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 №3

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:

Варіант18.

Базовий клас – «Фраза» похідні:

- «число», додаткові поля – система числення, довжина дробової частини, форма запису – з фіксованою або з плаваючою точкою,
- «речення», додаткові поля – кількість символів в алфавіті, чи ігнорувати регістр,
- визначити функцію `view()`: виведення самої фрази, або разом із значенням основи системи числення, або разом із кількістю символів в алфавіті.

У тестовому прикладі створити об'єкти похідних типів.

2.1 Лістинг програми:

```
#include <iostream>
#include <string.h>
#include <stdlib.h>
#include <ctype.h>
#define pedf "0"
#pragma warning(disable : 4996);
using namespace std;

class Alphabet
{
private:
 char* letters;
 char* symbols;
 int a;
 int s;
public:
 Alphabet()
 {
   cout << "This is standart constructor of Alphabet";
   letters = new char[30];
   strcpy(letters, "abcdml0");
   symbols = new char[30];
   strcpy(symbols, " ");
   con();
 }
```

```
Alphabet(char* pd)
{
  cout << "This is parameters constructor of Alphabet\n";
  letters = new char[30];
  symbols = new char[30];
  int k = 0;
  int n = 0;
  for (int i = 0; i < strlen(pd); i++)
  {
    if (isalnum(pd[i]))
    {
      if(pov(pd[i],letters))
      {
        letters[k] = pd[i];
      k++;
      }
    }
    else {
      if (pd[i] != ' ')
      {
        symbols[n] = pd[i];
        n++;
      }
    }
  }
  con();
}
Alphabet(const Alphabet& another)
{
  cout << "This is copying constructor of Alphabet\n";
  letters = new char[30];
  strcpy(letters, another.letters);
  symbols = new char[30];
  strcpy(symbols, another.symbols);
  con();
}
~Alphabet()
{
  cout << "Alphabet deleted\n";
  delete letters;
```

```
    delete symbols;
  }
  void con()
  {
    a = strlen(letters);
    s = strlen(symbols);
  }
  bool pov(char k, char* g)
  {
    for(int i = 0; i<strlen(g); i++)
    {
      if(k == g[i]) return 0;
    }
    return 1;
  }
  char* get_let() { return letters; }
  char* get_sym() { return symbols; }
  void set_let(char* ld) { strcpy(letters, ld); con(); }
  void set_sym(char* sd) { strcpy(symbols, sd); con(); }
  void print()
  {
    cout << "Alphabet with " << a << " letters: " << letters << " and
with " << s << " special symbols: " << symbols << "\n";
  }
  void prints()
  {
    cout << "With " << letters << " and " << symbols << "\n";
  }
};
class Phrase
{
private:
  char* phrase;
  Alphabet A;
public:
  Phrase() : A()
  {
    cout << "This is standart constructor of Phrase\n";
    phrase = new char[50];
    strcpy(phrase, "alabama0.");
```

```
 }
 Phrase(char* pd) : A(pd)
 {
   cout << "This is parameters constructor of Phrase\n";
   phrase = new char[50];
   strcpy(phrase, pd);
 }
 Phrase(const Phrase& other) : A(other.A)
 {
   cout << "This  is copiying constructor\n";
   phrase = new char[50];
   strcpy(phrase, other.phrase);
 }
 ~Phrase()
 {
   cout << "Deleted Phrase\n";
 }
 char* get_phrase() { return phrase; }
 char* get_letd() { return A.get_let(); }
 char* get_symd() { return A.get_sym(); }
 void set_phrase(char* pd) { strcpy(phrase, pd); }
 void set_let(char* ld) { A.set_let(ld); }
 void set_sym(char* sd) { A.set_sym(sd); }
 void print()
 {
   cout << "Phrase: " << phrase << " using alphabet " << "\n";
   A.print();
 }
 void prints()
 {
   cout << "Phrase: " << phrase << "\n";
   A.prints();
 }
};
class Chislo : public Phrase
{
private:
 int syst;
 int drob;
 bool fx;
```

