Практичне заняття 10. Статичний поліморфізм

Вправи до практичного заняття 10. Програмування операторів.

1. Точки площини визначено як структуру

```
// Програмований тип точок площини
struct Point
{
 double _x;
 double _y;
};
```

Операції порівняння точок площини визначаються так

```
// Операція порівняння точок площини на рівність
bool operator==(const Point &u, const Point &v)
{
 return(u._x==v._x)&&(u._y==v._y);
}
// Операція порівняння точок площини на нерівність
bool operator!=(const Point &u, const Point &v)
{
 return !(u==v);
}
```

Операції введення і виведення точок площини

```
// Операція виведення точок площини
ostream& operator<<(ostream &os, const Point &u)
{
 os<<'('<<u._x<<','<<u._y<<')';
 return os;
}
// Операція введення точок площини
istream& operator>>(istream &is, Point &u)
{
 is>>u._x>>u._y;
 return is;
}
```

Визначте точку тривимірного простору разом з операціями порівняння на рівність і нерівність, введення і виведення.

2. Комплексні числа визначено як структуру

```
// Програмований тип комплексних чисел
struct Complex
{
 double _re;
 double _im;
};
```

Реалізуйте операції порівняння комплексних чисел на рівність і нерівність, введення і виведення.

3. Реалізуйте операції додавання, віднімання, множення і ділення для комплексних чисел за зразком

```
// Три варіанти додавання комплексних чисел
const Complex operator+(const Complex &, const Complex &);
const Complex operator+(double, const Complex &);
const Complex operator+(const Complex &, double);
```

Що зміниться, якщо сигнатуру додавання визначити так:

```
// Три невірних варіанти додавання комплексних чисел
Complex operator+(const Complex &, const Complex &);
Complex operator+(double, const Complex &);
```

```
Complex operator+(const Complex &, double);
```

4. Реалізуйте присвоєння, суміщені з операціями додавання, віднімання, множення і ділення для комплексних чисел за зразком

```
// Присвоєння, суміщені з додаванням
Complex& operator+=(Complex & trgt, const Complex& src)
{
 trgt._x+=scr._x;
 trgt._y+=scr._y;
 return trgt;
}
Complex& operator+=(Complex & trgt, double x)
{
 trgt._x+=x;
 return trgt;
}
```

Проаналізуйте виконання команд (u,v,w — комплексні змінні)

```
 u+=v =w;
 u+=v+=w;
 (u+=v) =w;
 (u+=v)+=w;
```

Додому:

5. Запрограмуйте і порівняйте функції впорядкування масивів за кожною з трьох сигнатур

```
void sort(double * array, size_t n);
void sort(double * source, double * target, size_t n);
void sort(double * source, double ** target, size_t n);
```

6. Запрограмуйте узагальнену функцію піднесення до натурального степеня значення довільного підхожого типу за сигнатурою

```
template <typename T>
T power(const T& x, size_t n);
```

Застосуйте її для обчислення степенів цілих, дійсних і дійсних з подвоєною точністю чисел.

7. Дійсна функція дійсного аргументу f(x), задана на відрізку [a,b]. Запрограмуйте функцію

```
void tabulate (double (*const f)(double), const double a, const double b, const double
h);
```

табулювання (виводу значень) функції f(x) на заданому відрізку із заданим кроком h>0. Складіть і виконайте тестову програму табулювання тригонометричних функцій.

8. Запрограмувати узагальнені функції swap для випадку відсилок і указників. Скласти тестову програму для викликів конкретних і узагальненої функцій swap. Приклади конкретних функцій:

```
// Обміняти значення за відсилками дійсних
void swap(double &x, double &y)
{
 double z=x; x=y; y=z;
 return;
}
// Обміняти значення за указниками на дійсні
void swap(double *x, double *y)
{
 double z=*x; *x=*y; *y=z;
 return;
```

```
}
// Обміняти значення за відсилками цілих
void swap(int &x, int &y)
{
 x=x+y; y=x-y; x=x-y;
 return;
}
```

9. Необхідно розробити функції, які для кожного студента визначать його максимальний і середній бал залежно від кількості складеним ним іспитів, якщо максимальна кількість іспитів 5.

Варіант 1. Визначаємо п’ять функцій maxGrade

```
int maxGrade(int);
int maxGrade(int, int);
int maxGrade(int, int, int);
int maxGrade(int, int, int, int);
int maxGrade(int, int, int, int, int);
```

і записуємо п’ять реалізацій:

```
int maxGrade(int m)
{
 return m;
};
int maxGrade(int m1, int m2)
{
 return(m1>m2) ? m1 : m2;
};
int maxGrade(int m1, int m2, int m3)
{
 return maxGrade(m1, maxGrade(m2, m3));
};
int maxGrade(int m1, int m2, int m3, int m4)
{
 return maxGrade(m1, maxGrade(m2, m3, m4));
};
int maxGrade(int m1, int m2, int m3, int m4, int
m5)
{
 return maxGrade(m1, maxGrade(m2, m3, m4, m5));
};
```

Варіант 2. Оголошуємо замість п’яти попередніх одну функцію з п’ятьма параметрами, чотири з яких мають значення за замовчуванням:

```
int maxGrade(int m1,
 int m2=0, int m3=0, int m4=0, int m5=0);
```

Реалізуйте функцію з другого варіанту.

10. *Реалізуйте операції додавання комплексних чисел через операції присвоєння, суміщені з операціями додавання.

11. *Як застосувати узагальнену функцію піднесення до натурального степеня значення довільного підхожого типу за сигнатурою

```
 template <typename T>
 T power(const T& x, size_t n);
```

до комплексних чисел?