МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ

НАВЧАЛЬНО-НАУКОВИЙ КОМПЛЕКС «ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ» НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

КОМП’ЮТЕРНИЙ ПРАКТИКУМ №4

Варіант №18

З дисципліни: Програмування та алгоритмічні мови

Роботу виконав:

Студент 1 курсу групи КА-95

Петренко Денис

Перевірив:

Гуськова В.Г.

Київ-2020

1. Завдання:

Варіант 18. Визначити:

- арифметичний оператор “+” для класу «Число»;

- оператор порівняння “<” для «Число»;
- оператори порівняння “==” за вмістом для класу «Фраза»;
- оператори форматного уведення-виведення – для класів «Алфавіт» та «Фраза».

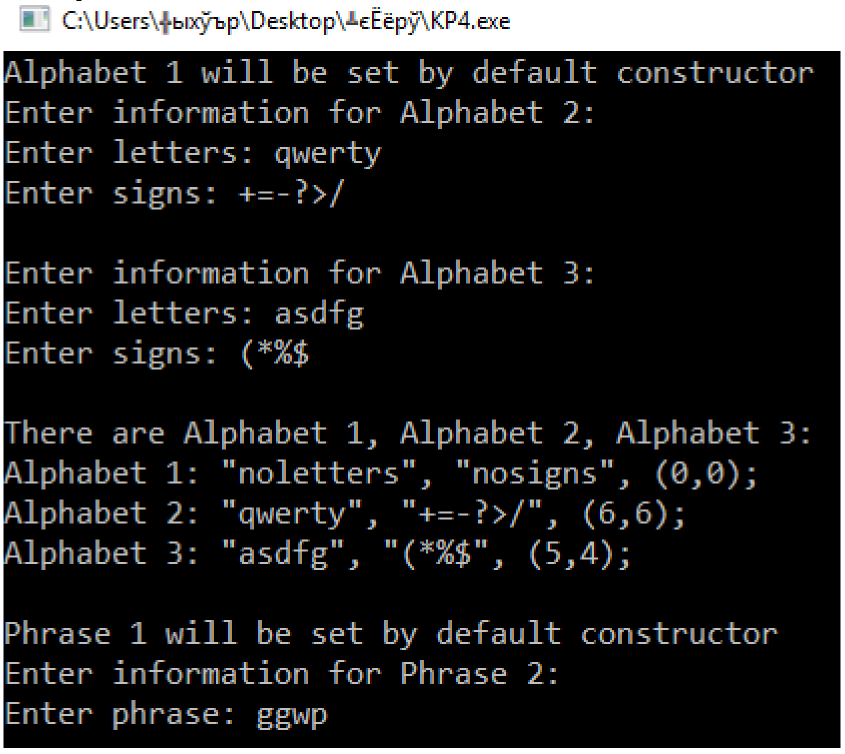
2.1 Лістинг програми:

```
#define _CRT_SECURE_NO_WARNINGS
#include <iostream>
using namespace std;

class Alphabet {
private:
   char* letters;
   char* signs;
   double l1, s1;

public:
   Alphabet();
   Alphabet(char*, char*, double, double);
   Alphabet(Alphabet&);
   ~Alphabet();
   friend ostream& operator<<(ostream& out, Alphabet& v);
   friend istream& operator>>(istream& in, Alphabet& v);
   Alphabet& setletters(char*);
   Alphabet& setsigns(char*);
   Alphabet& setl1(double);
   Alphabet& sets1(double);
   char* getletters();
   char* getsigns();
   double getl1();
   double gets1();
   void printAlphabet();
   void shortPrintAlphabet();
};
Alphabet::Alphabet() {
   letters = new char[50];
   signs = new char[50];
   strcpy(letters, "noletters");
   strcpy(signs, "nosigns");
   l1 = s1 = 0;
   //cout << "\n------------------------------Default constructor
Alphabet";
}
Alphabet::Alphabet(char* lett, char* sign, double l2, double s2) {
   letters = new char[strlen(lett) + 1];
   signs = new char[strlen(sign) + 1];
   strcpy(letters, lett);
   strcpy(signs, sign);
   l1 = l2;
   s1 = s2;
   //cout << "\n------------------------------Constructor Alphabet
with parametres";
}
Alphabet::Alphabet(Alphabet& a) {
   l1 = a.l1;
   s1 = a.s1;
   letters = new char[strlen(a.letters) + 1];
   signs = new char[strlen(a.signs) + 1];
   strcpy(letters, a.letters);
   strcpy(signs, a.signs);
   //cout << "\n------------------------------Constructor copy
Alphabet";
}
Alphabet::~Alphabet() { if (letters) delete[] letters; if (signs)
delete[] signs; /*cout << "\n------------------------------
Destructor Alphabet";*/ };
ostream& operator<<(ostream& out, Alphabet& v) {
  return out << '"' << v.letters << '"' << ',' << ' ' << '"' <<
v.signs << '"' << ',' << ' ' << '(' << v.l1 << ',' << v.s1 << ')' <<
';' << endl;
}
istream& operator>>(istream& in , Alphabet& v) {
  return in >> v.letters >> v.signs;
}
Alphabet& Alphabet::setletters(char* lett) {
   delete[]letters;
   letters = new char[strlen(lett) + 1];
   strcpy(letters, lett);
   return*this;
};
Alphabet& Alphabet::setsigns(char* sign) {
   delete[] signs;
   signs = new char[strlen(sign) + 1];
   strcpy(signs, sign);
   return*this;
};
Alphabet& Alphabet::setl1(double l2) { l1 = l2; return*this; };
Alphabet& Alphabet::sets1(double s2) { s1 = s2; return*this; };
char* Alphabet::getletters() { return letters; };
char* Alphabet::getsigns() { return signs; };
double Alphabet::getl1() { return l1; };
double Alphabet::gets1() { return s1; };
void Alphabet::printAlphabet() {
   cout << letters << ' ' << signs << ' ' << l1 << ' ' << s1 <<