```
public:
 Chislo() : Phrase()
 {
   cout << "This is standart constructor of Chislo\n";
   syst = 10;
   drob = 0;
   fx = 1;
   char st[2];
   strcpy(st, "0");
   set_phrase(st);
 }
 Chislo(char* pd, int sys, bool fxd) : Phrase(pd)
 {
   cout << "This is parameters constructor of Chislo\n";
   syst = sys;
   fx = fxd;
   if (fx) drob = drob1(pd);
   else drob = drob2(pd);
 }
 Chislo(const Chislo& other) : Phrase(other)
 {
   cout << "This is copying constructor of Chislo\n";
   syst = other.syst;
   fx = other.syst;
   drob = other.drob;
 }
 ~Chislo()
 {
   cout << "Chislo deleted\n";
 }
 int drob1(char* pd)
 {
   int m = 0;
   int k;
   k = strcspn(pd, ",");
   for (int i = k + 1; i < strlen(pd); i++) { m++; }
   return m;
 }
 int drob2(char* pd)
 {
```

```
    int k;
    char buf[30];
    k = strcspn(pd, "^");
    for (int i = 0;i < strlen(pd); i++) { buf[i] = pd[k]; k++; }
    return atoi(buf);
  }
  int get_syst() { return syst; }
  int get_drob() { return drob; }
  bool get_fx() { return fx; }
  int set_syst(int sysd) { syst = sysd; return syst; }
  bool set_fs(bool fxd) { fx = fxd; return fx; }
  int set_drob(int db) { drob = db; return drob; }
  void print1()
  {
    cout << "\n" << "Chislo" << "\n";
    print();
    cout << "Basis of the calculus system " << syst << " length of
fractional part is " << drob << "\n";
    if (fx) cout << "with fixed point \n";
    else cout << " with floting point\n";
  }
  void view()
  {
    cout << "\n" << "View\n";
    cout << "Chislo " << get_phrase() << " have basis of the calculus
system " << syst <<  "\n";
  }
};

class Rechenna : public Phrase
{
private:
  bool regis;
  int legth;
public:
  Rechenna() : Phrase()
  {
    cout << "This is standart constructor of Rechenna\n";
    regis = 0;
    legth = 6;
```

```
 }
 Rechenna(char* pd, bool rg) : Phrase(pd)
 {
   cout << "This is parameters constructor of Rechenna\n";
   regis = rg;
   legth = strlen(get_letd()) + strlen(get_symd());
 }
 Rechenna(const Rechenna& other) : Phrase(other)
 {
   cout << "This is copying constructor of Rechenna\n";
   regis = other.regis;
   legth = other.legth;
 }
 ~Rechenna()
 {
   cout << "Deleted Rechenna\n";
 }
 int get_legth() { return legth; }
 bool get_regis() { return regis; }
 int set_legth(int ld) { legth = ld; return legth; }
 bool set_regis(bool reg) { regis = reg; return regis; }
 void print1()
 {
   cout << "\n" << "Rechenna\n";
   print();
   cout << "There are " << legth << " symbols in alphabet\n";
   if (regis) cout << "Ignore register\n";
   else cout << "Not ignore register\n";
 }
 void view()
 {
   cout << "\n" << "View\n";
   cout << get_phrase() << " - have " << legth << " symbols\n";
 }
};

int main()
{
 int num;
 bool f, ig;
```

```
 char chis[30];
 char phr[50];
 cout << "In what numeral system is your chislo? (enter number), Does
it have floating point(1 - no, 0 - yes), and after all - chislo\n";
 cin >> num >> f >> chis;
 Chislo C1(chis, num, f);
 Chislo C2(C1);
 Chislo C3;
 cout << "To ignore register enter 1, not ignore - 0. Then - enter
phrase\n";
 cin >> ig ;
 cin.ignore();
 cin.getline(phr,50);
 Rechenna R1(phr, ig);
 Rechenna R2(R1);
 Rechenna R3;
 cout << "\n" << "Created by parameters\n";
 C1.print1();
 R1.print1();
 C1.view();
 R1.view();
 cout << "\n" << "Created by standart\n";
 C3.print1();
 R3.print1();
 C3.view();
 R3.view();
 cout << "\n" << "Created by copying of first\n";
 C2.print1();
 R2.print1();
 C2.view();
 R2.view();
 cout << "\n";
}
```

