totalMembRevenue = 0.0;  double indivMembRevPercent = 0.0;  double dualMembRevPercent = 0.0;  double famMembRevPercent = 0.0; |
| **Algorithm** | 1. calculate the revenue made by each membership by multiplying their respective prices by their respective number of members 2. calculate the total membership revenue by adding together all of the revenue made by each membership 3. calculate the percentage of revenue generated by each membership by dividing their respective revenues by the total membership revenue generated then multiply by 100 percent 4. display the total amount of revenue generated by the memberships combined and display the percentage that each membership contributed towards the overall revenue | indivMembRevenue = numIndivMembers \* indivMembPrice;  dualMembRevenue = numDualMembers \* dualMembPrice;  famMembRevenue = numFamMembers \* famMembPrice;  totalMembRevenue = indivMembRevenue + dualMembRevenue + famMembRevenue;  indivMembRevPercent = indivMembRevenue / totalMembRevenue \* 100;  dualMembRevPercent = dualMembRevenue / totalMembRevenue \* 100;  famMembRevPercent = famMembRevenue / totalMembRevenue \* 100;  cout << “The total amount of revenue generated by all memberships combined is $” << totalMembRevenue << endl;  cout << “The individual membership generated $“ << indivMembRevenue << “ accounting for “ << indivMembRevPercent << “ percent of overall revenue” << endl;  cout << “The dual membership generated $“ << dualMembRevenue << “ accounting for “ << dualMembRevPercent << “ percent of overall revenue” << endl;  cout << “The family membership generated $“ << famMembRevenue << “ accounting for “ << famMembRevPercent << “ percent of overall revenue” << endl; |

Desk check #2

|  |  |  |
| --- | --- | --- |
|  | **IPO chart** | **Coding instructions** |
| **Input** | number of individual members  number of dual members  number of family members  price of individual membership  price of dual membership  price of family membership | double numIndivMembers = 35.0;  double numDualMembers = 150.0;  double numFamMembers = 265.0;  const double indivMembPrice = 99.0;  const double dualMembPrice = 175.0;  const double famMembPrice = 225.0; |
| **Process** | individual membership revenue ($3,465)  dual membership revenue ($26,250)  family membership revenue ($59,625) | double indivMembRevenue = 0.0;  double dualMembRevenue = 0.0;  double famMembRevenue = 0.0; |
| **Output** | total membership revenue ($89,340)  percentage of revenue made from individual membership (3.9%)  percentage of revenue made from dual membership (29.4%)  percentage of revenue made from family membership (66.7%) | double totalMembRevenue = 0.0;  double indivMembRevPercent = 0.0;  double dualMembRevPercent = 0.0;  double famMembRevPercent = 0.0; |
| **Algorithm** | 1. calculate the revenue made by each membership by multiplying their respective prices by their respective number of members 2. calculate the total membership revenue by adding together all of the revenue made by each membership 3. calculate the percentage of revenue generated by each membership by dividing their respective revenues by the total membership revenue generated then multiply by 100 percent 4. display the total amount of revenue generated by the memberships combined and display the percentage that each membership contributed towards the overall revenue | indivMembRevenue = numIndivMembers \* indivMembPrice;  dualMembRevenue = numDualMembers \* dualMembPrice;  famMembRevenue = numFamMembers \* famMembPrice;  totalMembRevenue = indivMembRevenue + dualMembRevenue + famMembRevenue;  indivMembRevPercent = indivMembRevenue / totalMembRevenue \* 100;  dualMembRevPercent = dualMembRevenue / totalMembRevenue \* 100;  famMembRevPercent = famMembRevenue / totalMembRevenue \* 100;  cout << “The total amount of revenue generated by all memberships combined is $” << totalMembRevenue << endl;  cout << “The individual membership generated $“ << indivMembRevenue << “ accounting for “ << indivMembRevPercent << “ percent of overall revenue” << endl;  cout << “The dual membership generated $“ << dualMembRevenue << “ accounting for “ << dualMembRevPercent << “ percent of overall revenue” << endl;  cout << “The family membership generated $“ << famMembRevenue << “ accounting for “ << famMembRevPercent << “ percent of overall revenue” << endl; |