endl;
};
void Alphabet::shortPrintAlphabet() {
   cout << l1 << ' ' << s1 << endl;
}
class Phrase {
private:
   char* phrase;
   Alphabet alph;
public:
   Phrase();
   Phrase(char*, Alphabet&);
   Phrase(Phrase&);
   ~Phrase();
   friend ostream& operator<<(ostream& out, Phrase& v);
   friend istream& operator>>(istream& in, Phrase& v);
   friend bool operator==(const Phrase& a, const Phrase& b);
   Phrase& setphrase(char*);
   Phrase& setalph(char*, char*, double, double);
   char* getphrase();
   Alphabet& getalph(char* gl1, char* gl2, double gl3, double gl4);
   void printPhrase();
   void shortPrintPhrase();
};
Phrase::Phrase() {
   phrase = new char[50];
   strcpy(phrase, "nophrase");
   //cout << "\n------------------------------Default constructor
Phrase";
}
Phrase::Phrase(char* phr, Alphabet& alpha) :alph(alpha) {
   phrase = new char[strlen(phr) + 1];
   strcpy(phrase, phr);
   //cout << "\n------------------------------Constructor Phrase
with parametres";
}
Phrase::Phrase(Phrase& b) : alph(b.alph) {
   phrase = new char[strlen(b.phrase) + 1];
   strcpy(phrase, b.phrase);
   //cout << "\n------------------------------Constructor copy
Phrase";
}
Phrase::~Phrase() { if (phrase) delete[] phrase; };
bool operator==(const Phrase& a, const Phrase& b) {
   return (a.phrase == b.phrase);
}
ostream& operator<<(ostream& out, Phrase& v) {
   return out << '"' << v.phrase << '"' << ';' << endl;
}
istream& operator>>(istream& in, Phrase& v) {
   return in >> v.phrase >> v.alph;
}
Phrase& Phrase::setphrase(char* phr) {
   delete[]phrase;
   phrase = new char[strlen(phr) + 1];
   strcpy(phrase, phr);
   return*this;
};
Phrase& Phrase::setalph(char* sl1, char* sl2, double sl3, double
sl4) {
   alph.setletters(sl1);
   alph.setsigns(sl2);
   alph.setl1(sl3);
   alph.sets1(sl4);
   return*this;
}
char* Phrase::getphrase() { return phrase; };
Alphabet& Phrase::getalph(char* gl1, char* gl2, double gl3, double
gl4) {
   alph.getletters();
   alph.getsigns();
   alph.getl1();
   alph.gets1();
   return alph;
}
void Phrase::printPhrase() {
   cout << phrase << endl;
   alph.printAlphabet();
};
void Phrase::shortPrintPhrase() {
   cout << phrase << endl;
}
class Number : public Phrase {
private:
   double number_system;
   double length_fraction;
public:
   Number();
   Number(char*, Alphabet&, double, double);
   Number(Number&);
   ~Number();
   friend Number operator+(Number&, Number&);
   friend bool operator<(Number&, Number&);
   friend ostream& operator<<(ostream& out, Number& v);
   Number& setnumber_system(double);
   Number& setlength_fraction(double);
   double getnumber_system();
   double getlength_fraction();
   void printNumber();
   void viewNumber();
};
Number::Number() {
   number_system = length_fraction = 0;
   //cout << "\n------------------------------Default constructor
Number\n";
}
Number::Number(char* phr, Alphabet& alpha, double ns, double lf)
:Phrase(phr, alpha) {
   number_system = ns;
   length_fraction = lf;
   //cout << "\n------------------------------Constructor Number
with parametres\n";
}
Number::Number(Number& n) : Phrase(n) {
   number_system = n.number_system;
   length_fraction = n.length_fraction;
   //cout << "\n------------------------------Constructor copy
Number\n";
}
```

