Лабораторна роботи №12

Створення нових типів даних на основі класів C++. Основи ООП. Перевантаження операцій, конструктори і деструктори.

Мета роботи: знайомство з основними поняттями та принципами об'єктно-орієнтованого програмування.

Завдання Реалізувати базовий та похідний класи згідно варіанта. В головній функції створити декілька екземплярів кожного класу та протестувати роботу усіх реалізованих функцій-членів класу.

Варіанти завдань

Варіант 1. Створити клас ПАРА ЦІЛИХ ЧИСЕЛ. Визначити потрібні конструктори, функції доступу до полів. Створити похідний клас ЗВИЧАЙНИЙ ДРІБ з полями: чисельник, знамнник. Визначити потрібні конструктори, деструктор, функції введення-виведення та оператори порівняння дробів.

Варіант 2. Створити клас ПАРА ЧИСЕЛ. Визначити потрібні конструктори, функції доступу до полів, введення-виведення даних. Створити похідний клас ПРЯМОКУТНИЙ ТРИКУТНИК з полями-катетами. Визначити конструктори з різним числом параметрів, деструктор, функції доступу до полів, введення-виведення, обчислення площі та гіпотенузи трикутника.

Варіант 3. Створити клас ТРІЙКА ЧИСЕЛ. Визначити конструктори, деструктор, функції доступу до полів, введення-виведення та обчислення суми чисел. Створити похідний клас ВЕКТОР з полями-координатами. Визначити конструктори із різним числом параметрів, деструктор, функції доступу до полів, введення-виведення. Перевантажити оператори додавання, віднімання та множення (скалярного) векторів.

Варіант 4. Створити клас ПАРА ЧИСЕЛ. Визначити потрібні конструктори, функції доступу до полів, введення-виведення. Створити похідний клас ГЕОМЕТРИЧНА ПРОГРЕСІЯ з полями: перший член та знаменник прогресії. Визначити конструктори і з різним числом параметрів, деструктор, функції доступу до полів, введення-виведення, обчислення суми для заданої кількості елементів прогресії.

Варіант 5. Створити клас ТРІЙКА ЧИСЕЛ. Визначити конструктори, функції доступу до полів, введення-виведення. Створити похідний клас ПАРАЛЕЛЕПІПЕД з полями-сторонами. Визначити конструктори і з різним

числом параметрів, деструктор, функції, введення-виведення, обчислення об’єму та площі поверхні.

Варіант 6. Створити клас ТРІЙКА ЦІЛИХ ЧИСЕЛ. Визначити конструктори, функції доступу до полів, введення-виведення чисел. Створити похідний клас ЧАС з полями: година, хвилина і секунда. Визначити конструктори і з різним числом параметрів, деструктор, функції доступу до полів, введення-виведення. В конструкторах та при вводі перевіряти коректність даних. Перевантажити оператори додавання та віднімання для двох моментів часу.

Варіант 7. Створити клас ПАРА ЦІЛИХ ЧИСЕЛ. Визначити потрібні конструктори, функції доступу до полів. Створити похідний клас ЗВИЧАЙНИЙ ДРІБ з полями: чисельник, знамнник. Визначити потрібні конструктори, деструктор, функції введення-виведення та оператори додавання, віднімання, множення та ділення дробів.

Варіант 8. Створити клас ТРІЙКА ЧИСЕЛ. Визначити конструктори, функції доступу до полів, введення-виведення. Створити похідний клас КВАДРАТНЕ РІВНЯННЯ з полями-коефіцієнтами. Визначити конструктори із різним числом параметрів, деструктор, функції доступу до полів, введення-виведення, обчислення коренів рівняння.

Варіант 9. Створити клас ПАРА ЧИСЕЛ. Визначити конструктори, деструктор, функції доступу до полів, введення-виведення та обчислення суми чисел. Створити похідний клас ВЕКТОР з полями-координатами. Визначити конструктори із різним числом параметрів, деструктор, функції доступу до полів, введення-виведення. Перевантажити оператори додавання, віднімання та множення (скалярного) векторів.

Варіант 10. Створити клас ТРІЙКА ЧИСЕЛ. Визначити потрібні конструктори, функції доступу до полів, та виведення на консоль. Створити похідний клас ТРКУТНИК з полями – довжинами сторін. Визначити конструктори із різним числом параметрів, деструктор, функції доступу до полів, введення-виведення. Перевантажити оператори порівняння двох трикутників, вважаючи, що більшим є трикутник, у якого більший периметр.