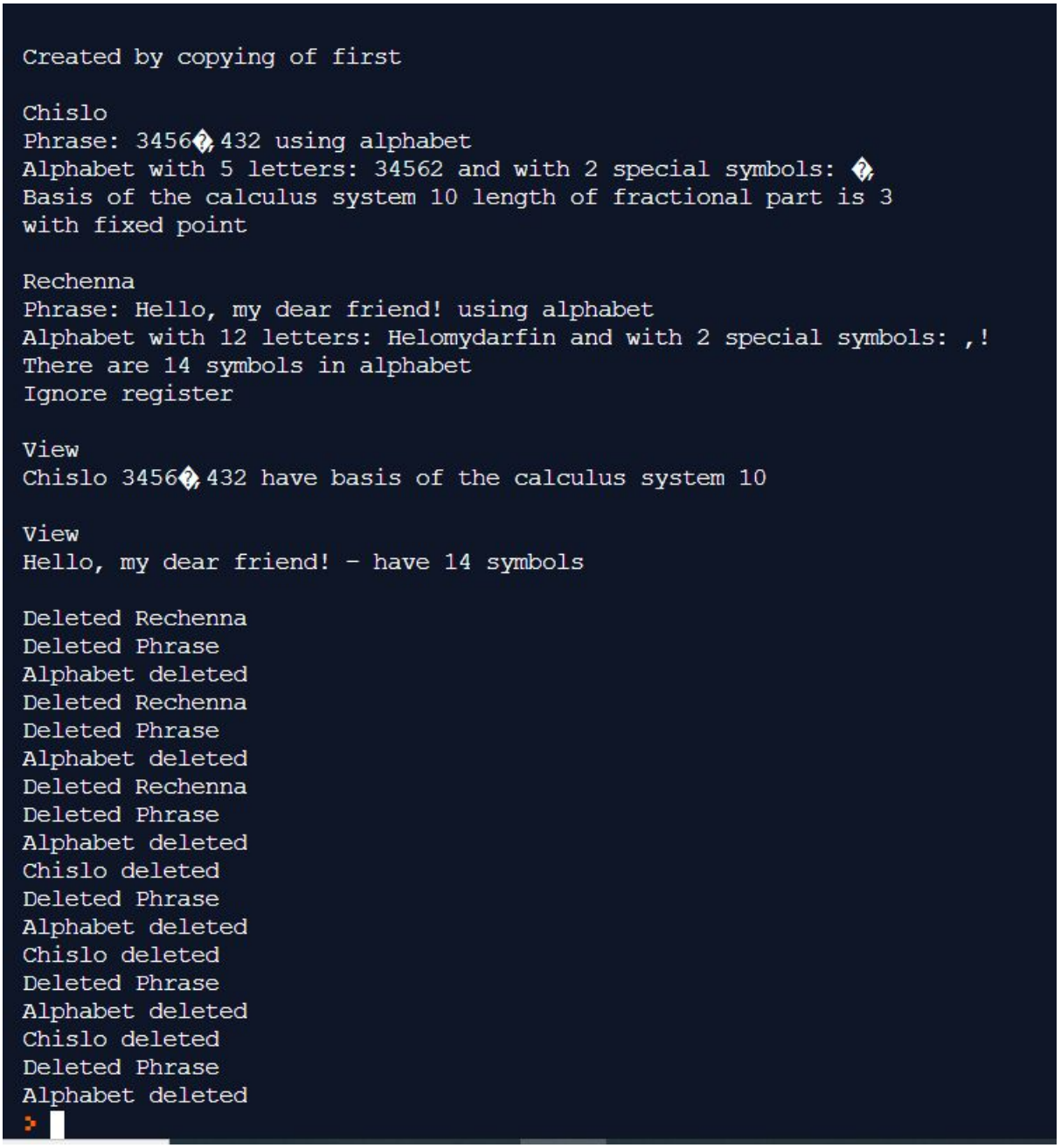
2.2 Результати:

```
clang version 7.0.0-3~ubuntu0.18.04.1 (tags/RELEASE_700/final)
> clang++-7 -pthread -o main main.cpp
> ./main
In what numeral system is your chislo? (enter number), Does it have float
ing point(1 - no, 0 - yes), and after all - chislo
```

```
clang version 7.0.0-3~ubuntu0.18.04.1 (tags/RELEASE_700/final)
> clang++-7 -pthread -o main main.cpp
> ./main
In what numeral system is your chislo? (enter number), Does it have floating point(1 - no, 0 - yes), and after all - chislo
10
1
3456,432
This is parameters constructor of Alphabet
This is parameters constructor of Phrase
This is parameters constructor of Chislo
This is copying constructor of Alphabet
This  is copiying constructor
This is copying constructor of Chislo
This is standart constructor of AlphabetThis is standart constructor of Phrase
This is standart constructor of Chislo
To ignore register enter 1, not ignore - 0. Then - enter phrase
```

```
To ignore register enter 1, not ignore - 0. Then - enter phrase
1
Hello, my dear friend!
This is parameters constructor of Alphabet
This is parameters constructor of Phrase
This is parameters constructor of Rechenna
This is copying constructor of Alphabet
This  is copiying constructor
This is copying constructor of Rechenna
This is standart constructor of AlphabetThis is standart constructor of Phrase
This is standart constructor of Rechenna

Created by parameters

Chislo
Phrase: 3456,432 using alphabet
Alphabet with 5 letters: 34562 and with 2 special symbols: ,