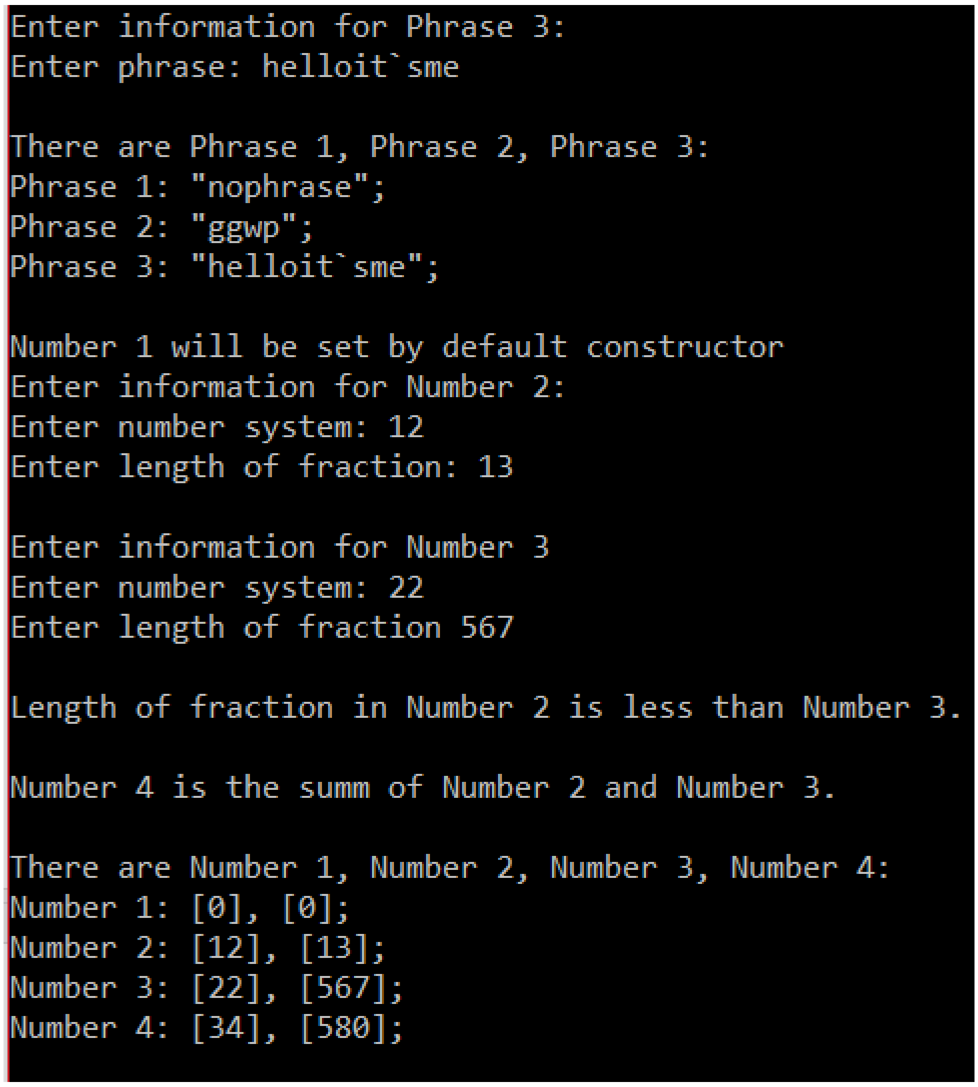
```
Number::~Number() {
   //cout << "\n------------------------------Destructor Number\n";
}
Number operator+(Number& a, Number& b) {
   Number temp;
   temp.number_system = a.number_system + b.number_system;
   temp.length_fraction = a.length_fraction + b.length_fraction;
   return temp;
}
bool operator<(const Number& l1, const Number& l2) {
   return (l1 < l2);
}
ostream& operator<<(ostream& out, Number& v) {
   return out << '[' << v.number_system << ']' << ',' << ' ' << '['
<< v.length_fraction << ']' << ';' << endl;
}
Number& Number::setnumber_system(double ns) {
   number_system = ns;
   return*this;
}
Number& Number::setlength_fraction(double lf) {
   length_fraction = lf;
   return*this;
}
double Number::getnumber_system() {
   return number_system;
}
double Number::getlength_fraction() {
   return length_fraction;
}
void Number::printNumber() {
   Phrase::printPhrase();
   cout << number_system << endl;
   cout << length_fraction << endl;
}
void Number::viewNumber() {
   Phrase::printPhrase();
   cout << number_system << endl;
}

class Sentence : public Phrase {
   double len_Alph;
   int ignor_register;
public:
   Sentence();
   Sentence(char*, Alphabet&, double, int);
   Sentence(Sentence&);
   ~Sentence();
   friend ostream& operator<<(ostream& out, Sentence& v);
   Sentence& setlen_Alph(double);
   Sentence& setignor_register(int);
   double getlen_Alph();
   int getignor_register();
   void printSentence();
   void viewSentence();
};
Sentence::Sentence() {
   len_Alph = ignor_register = 0;
   //cout << "\n------------------------------Default constructor 
Sentence\n";
}
Sentence::Sentence(char* phr, Alphabet& alpha, double lA, int ir)
:Phrase(phr, alpha) {
   len_Alph = lA;
   ignor_register = ir;
   //cout << "\n------------------------------Constructor Sentence 
with parametres\n";
}
Sentence::Sentence(Sentence& s) : Phrase(s) {
   len_Alph = s.len_Alph;
   ignor_register = s.ignor_register;
   //cout << "\n------------------------------Constructor copy 
