Содержание

Практическая работа. Абстрактный тип данных (ADT) «Полином» **2**
Тема: Классы C++, библиотека STL, библиотечный класс Tlist C++ Builder. 2
Задание 2
Рекомендации к выполнению 5
Порядок выполнения 13
Содержание отчета 14
Контрольные вопросы 14

Практическая работа. Абстрактный тип данных (ADT) «Полином»

Тема: Классы C++, библиотека STL, библиотечный класс Tlist C++ Builder.

Цель: Сформировать практические навыки реализации абстрактных типов данных с помощью классов.

Задание

1. Реализовать тип «полином», в соответствии с приведенной ниже спецификацией.
2. Протестировать каждую операцию, определенную на типе данных.

Спецификация абстрактного типа данных «Полином».
ADT TPoly

Данные

Полиномы Tpoly - это неизменяемые полиномы с целыми коэффициентами.

Операции

Операции могут вызываться только объектом «полином» (тип TPoly), указатель на который передаётся в них по умолчанию. При описании операций этот объект в разделе «Вход» не указывается.

Таблица 1. Описание операций на ADT TPoly

Наименование операции	Описание
Конструктор	
Начальные значения:	Коэффициент (c) и степень (n) одночленного полинома
Процесс:	Создаёт одночленный полином с коэффициентом (c) и степенью (n), или ноль-полином, если коэффициент (c) равен 0 и возвращает указатель на него. Например: Конструктор(6,3) = 6x^3 Конструктор(3,0) = 3 Конструктор() = 0
Выход:	Указатель на созданный полином.
Постусловия:	Нет.
Степень	
Вход:	Нет.
Предусловия:	Нет.

Процесс:	Отыскивает степень n полинома, т.е. наибольшую степень при ненулевом коэффициенте (c). Степень нулевого полинома равна 0. Например: a = (x^2+1), a.Степень = 2 a = (17), a. Степень = 0
Выход:	n - целое число - степень полинома.
Постусловия:	Нет.
Коэффициент	
Вход:	n - целое число - степень полинома.
Предусловия:	Полином – не нулевой.
Процесс:	Отыскивает коэффициент (c) при члене полинома со степенью n (c*x^n). Возвращает коэффициент (c) найденного члена или 0, если n больше степени полинома. Например: p = (x^3+2x+1), p.Coeff (4) = 0 p = (x^3+2x+1), p.Coeff (1) = 2
Выход:	Целое число.
Постусловия:	Нет.
Очистить (Clear)	
Вход:	q - полином.
Предусловия:	Нет
Процесс:	Удаляет члены полинома.
Выход:	Полином.
Постусловия:	q – нуль-полином.
Сложить	
Вход:	q - полином.
Предусловия:	Нет
Процесс:	Создаёт полином, являющийся результатом сложения полинома с полиномом q и возвращает его.
Выход:	Полином.
Постусловия:	Нет.
Умножить	

Вход:	q - полином.
Предусловия:	Нет.
Процесс:	Создаёт полином, являющийся результатом умножения полинома на полином q и возвращает его.
Выход:	Полином.
Постусловия:	Нет.
Вычесть	
Вход:	q - полином.
Предусловия:	Нет.
Процесс:	Создаёт полином, являющийся результатом вычитания из полинома полинома q, и возвращает его.
Выход:	Полином.
Постусловия:	Нет.
Минус	
Вход:	Нет.
Предусловия:	Нет.
Процесс:	Создаёт полином, являющийся разностью ноль-полинома, и полинома и возвращает его.
Выход:	Полином.
Постусловия:	Нет.
Равно	
Вход:	q - полином.
Предусловия:	Нет.
Процесс:	Сравнивает полином с полиномом q на равенство. Возвращает значение True, если полиномы равны, т.е. имеют одинаковые коэффициенты при соответствующих членах, и значение False - в противном случае.
Выход:	Булевское значение.
Постусловия:	Нет.
Дифференцировать	
Вход:	Нет.
Предусловия:	Нет.