Basis of the calculus system 10 length of fractional part is 3
with fixed point

Rechenna
Phrase: Hello, my dear friend! using alphabet
Alphabet with 12 letters: Helomydarfin and with 2 special symbols: ,!
There are 14 symbols in alphabet
Ignore register

View
Chislo 3456,432 have basis of the calculus system 10

View
Hello, my dear friend! - have 14 symbols

Created by standart

Chislo
Phrase: 0 using alphabet
Alphabet with 7 letters: abcdml0 and with 1 special symbols:
Basis of the calculus system 10 length of fractional part is 0
with fixed point
```

```
Rechenna
Phrase: alabama0. using alphabet
Alphabet with 7 letters: abcdml0 and with 1 special symbols:
There are 6 symbols in alphabet
Not ignore register

View
Chislo 0 have basis of the calculus system 10

View
alabama0. - have 6 symbols

Created by copying of first

Chislo
Phrase: 3456,432 using alphabet
Alphabet with 5 letters: 34562 and with 2 special symbols: ,
Basis of the calculus system 10 length of fractional part is 3
with fixed point

Rechenna
Phrase: Hello, my dear friend! using alphabet
Alphabet with 12 letters: Helomydarfin and with 2 special symbols: ,!
There are 14 symbols in alphabet
Ignore register

View
Chislo 3456,432 have basis of the calculus system 10

View
Hello, my dear friend! - have 14 symbols

Deleted Rechenna
Deleted Phrase
Alphabet deleted
Deleted Rechenna
Deleted Phrase
Alphabet deleted
Deleted Rechenna
Deleted Phrase
```



Висновок:

Виконавши цю роботу я навчився правильно будувати ієрархії об’єктів з використанням успадкування та агрегації в C++.