|  |  |
| --- | --- |
| Edith Cowan University School of Science |  |

Workshop 9

C++ Class Features and Design Issues

**Related Objectives:**

* Classify the roles of member functions
* Understand constructors
* Write constructors both with and without arguments
* Overload constructors
* Create destructors

**Activity:**

1. Using the constructor below, write a class and application around this to be able to provide the output examined below:

Pizza::Pizza(const string top, const int size, const double price) {

topping = top;

diameter = size;

this->price = price;

}

Output:

The standard pizza is: a 12 inch cheese pizza. Price $8.99

Let me take your order

Do you want the standard pizza – y or n? y

Your order is a 12 inch cheese pizza. Price $8.99

1. Amend the Pizza class to use the following main() function. The desired output is examined below:

int main() {

Pizza stdPizza;

Pizza special(“pineapple”);

Pizza deluxeSpecial(“sausage”, 16);

Pizza veryDeluxeSpecial(“lobster”, 20, 17.99);

cout << “The standard pizza is: “;

stdPizza.displayValues();

cout << “Today’s special is: “;

special.displayValues();

cout << “The deluxe special is: “;

deluxeSpecial.displayValues();

cout << “And the very deluxe special is: “;

veryDeluxeSpecial.displayValues();

}

Output:

The standard pizza is: a 12 inch cheese pizza. Price $8.99

Today’s special is: a 12 inch pineapple pizza. Price $8.99

The deluxe special is: a 16 inch sausage pizza. Price $8.99

And the very deluxe special is: a 20 inch lobster pizza. Price $17.99

1. Given the following source code, write and compile the main() function that will demonstrate the use of object composition. The output is examined below the source:

Class 1 (Name):

class Name {

private:

string first;

string middle;

string last;

public:

Name(string, string, string);

void displayFullName();

};

Name::Name(string first, string middle, string last) {

this->first = first;

this->middle = middle;

this->last = last;

}

void Name::displayFullName() {

cout << first << “ “ << middle << “ “ << last << endl;

}

Class 2 (CreditData):

class CreditData {

private:

double currentBalance;

double maxBalance;

public:

CreditData(double, double = 0);

void displayCreditData();

};

CreditData::CreditData(double currBal, double maxBal) {

currentBalance = currBal;

if (maxBal < currBal)

maxBalance = currBal;

else

maxBalance = maxBal;

}

void CreditData::displayCreditData() {

double creditLeft = maxBalance – currentBalance;

cout << “Current balance: $” << currentBalance << “\nMaximum balance: $” << maxBalance << “\nCredit left: $” << creditLeft << endl;

}

Class 3 (Customer):

class Customer {

private:

Name name;

CreditData credit;

string phoneNumber;

public:

Customer(string, string, string, double, double, string);

void showCustomer();

};

Customer::Customer(string firstName, string middleName, string lastName, double bal, double max, string phone) : name(firstName, middleName, lastName), credit(bal, max){

phoneNumber = phone;

}

void Customer::showCustomer() {

cout << “Customer data:” << endl;

name.displayFullName();

cout << phoneNumber << endl;

credit.displayCreditData();

}

Example Output:

Please enter first name for customer: John

Please enter middle name: A

Please enter last name: Smith

Enter current balance: $100

Enter credit limit: $1000

Enter phone number: +61412345678

Customer data:

John A Smith

+61412345678

Current balance: $100

Maximum balance: $1000

Credit left: $900