Варіант 11. Створити клас ПАРА ЧИСЕЛ. Визначити конструктори, функції доступу до полів, виведення чисел та обчислення їх добутку. Створити похідний клас ПРЯМОКУТНИК з полями-сторонами. Визначити конструктори із різним числом параметрів, деструктор, функції доступу до полів, введення-виведення, обчислення площі та периметра прямокутника.

Варіант 12. Створити клас ПАРА ЧИСЕЛ. Визначити потрібні конструктори, функції доступу до полів, виведення на консоль. Створити похідний клас АРИФМЕТИЧНА ПРОГРЕСІЯ з полями: перший елемент та різниця прогресії. Визначити конструктори із різним числом параметрів, деструктор, функції доступу до полів, введення-виведення, обчислення суми для заданої кількості елементів прогресії.

Варіант 13. Створити клас ТРІЙКА ЦІЛИХ ЧИСЕЛ. Визначити конструктори, функції доступу до полів, введення-виведення чисел, та оператор порівняння двох трійок на рівність. Створити похідний клас ЧАС з полями: година, хвилина і секунда. Визначити конструктори і з різним числом параметрів, деструктор, функції доступу до полів, введення-виведення. В конструкторах та при вводі перевіряти коректність даних. Перевантажити оператори порівняння для двох моментів часу.

Варіант 14. Створити клас ТРІЙКА ЦІЛИХ ЧИСЕЛ. Визначити конструктори, функції доступу до полів, та виводу на консоль. Створити похідний клас ДАТА з полями: рік, місяць і день. Визначити потрібні конструктори, деструктор, функції доступу до полів та вводу даних з клавіатури, перевантажити оператори порівняння дат.

Варіант 15. Створити клас ТРІЙКА ЧИСЕЛ. Визначити конструктори, деструктор, функції доступу до полів, введення-виведення та обчислення суми чисел. Створити похідний клас ВЕКТОР з полями-координатами. Визначити конструктори із різним числом параметрів, деструктор, функції доступу до полів, функції введення-виведення та функцію визначення модуля вектора. Перевантажити оператор множення для отримання векторного добутку двох векторів.