Sentence\n";
}
Sentence::~Sentence() {
    //cout << "\n------------------------------Destructor 
Sentence\n";
}
ostream& operator<<(ostream& out, Sentence& v) {
   return out << '[' << v.len_Alph << ']' << ',' << ' ' << '(' << 
v.ignor_register << ')' << ';' << endl;
}
Sentence& Sentence::setlen_Alph(double lA) {
   len_Alph = lA;
   return*this;
}
Sentence& Sentence::setignor_register(int ir) {
   ignor_register = ir;
   return*this;
}
double Sentence::getlen_Alph() {
   return len_Alph;
}
int Sentence::getignor_register() {
   return ignor_register;
}
void Sentence::printSentence() {
   Phrase::printPhrase();
   cout << len_Alph << endl;
   cout << ignor_register << endl;
}
void Sentence::viewSentence() {
   Phrase::printPhrase();
   cout << len_Alph << endl;
}

int main()
{
   double l2, s2;
   char lett[50];
   char sign[50];
   char phr1[50];
   char phr2[50];
   double ns, lf1, lf2, lA;
   int ir;
   int choice;
   Alphabet obj1;
   cout << "Alphabet 1 will be set by default constructor" <<
"\nEnter information for Alphabet 2:" << endl;
   cout << "Enter letters: ";  cin >> lett;    l2 = strlen(lett);
   cout << "Enter signs: ";  cin >> sign;    s2 = strlen(sign);
   Alphabet obj2(lett, sign, l2, s2);
   Alphabet obj3 = obj2;
   cout << "\nEnter information for Alphabet 3:" << endl;
   cout << "Enter letters: ";  cin >> lett;    l2 = strlen(lett);
   cout << "Enter signs: ";  cin >> sign;    s2 = strlen(sign);
   obj3.setletters(lett).setsigns(sign).setl1(l2).sets1(s2);
   cout << "\nThere are Alphabet 1, Alphabet 2, Alphabet 3: " <<
"\nAlphabet 1: " << obj1 << "Alphabet 2: " << obj2 << "Alphabet 3: "
<< obj3 << endl;

   Phrase p1;
   cout << "Phrase 1 will be set by default constructor" << "\nEnter
information for Phrase 2: " << "\nEnter phrase: ";
   cin >> phr1;
   Phrase p2(phr1, obj3);
   Phrase p3 = p2;
   cout << "\nEnter information for Phrase 3: " << "\nEnter phrase:
";
   cin >> phr2;
   p3.setphrase(phr2).setalph(lett, sign, l2, s2);
