## Лабораторна робота №2. Перевантаження операцій.

**Мета.** Одержати практичні навички створення абстрактних типів даних і перевантаження операцій у мові С++.

**Основний зміст роботи.**

Визначити і реалізувати клас — абстрактний тип даних. Визначити і реалізувати операції над даними цього класу. Написати і виконати EasyWin програму повного тестування цього класу.

**Умова:**

Варіанти завдань для виконання роботи.

Постановка задачі: Розширити функціональність класу, реалізовану в попередній роботі, згідно тогож варіанту.

**10. Додати в клас для роботи з датою:**

* + Перевантаження:
    - операції !: повертає значення true, якщо встановлена дата не є останнім днем місяця, інакше false;
    - констант true і false: звернення до екземпляра класу дає значення true, якщо встановлена дата є початком року, інакше false;
    - операції &: повертає значення true, якщо поля двох об'єктів рівні, інакше false;
    - перетворення класу DataTime в тип string (і навпаки).

**Код програми:**

|  |  |
| --- | --- |
| **LR2\_01.cpp** #include "DataTimes.h"  #include "tchar.h"  #include <iostream>  using namespace std;  int \_tmain(int argc, \_TCHAR\* argv[])  {  int n;  setlocale(LC\_ALL, "");  cout << "Введiть кiлькiсть об'єктiв:";  cin >> n;  DataTimes ob[100];  for (int i = 0; i < n; i++)  {  cout << "Введiть об'єкт №" << i + 1 << "\n";  ob[i].Input();  cout << "Кiлькiсть днiв до кiнця мiсяця = " << ob[i].CountDay() << "\n";  DataTimes tmp;  tmp.PrevDay(ob[i]);  cout << "Дата попереднього дня:";  tmp.Print();  tmp.NextDay(ob[i]);  cout << "Дата наступного дня:";  tmp.Print();  if (!ob[i]) cout << "Дата не останній день мiсяця\n";  else cout << "Дата є останнiм днем мiсяця\n";  if (ob[i]) cout << "Це 1 сiчня!\n";  else cout << "Це не 1 сiчня.\n";  if (i > 0) {  if (ob[i] & ob[i - 1])  cout << "Ця дата однакова з попередньою\n";  else  cout << "Ця дата інша\n";  }  string s = ob[i];  cout << "Перетворення в string: " << s << "\n";  }  system("pause");  return 0;  }  **DataTimes.h** #pragma once  #include <string>  #include <iostream>  #include <sstream>  using namespace std;  class DataTimes  {  public:  unsigned short int dd;  unsigned short int mm;  unsigned int yyyy;  public:  DataTimes();  DataTimes(unsigned short int d, unsigned short int m, unsigned int y);  explicit DataTimes(const std::string& str);  ~DataTimes();  void setAll(unsigned short int d, unsigned short int m, unsigned int y);  int getYear();  int getMounth();  int getDay();  DataTimes Input();  void Print();  DataTimes NextDay(DataTimes ob);  DataTimes PrevDay(DataTimes ob);  int CountDay();  bool operator!() const;  operator bool() const;  bool operator&(const DataTimes& other) const;  operator std::string() const;  };  **DataTimes.cpp**  #include "DataTimes.h"  #include <iostream>  #include <sstream>  #include <iomanip>  using namespace std;  DataTimes::DataTimes() :dd(1), mm(1), yyyy(2009) {};  DataTimes::DataTimes(unsigned short int d, unsigned short int m, unsigned int y) {  dd = d;  mm = m;  yyyy = y;  }  DataTimes::DataTimes(const std::string& str) {  char dot;  std::istringstream iss(str);  iss >> dd >> dot >> mm >> dot >> yyyy;  }  DataTimes::~DataTimes(void) {}  void DataTimes::setAll(unsigned short int d, unsigned short int m, unsigned int y) {  dd = d; mm = m; yyyy = y;  }  int DataTimes::getYear() { return yyyy; }  int DataTimes::getMounth() { return mm; }  int DataTimes::getDay() { return dd; }  DataTimes DataTimes::Input() {  setlocale(0, "UKR");  cout << "Введiть рiк:"; cin >> yyyy;  cout << "Введiть мiсяць:"; cin >> mm;  if (mm <= 12 && mm > 0) {  cout << "Введiть