Приклад виконання завдання

Варіант 16. Створити клас ТРІЙКА ЦІЛИХ ЧИСЕЛ. Визначити конструктори, функції доступу до полів, та виводу на консоль. Створити похідний клас ДАТА з полями: рік, місяць і день. Визначити потрібні конструктори, деструктор, функції доступу до полів та вводу даних з клавіатури, перевантажити оператори порівняння дат.

```
#include <iostream>
#include <cmath>


using namespace std;

class int_triplet{
protected:
    int first, second, third;
public:
    int_triplet();
    int_triplet(int first, int second, int third);

    int get_first()
     {
      return first;
    }
    int get_second()
     {
      return second;
    }
    int get_third()
     {
      return third;
    }
    void print();
     void input(int first, int second, int third);

     //перевірка рівності наслідується похідними класами
     bool operator==(int_triplet other)
     {
          return other.first==this->first && other.second==this-
>second && other.third==this->third;
     }
     bool operator!=(int_triplet other)
     {
          return !(*this == other);
     }
};


class _date : public int_triplet{
```

```
public:
    _date() : int_triplet(1,1,1970){}
          //дата з одним числом - рік
    _date(int year) : int_triplet(1, 1, year){}
    _date(int, int) ;//місяць, рік
    _date(int, int, int ) ;
    ~_date()
    {
          // в деструкторі нема необхідності,
          //для деструктора нема роботи, бо
          //1) не виділяється пам'ять, яку потім треба чистити
          //2) не йде з'єднання з ресурсами, яке треба закривати
          //деструктор просто повідомляє, коли видаляється об'єкт
          //для перегляду результатів роботи краще закоментувати
          cout<<"Object date is deleted from memory... "
          <<"\nI'm sorry...(((\n";
    }

    int getDay()
    {// використовуємо метод базового класу
          return get_first();
    }
    int getMonth()
    {// використовуємо метод базового класу
          return get_second();
    }
    int getYear()
    {// використовуємо метод базового класу
          return get_third();
    }

    void print()
    {
          //перегружаємо вивід дати свій
          cout<<"#"<<first<<":"<<second<<":"<<third<<"#";
    }

    void input(int,int,int);//треба перегрузити з перевіркою
коректності
    void input();// перегрузка вводу, що вводить дату з консолі в
діалоговому режимі

    bool operator>(_date);
    bool operator<(_date other)
    {
          return other > *this;
    }
    //оператори == і != наслідуємо з базового класу

};
```

```
int main()
{
  int_triplet a(2,3,4), b(1,1,1);
        a.print();
        b.print();
        cout<<endl;
  if(a!=b)
  {
        a.print();
        cout<<" != ";
        b.print();
        cout<<endl;
  }

  b.input(2,3,4); //мають бути рівні
  if(a==b)
  {
        a.print();
        cout<<" == ";
        b.print();
        cout<<endl;
  }

  // робота з датами
  _date day1(20,12,2012), day2(20,12,2012);

  if(day1==day2)// оператор == наслідується від дати
  {
        day1.print();
        cout<<"==";
        day2.print();
        cout<<endl;
  }
  day2.input(11,03,2017);
  if(day1!=day2)// оператор == наслідується від дати
  {
        day1.print();
        cout<<"!=";
        day2.print();
        cout<<endl;
  }


// ввід дати з клавіатури

  day1.input();

  day1.print();
```

```
        if(day1<day2) cout<<" is before to ";
        if(day1>day2)  cout<<" is after to ";
        if(day1==day2)  cout<<" is equal to ";
        day2.print();
        cout<<endl;

}


// Реалізація членів класу "Три числа" int_triplet
int_triplet::int_triplet()
     {
          first=0;
          second=0;
          third=0;
    }

int_triplet::int_triplet(int first, int second, int third)
     {
      this->first = first;
      this->second = second;
      this->third = third;
    }

void int_triplet::print()
     {
          cout<<"{ "<<first<<"; "
               <<second<<"; "<<third<<" }";
     }


void int_triplet::input(int first, int second, int third)
     {
      this->first = first;
      this->second = second;
      this->third = third;
     }

//Конструктори дати будуть перевіряти, коректність даних

_date::_date(int month, int year)//місяць, рік
{
     first = 1;
     if(month>=1 && month<=12)
          second = month;
     else
          second=1;
     third = year;
}
```

```
_date::_date(int day, int month, int year)//день, місяць, рік
{
	if(month>=1 && month<=12)
		second = month;
	else
		second=month=1;
	switch(month)
	{
	case 2: if(year%4==0)
			{
				if(day>=1 && day<=29)first = day;
				else first=1;
			}
			else
			{
				if(day>=1 && day<=28)first = day;
				else first=1;
			}
			break;
	case 1:
	case 3:
	case 5:
	case 7:
	case 8:
	case 10:
	case 12:if(day>=1 && day<=31)first = day;
				else first=1;
			break;
	case 4:
	case 6:
	case 9:
	case 11:if(day>=1 && day<=31)first = day;
				else first=1;
			break;
	}
	third = year;
}

//ввід дати з перевіркою, аналогічно до коструктора
//можна зробити конструктор "день, місяць, рік" з викликом input
void _date::input(int day, int month, int year)
{
	if(month>=1 && month<=12)
		second = month;
	else
		second=month=1;
	switch(month)
	{
	case 2: if(year%4==0)
			{
				if(day>=1 && day<=29)first = day;
```

```
                    else first=1;
                }
                else
                {
                    if(day>=1 && day<=28)first = day;
                    else first=1;
                }
                break;
        case 1:
        case 3:
        case 5:
        case 7:
        case 8:
        case 10:
        case 12:if(day>=1 && day<=31)first = day;
                    else first=1;
                break;
        case 4:
        case 6:
        case 9:
        case 11:if(day>=1 && day<=31)first = day;
                    else first=1;
                break;
        }
        third = year;
}

void _date::input()
{
        int day, month, year;
        cout<<"Enter month ->";
        do
        {
        cin>>month;
        }while(month<1 || month>12);
        second = month;

        cout<<"Enter year ->";
        cin>>year;
        third = year;

        switch(month)
        {
        case 2: if(year%4==0)
                {
                    cout<<"Enter day ->";
                    do
                    {
                        cin>>day;
                    }while(day<1 || day>29);
```

```
                }
                else
                {
                        cout<<"Enter day ->";
                        do
                        {
                                cin>>day;
                        }while(day<1 || day>28);
                }
                break;
        case 1:
        case 3:
        case 5:
        case 7:
        case 8:
        case 10:
        case 12:cout<<"Enter day ->";
                        do
                        {
                                cin>>day;
                        }while(day<1 || day>31);
                break;
        case 4:
        case 6:
        case 9:
        case 11:cout<<"Enter day ->";
                        do
                        {
                                cin>>day;
                        }while(day<1 || day>30);
                break;
        }
        first = day;
}

bool _date::operator>(_date other)
{
        if(this->getYear()>other.getYear())return true;
        if(this->getYear()==other.getYear() &&
                this->getMonth()>other.getMonth() )return true;
        if(this->getYear()==other.getYear() &&
                this->getMonth()==other.getMonth() &&
                        this->getDay()>other.getDay() )return true;
        return false;
}
```