   if (strcmp(phr1, phr2) == 0) {
       cout << "\nPhrase 2 and Phrase 3 are the same." << "\nThere
are Phrase 1 and Phrase 2: " << "\nPhrase 1: " << p1 << "Phrase 2: "
<< p2 << endl;
   }
   else {
        cout << "\nThere are Phrase 1, Phrase 2, Phrase 3: " <<
"\nPhrase 1: " << p1 << "Phrase 2: " << p2 << "Phrase 3: " << p3 <<
endl;
   }

   Number n1;
   cout << "Number 1 will be set by default constructor" << "\nEnter
information for Number 2:" << endl;
   cout << "Enter number system: ";      cin >> ns;
   cout << "Enter length of fraction: ";  cin >> lf1;
   Number n2(phr1, obj3, ns, lf1);
   Number n3 = n2;
   cout << "\nEnter information for Number 3" << endl;
   cout << "Enter number system: ";      cin >> ns;
   cout << "Enter length of fraction ";  cin >> lf2;
   n3.setnumber_system(ns).setlength_fraction(lf2);
   if (lf1 < lf2) {
        cout << "\nLength of fraction in Number 2 is less than
Number 3." << endl;
   }
   else {
        cout << "\nLength of fraction in Number 3 is less than
Number 2." << endl;
   }
   Number n4; n4 = n3 + n2;
   cout << "\nNumber 4 is the summ of Number 2 and Number 3." <<
endl;
```

```
   cout << "\nThere are Number 1, Number 2, Number 3, Number 4: " <<
"\nNumber 1: " << n1 << "Number 2: " << n2 << "Number 3: " << n3 <<
"Number 4: "<< n4 <<  endl;


   Sentence sen1;
   lA = l2 + s2;
   cout << "\nEnter Sentence: " << "\nEnter 0 or 1 in the value
whether to ignore case: ";  cin >> ir;
   Sentence sen2(phr1, obj3, lA, ir);
   Sentence sen3 = sen2;
   sen3.setlen_Alph(lA);
   cout << "\nThere are Sentence 1, Sentence 2, Sentence 3: " <<
"\nSentence 1: " << sen1 << "Sentence 2: " << sen2 << "Sentence 3: "
<< sen3 << endl;

   cout << "\nPrint Phrase";
   cout << "\n1. Print Phrase with number of system.";
   cout << "\n2. Print Phrase with lenght of alphabet.";
   cout << "\nEnter your choice: "; cin >> choice;
   switch (choice) {
   case 1: {
   n3.viewNumber();
    break;
   }
   case 2: {
   sen3.viewSentence();
    break;
   }
   }


return 0;
}
```