Процесс:	Создаёт полином, являющийся производной полинома и возвращает его. Например: a = (x^3+7x+5), a.Дифференцировать = $3x^2$+7
Выход:	Полином.
Постусловия:	Нет.
Вычислить	
Вход:	x – действительное число.
Предусловия:	Нет.
Процесс:	Вычисляет значение полинома в точке x и возвращает его. Например: a = (x^2+3x), a.Вычислить(2) = 10
Выход:	Действительное число.
Постусловия:	Нет.
Элемент	
Вход:	i - целое число - номер члена полинома.
Предусловия:	Нет.
Процесс:	Обеспечивает доступ к члену полинома с индексом i для чтения его коэффициента (c) и степени (n) так, что если изменять i от 0 до количества членов в полиноме минус один, то можно просмотреть все члены полинома.
Выход:	Коэффициент – целое число, степень – целое число.
Постусловия:	Полином не модифицируется.

end Tpoly

Рекомендации к выполнению

1. Тип данных реализовать, используя классы C++ и библиотеку STL.
2. Полином можно рассматривать как список одночленных полиномов, поэтому для реализации полинома полезно реализовать абстрактный вспомогательный тип данных одночленный полином. Спецификация для него приведена ниже.

Спецификация абстрактного типа данных Одночлен.

ADT TMember

Данные

Одночлен TMember - это изменяемые одночленные полиномы с целыми коэффициентами. Коэффициент и степень хранятся в полях целого типа FCoeff и FDegree соответственно.

Операции

Операции могут вызываться только объектом «одночлен» (тип TMember), указатель на который передаётся в них по умолчанию. При описании операций этот объект в разделе «Вход» не указывается.

Таблица 2. Описание операций на **ADT TMember**.

Наименование операции	Описание
Конструктор	
Начальные значения:	Коэффициент (c) и степень (n) одночленного полинома
Процесс:	Создаёт одночленный полином с коэффициентом (c) и степенью (n), или ноль-полином, если коэффициент (c) равен 0 и возвращает указатель на него. Например: Конструктор(6,3 = 6x^3 Конструктор(3,0) = 3 Конструктор() = 0
Выход:	Указатель на созданный одночленный полином.
Постусловия:	Нет.
ЧитатьСтепень	
Вход:	Нет.
Предусловия:	Нет.
Процесс:	Возвращает степень n одночленного полинома (содержимое поля FDegree). Степень нулевого полинома равна 0. Например: a = (1x^2), a.Степень = 2
Выход:	n - целое число - степень полинома.
Постусловия:	Нет.
ПисатьСтепень	

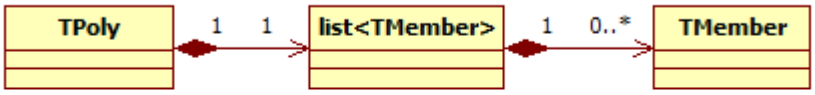
Вход:	n - целое число - степень полинома.
Предусловия:	Нет.
Процесс:	Записывает степень n одночленного полинома в поле FDegree.
Выход:	Нет.
Постусловия:	Поле FDegree = n.
ПисатьКоэффициент	
Вход:	c - целое число - коэффициент полинома.
Предусловия:	Нет.
Процесс:	Записывает коэффициент c одночленного полинома в поле FCoeff.
Выход:	Нет.
Постусловия:	Поле FCoeff = c.
Равно	
Вход:	q - одночлен.
Предусловия:	Нет.
Процесс:	Сравнивает одночлен с одночленом q на равенство. Возвращает значение True, если одночлены равны, т.е. имеют одинаковые коэффициенты и степени, и значение False - в противном случае.
Выход:	Булевское значение.
Постусловия:	Нет.
Дифференцировать	
Вход:	Нет.
Предусловия:	Нет.
Процесс:	Создаёт одночлен, являющийся производной одночлена и возвращает его. Например: a = (x^3), a.Дифференцировать = $3x^2$.
Выход:	Одночлен.
Постусловия:	Нет.

Вычислить	
Вход:	x – действительное число.
Предусловия:	Нет.
Процесс:	Вычисляет значение одночлена в точке x и возвращает его. Например: $a = (1x^2)$, a.Вычислить(2) = 4.
Выход:	Действительное число.
Постусловия:	Нет.
ОдночленВСтроку	
Вход:	Нет.
Предусловия:	Нет.
Процесс:	Формирует строковое представление одночлена.
Выход:	Строка.
Постусловия:	Нет.

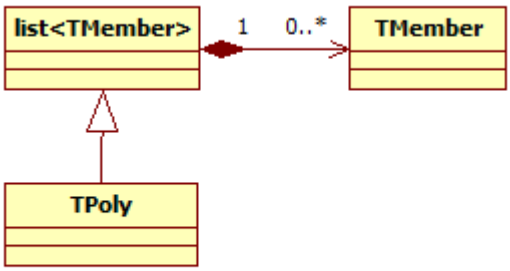
end TMember

3. Члены полинома храните в контейнере STL. Проанализируйте операции на полиноме и выберите тип контейнера.
4. Для реализации одночленного полинома (одночлена) создайте класс Tmember, в который вынесите все операции на членах полинома. После выполнения каждой операции приводите полином к нормализованному виду: упорядочить, привести подобные, удалить нулевые члены (члены с нулевыми коэффициентами).
5. Тип данных реализуйте в отдельном модуле UPoly.