день:"; cin >> dd;  if (dd > 0 && dd <= 31) {  setlocale(0, "");  return DataTimes(dd, mm, yyyy);  }  else {  cout << "Введено не коректний день!" << endl;  exit(0);  }  }  else {  cout << "Введено не коректний мiсяць!" << endl;  exit(0);  }  } | DataTimes DataTimes::NextDay(DataTimes ob) {  this->dd = ob.dd;  this->mm = ob.mm;  this->yyyy = ob.yyyy;  this->dd++;  if ((mm == 1) || (mm == 3) || (mm == 5) || (mm == 7) || (mm == 8) || (mm == 10) || (mm == 12)) {  if (dd > 31) { dd = 1; mm++; }  }  else {  if ((mm == 4) || (mm == 6) || (mm == 9) || (mm == 11)) {  if (dd > 30) { dd = 1; mm++; }  }  else {  if (((yyyy % 4 == 0) && (yyyy % 100 != 0)) || (yyyy % 400 == 0)) {  if (dd > 29) { dd = 1; mm++; }  }  else {  if (dd > 28) { dd = 1; mm++; }  }  }  }  if (mm > 12) { mm = 1; yyyy++; }  return DataTimes(dd, mm, yyyy);  }  DataTimes DataTimes::PrevDay(DataTimes ob) {  this->dd = ob.dd;  this->mm = ob.mm;  this->yyyy = ob.yyyy;  this->dd--;  if ((mm == 1) || (mm == 3) || (mm == 5) || (mm == 7) || (mm == 8) || (mm == 10) || (mm == 12)) {  if (dd < 0) {  dd = 30; mm--;  if ((yyyy % 4 == 0 && yyyy % 100 != 0 && mm == 2) || (yyyy % 400 == 0 && mm == 2)) {  if (dd < 1) { dd = 29; }  }  else {  if (dd < 1) { dd = 28; }  }  }  }  else {  if ((mm == 2) || (mm == 4) || (mm == 6) || (mm == 9) || (mm == 11)) {  if (dd < 0) { dd = 31; mm--; }  }  }  if ((dd < 1) && (mm == 1)) {  dd = 31; mm = 12; yyyy--;  }  return DataTimes(dd, mm, yyyy);  }  int DataTimes::CountDay() {  switch (mm) {  case 1:case 3:case 5:case 7:case 8:case 10:case 12: return 31 - dd;  case 4:case 6:case 9:case 11: return 30 - dd;  case 2:  if ((yyyy % 4 == 0 && yyyy % 100 != 0) || (yyyy % 400 == 0))  return 29 - dd; else return 28 - dd;  default: return 0;  }  }  void DataTimes::Print() {  printf("%02d.%02d.%04d\n", dd, mm, yyyy);  }  bool DataTimes::operator!() const {  int daysInMonth;  switch (mm) {  case 1: case 3: case 5: case 7: case 8: case 10: case 12: daysInMonth = 31; break;  case 4: case 6: case 9: case 11: daysInMonth = 30; break;  case 2: daysInMonth = ((yyyy % 4 == 0 && yyyy % 100 != 0) || (yyyy % 400 == 0)) ? 29 : 28; break;  default: daysInMonth = 30;  }  return dd != daysInMonth;  }  DataTimes::operator bool() const {  return (dd == 1 && mm == 1);  }  bool DataTimes::operator&(const DataTimes& other) const {  return (dd == other.dd && mm == other.mm && yyyy == other.yyyy);  }  DataTimes::operator std::string() const {  std::ostringstream oss;  oss << std::setfill('0') << std::setw(2) << dd << "."  << std::setw(2) << mm << "."  << std::setw(4) << yyyy;  return oss.str();  } |

**Результат роботи програми:**

**Введiть кiлькiсть об'єктiв:2**

**Введiть об'єкт №1**

**Введiть рiк:2008**

**Введiть мiсяць:10**

**Введiть день:3**

**Кiлькiсть днiв до кiнця мiсяця = 28**

**Дата попереднього дня:02.10.2008**

**Дата наступного дня:04.10.2008**

**Дата не останн?й день мiсяця**

**Це не 1 сiчня.**

**Перетворення в string: 03.10.2008**

**Введiть об'єкт №2**

**Введiть рiк:2008**

**Введiть мiсяць:10**

**Введiть день:3**

**Кiлькiсть днiв до кiнця мiсяця = 28**

**Дата попереднього дня:02.10.2008**

**Дата наступного дня:04.10.2008**

**Дата не останн?й день мiсяця**

**Це не 1 сiчня.**

**Ця дата однакова з попередньою**

**Перетворення в string: 03.10.2008**

**Опис програми:**

Ця програма реалізує роботу з датами, дозволяючи вводити дату, визначати кількість днів до кінця місяця, знаходити попередній та наступний день, перевіряти особливі дати (1 січня або останній день місяця) та порівнювати дати між собою. Крім того, вона підтримує конвертацію дати у рядок для зручного виводу.