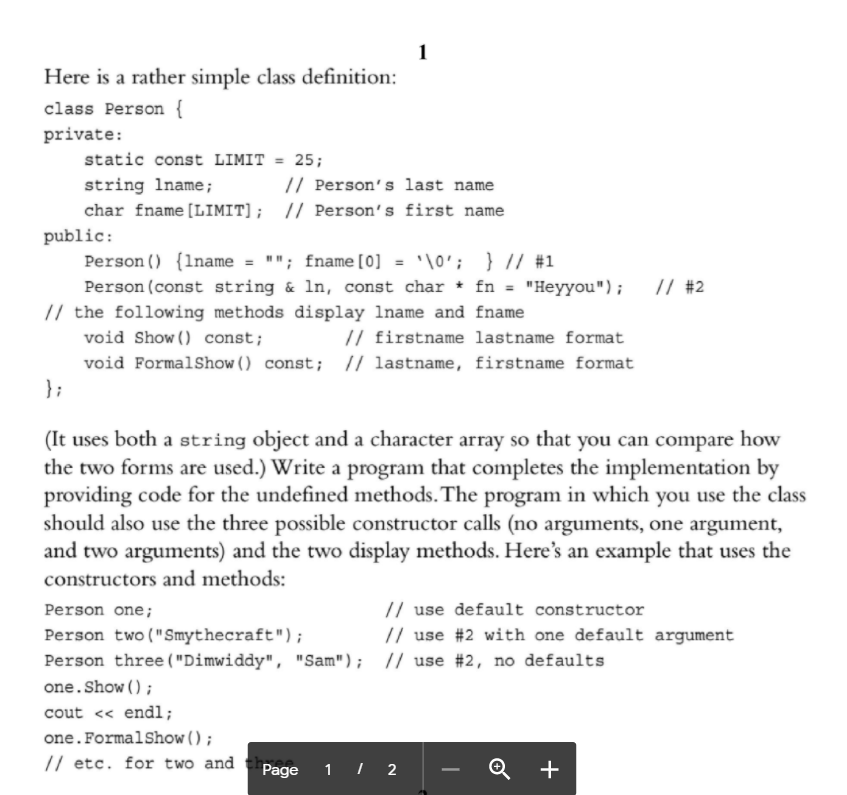
Rudenko Ruslan, SE-TE 2.01

Task:



Code:

#include <iostream>

#include <string>

using namespace std;

class Person {

private:

static const int LIMIT = 25;

std::string lname; // Person's last name

char fname[LIMIT]; // Person's first name

public:

Person() {

lname = "";

fname[0] = '\0';

std::cout << "\nUsing default constructor.\nInitialisation done.\n";

} // #1

Person(const std::string& ln, const char\* fn = "Hey you"); // #2

// the following methods display lname and fname

void Show() const; // first name last name format

void FormalShow() const; // last name, first name format

};

Person::Person(const std::string& ln, const char\* fn) {

lname = ln;

strcpy(fname, fn);

std::cout << "\nUsing constructor with parameters.\nInitialisation done.\n";

}

void Person::Show() const {

std::cout << "\nMethod Show(): " << this->fname << " " << this->lname << std::endl;

} // first name last name format

void Person::FormalShow() const {

std::cout << "\nMethod FormalShow(): " << this->lname << " " << this->fname << std::endl;

}// last name, first name format

int main() {

cout << "Setting object with constructor by default.";

Person my\_person;

my\_person.Show();

my\_person.FormalShow();

string s1 = "John";

char\* ch = (char \*)"Doe";

cout << "\nInitializing object with 1 parameter.\n";

Person my\_person2(s1);

my\_person2.Show();

my\_person2.FormalShow();

cout << "\nInitializing object with 2 parameters.\n";

Person my\_person3(s1, ch);

my\_person3.Show();

my\_person3.FormalShow();

system("pause");

}

Results