Ниже приведены два примера классов, реализующих абстракцию данных полином.



Вложение



Наследование

Пример реализация полинома с помощью классов C++ Builder.

Для реализации полинома на C++ Builder можно использовать его библиотечный класс TList, который представляет собой список указателей на объекты класса TObject. Класс TObject является предком всех классов в модели ООП реализованной в C++ Builder.

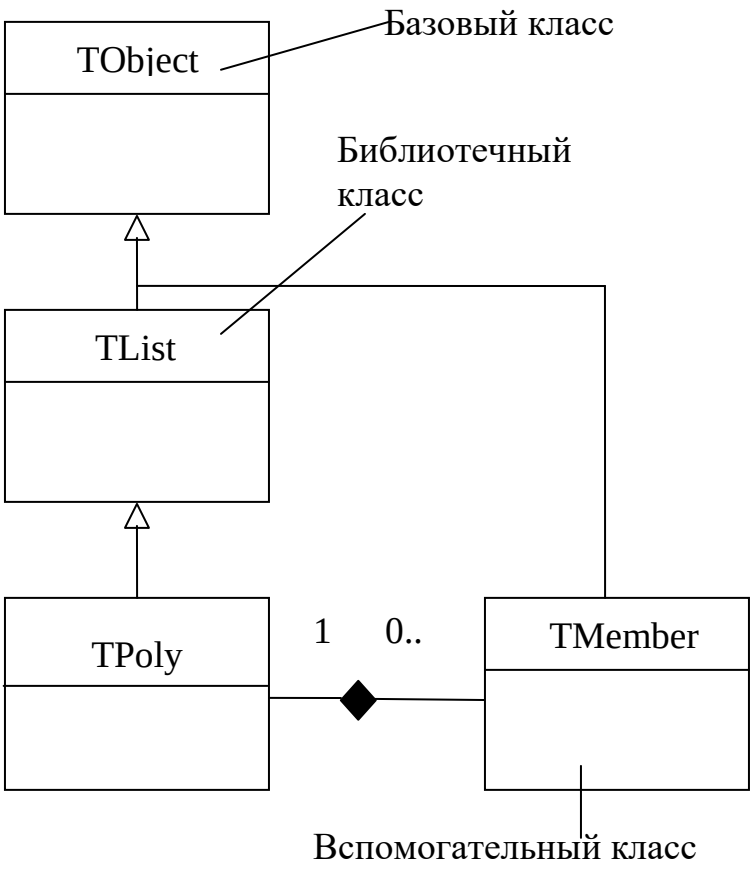


Рисунок 1. Диаграмма UML – описание реализации полинома.