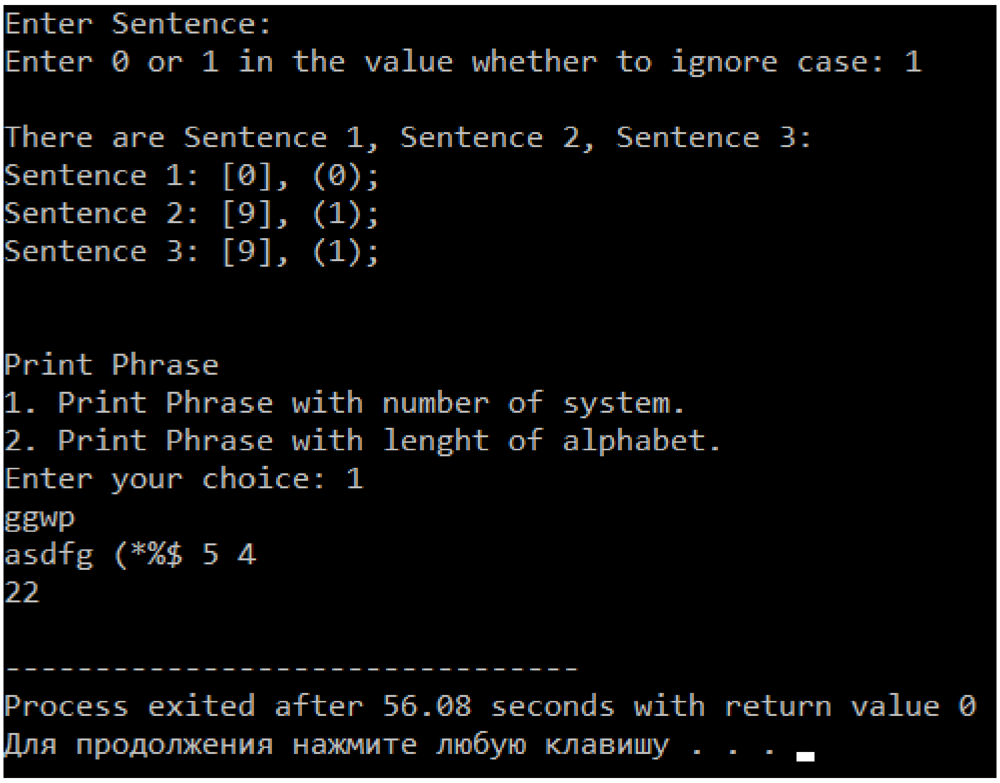
2.2 Результати:



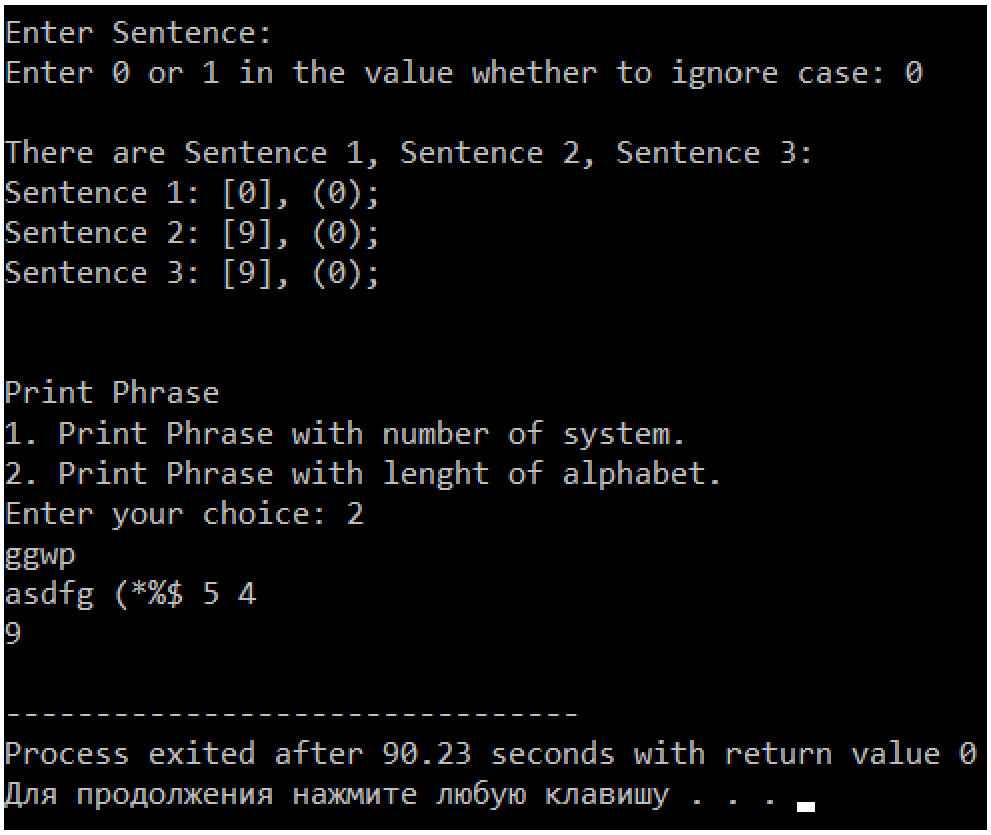
```
C:\Users\╬ыхўър\Desktop\╩єЁёр\KP4.exe
Alphabet 1 will be set by default constructor
Enter information for Alphabet 2:
Enter letters: qwerty
Enter signs: +=-?>/

Enter information for Alphabet 3:
Enter letters: asdfg
Enter signs: (*%$

There are Alphabet 1, Alphabet 2, Alphabet 3:
Alphabet 1: "noletters", "nosigns", (0,0);
Alphabet 2: "qwerty", "+=-?>/", (6,6);
Alphabet 3: "asdfg", "(*%$", (5,4);

Phrase 1 will be set by default constructor
Enter information for Phrase 2:
Enter phrase: ggwp
```



```
Enter information for Phrase 3:
Enter phrase: helloit`sme

There are Phrase 1, Phrase 2, Phrase 3:
Phrase 1: "nophrase";
Phrase 2: "ggwp";
Phrase 3: "helloit`sme";

Number 1 will be set by default constructor
Enter information for Number 2:
Enter number system: 12
Enter length of fraction: 13

Enter information for Number 3
Enter number system: 22
Enter length of fraction 567

Length of fraction in Number 2 is less than Number 3.

Number 4 is the summ of Number 2 and Number 3.

There are Number 1, Number 2, Number 3, Number 4:
Number 1: [0], [0];
Number 2: [12], [13];
Number 3: [22], [567];
Number 4: [34], [580];
```



```
Enter Sentence:
Enter 0 or 1 in the value whether to ignore case: 1

There are Sentence 1, Sentence 2, Sentence 3:
Sentence 1: [0], (0);
Sentence 2: [9], (1);
Sentence 3: [9], (1);


Print Phrase
1. Print Phrase with number of system.
2. Print Phrase with lenght of alphabet.
Enter your choice: 1
ggwp
asdfg (*%$ 5 4
22

--------------------------------
Process exited after 56.08 seconds with return value 0
Для продолжения нажмите любую клавишу . . .
```



```
Enter Sentence:
Enter 0 or 1 in the value whether to ignore case: 0

There are Sentence 1, Sentence 2, Sentence 3:
Sentence 1: [0], (0);
Sentence 2: [9], (0);
Sentence 3: [9], (0);


Print Phrase
1. Print Phrase with number of system.
2. Print Phrase with lenght of alphabet.
Enter your choice: 2
ggwp
asdfg (*%$ 5 4
9

--------------------------------
Process exited after 90.23 seconds with return value 0
Для продолжения нажмите любую клавишу . . .
```

Висновок:

Виконавши цю роботу я навчився правильно користуватися можливостями перевантаження звичайних і операторних функцій в С++.