Реализация полинома будет представлена модулем, состоящим из двух файлов: заголовочного файла UPoly.h с описанием классов TPoly, TMember и файла UPoly.cpp, содержащего описание методов классов. Заголовочный файл UPoly.h с описанием классов TPoly, TMember и файл головной программы для тестирования представлены ниже.

```
//---файл UPoly.h-------------------------------
#ifndef UPolyH
#define UPolyH
#include <vcl.h>
#include <math.h>
//----------------------------------------------
#endif
```

```
//---------------------------------------------
class TMember : public TObject
{
    int c,d;//Коэффициент и степень одночлена
    String Get(void) const;
  public:
    TMember(int nc = 0, int nd = 0):
c(nc),d(nd){};//Конструктор
    TMember(const TMember& P);//Конструктор копирования
    bool       operator == (const TMember& T);//Сравнение
//двух объектов
    void       operator - (void){ Coeff = -
Coeff;};//Смена знака //одночлена на обратный
    TMember*   Diff(void);//Дифференцирование одночлена
    double     Eval(double r){return
c*pow(r,d);};//Вычисление //одночлена
   __property String Member = {read = Get};//Читает объект
//в формате строки
   __property int Degree  = {read = d, write = d};//Читать
//и писать степень
   __property int Coeff = {read = c, write = c};//Читать и
//писать коэффициент
};

//---------------------------------------------
class TPoly : public TList
{
    String    Get(void) const;
    TMember*  GetMember(int i) const;
    void      Likeness(void);//Приведение подобных
    void      RemoveZero(void);//Удаление нулевых
    void      Invar(void);//Приведение полинома к
//нормализованному виду
    TPoly*    MulNum(TMember* Num);
  public:
    TPoly(int nc = 0, int nd = 0);//Конструктор
    TPoly(const TPoly& P);//Копирующий конструктор
    virtual __fastcall ~TPoly(void);//Деструктор
    int     Degree(void) const;//Степень полинома
    double  Eval(double x);//Вычисление значения полинома
    void    operator -(void);//Изменение знака полинома
    TPoly*  Diff(void);//Дифференциирование полинома
```

```
    TPoly*  operator + (const TPoly& T);//Сложение
//полиномов
    TPoly*  operator - (const TPoly& T);//Вычитание
//полиномов
    TPoly*  operator * (const TPoly& B);//Умножение
//полиномов
    TMember*  operator [] (int i);//Одночлен по индексу
    bool    operator == (const TPoly& T);//Сравнение двух
//полиномов
  __property String Poly = {read = Get};//Чтение полинома
//в формате строки
  __property TMember* Member[int index] = {read =
GetMember};
};

//-------файл UMain.cpp--------------------------
#include <vcl.h>
#pragma hdrstop
#include "UPoly.h"
#include <iostream.h>
#include <stdarg.h>
//-----------------------------------------------
#pragma argsused
int main(int argc, char* argv[])
{
  int i = 0;
  TMember* M;
  TPoly* P0 = new TPoly();
  TPoly* P1 = new TPoly(1,0);
  TPoly* P2 = new TPoly(1,1);
  TPoly* P4 = new TPoly(2,2);
  TPoly* P3 = *P1 + *P2;//1X^0 + 1X^1
  cout << "1X^0 + 1X^1" <<P3->Poly.c_str()<<endl;//1X^0 +
1X^1
  TPoly* P5 = (*P1) + (*P2);//1X^0 + 1X^1
  cout << "1X^0 + 1X^1" << P5->Poly.c_str()<<endl;
  -*P5;
  cout << "- 1X^0 - 1X^1" << P5->Poly.c_str()<<endl;
  TPoly* P6 = (*P5) * (*P2);
  cout << "- 1X^0 - 1X^2" << P6->Poly.c_str()<<endl;
  -*P5;
  P6 = (*P5) * (*P3);
```

```
  cout << "1X^0 + 2X^1 + 1X^2" << P6->Poly.c_str()<<endl;
  TPoly* P7 = (*P5).Diff();
  cout << "0X^0 + 1X^0" << P7->Poly.c_str()<<endl;
  double r = (*P5).Eval(1);
  cout << "P8.Eval(1) = 2 " << r <<endl;
  //M = static_cast<TMember*>(P4->Items[i]);
  //TPoly* P5 = P3->MulNum(M );
  //M = static_cast<TMember*>(Poly->Items[0]);
  //cout <<M->Member.c_str()<<endl;
  //Poly->_Add(new TMember(6,6));
  //M = static_cast<TMember*>(Poly->Items[1]);
  cin >> i;
  return 0;
}
//---------------------------------------------------------
```

Порядок выполнения

В режиме консольного приложения

- опишите класс TMember, реализуйте класс и оттестируйте каждый метод;
- опишите класс TPoly, реализуйте класс и оттестируйте каждый метод.

Тестовые наборы поместите в таблицу 3 следующего вида:

Таблица 3. Тестовый набор для тестирования операции «Умножить» на типе «Полином»

Тестовый набор для тестирования операции «Умножить»				
Номер теста	Исходные данные		Ожидаемый результат	
	Вход	Полином	Возвращаемое значение	Полином
1	0*X^0	0*X^0	0*X^0	0*X^0
2	0*X^0	1*X^0	0*X^0	1*X^0
3	1*X^0	1*X^0	1*X^0	1*X^0
4	1*X^0	2*X^1	2*X^1	2*X^1
5	1*X^0	2*X^1+3*X^2	2*X^1+3*X^2	2*X^1+3*X^2
6	1*X^0+1*X^1	1*X^0-1*X^1	1*X^0-1*X^2	1*X^0-1*X^1

Содержание отчета

1. Задание.
2. Текст программы на Object Pascal, C++ Builder.
3. Тестовые наборы данных для тестирования типа данных.

Контрольные вопросы

1. Чем определяется размер памяти, выделяемой под экземпляр класса?
2. Что такое RTTI класса?
3. Как и когда происходит связывание объекта с RTTI класса?
4. Как описываются и переопределяются виртуальные и динамические методы?
5. Что такое раннее связывание, и для каких методов оно выполняется?
6. Что такое позднее связывание, и для каких методов оно выполняется?
7. Когда для класса необходимо описать собственный деструктор?