**Лабораторный практикум**

**Языковые процессоры интеллектуальных систем**

* Прием лабораторных и практических производится только на занятии своей группы или подгруппы.
* Сроки сдачи лабораторных работ указаны в файле с учетом успеваемости, после дедлайна оценка снижается.
* Для рекомендации на автомат нужно сдать все работы в срок и иметь посещаемость 80%+ на лекциях и лабораторных.
* Проверка посещаемости на каждой паре
* При плохой дисциплине будут добавляться вопросы или задания к защите работ
* Варианты определяются по номеру в таблице

**Лабораторная работа 1**

Написать 3 примера фрагментов кода языка по варианту, все свойства должны быть использованы, встроенные типы, операции и функции должны быть использованы и показаны на примерах.

**Лабораторная работа 2**

Составить файл граматики с использованием ANTLR, можно отлаживать здесь(<http://lab.antlr.org/>), проверить на примерах из задания 1.1. Показывать выполнение задания в среде разработки.

**Лабораторная работа 3**

Написать синтаксический анализатор, который выдает ошибки, тестируется на примерах из задания 1.1, протестировать на новых примерах, показывающих обработку ошибок.

**Лабораторная работа 4**

Написать семантический анализатор, который выдает ошибки, тестируется на примерах из задания 1.1, протестировать на новых примерах, показывающих обработку ошибок

**Лабораторная работа 5**

Написать компилятор в исходный код по варианту и написать отчет

Содержание отчета лабораторного практикума следующее:

1. Спецификация разработанного языка программирования.
   1. Синтаксис объявления переменных и подпрограмм
   2. Синтаксис операций над данными (их должно быть не менее 10 штук)
   3. Синтаксис всех управляющих конструкций
2. Оформленный (отформатированный и прокомментированный) файл грамматики (.g).
3. Описание дополнительно разработанных классов.
4. Перечень генерируемых ошибок.
5. Примеры работы компилятора.

***1.1 Требования к разрабатываемому языку***

1. Встроенные типы не менее трех
2. Возможность инициализация переменных всех типов при объявлении: <тип>? <имя\_переменной> = <выражение>
   1. Инициализирующее выражение может быть константным
3. Встроенные операции не менее 10 штук
4. Встроенные функции
   1. Встроенные функции ввода\вывода для работы со встроенными типами read() write()
5. Использование сложных выражений (составных и со скобками)
6. Блочный оператор
7. Управляющие структуры
   1. Условный оператор (if-then-else)
   2. Операторы цикла (while и until)
   3. Оператор цикла с итерациями (for)
8. Пользовательские подпрограммы
   1. Передача и возврат параметров
   2. Задание локальной и глобальной области видимости для имен переменных

***1.2 Варианты свойств языка***

1. Объявление переменных
   1. Явное
   2. Не явное
2. Преобразование типов
   1. Явное, например, a = (int) b
   2. Не явное
3. Оператор присваивания
   1. Одноцелевой, например, a = b
   2. Многоцелевой, например, a, b = c, d
4. Структуры, ограничивающие область видимости
   1. Подпрограммы
   2. Подпрограммы и блочные операторы
5. Маркер блочного оператора
   1. Явные, например, { } или begin end
   2. Не явный, например как в python
6. Условные операторы
   1. Двух вариантный оператор if-then-else
   2. Двух вариантный оператор и многовариантный switch-case
7. Перегрузка подпрограмм
   1. Отсутствует
   2. Присутствует
8. Передача параметров в подпрограмму
   1. Только по значению и возвращаемому значению
   2. По значению и результату
   3. По ссылке
9. Допустимое место объявления подпрограмм:
   1. В начале программы
   2. В любом месте программы, также и внутри другой подпрограммы.

***1.3 Варианты языков***

1. Язык, описывающий математические вычисления
   1. Встроенные типы: int, float
   2. Операции: +, -, \*, \, %, ^, ==, !=, <, >, <=, >=
   3. Встроенные функции log, ln, sin, cos, tan, asin, acos, atan
2. Язык для работы с векторами и матрицами
   1. Встроенные типы: vector, matrix
   2. Операции: v + v, v - v, n \* v, v \* v, |v|, m + m, m - m, m \* n, m \* m, m[n], m[n] \* n, |m|, v \* m
3. Язык для работы с графовыми структурами
   1. Встроенные типы: node, arc, graph
   2. Операции: переопределить +, -, \*, \ и т.д. для встроенных типов
   3. Встроенные функции поиска в ширину, поиска в глубину, поиска кратчайшего пути
4. Язык для работы с множествами
   1. Встроенные типы: element, set
   2. Операции: переопределить +, -, \*, \ и т.д. для встроенных типов
5. Язык для работы со строками
   1. Встроенные типы: char, string, массив string
   2. Операции: переопределить +, -, \*, \ , [] и т.д. для встроенных типов
   3. Встроенные функции поиска подстроки, извлечения среза, разбиение, замена по шаблону
6. Язык для работы со списковыми структурами
   1. Встроенные типы: element, list
   2. Операции: переопределить +, -, \*, \ и т.д. для встроенных типов
   3. Списковые структуры: дерево, очередь
   4. Встроенные функции балансировки, слияния
7. Язык для работы с xml данными
   1. Встроенные типы: document, node, attribute
   2. Встроенные функции поиска и преобразования
8. Язык для работы с реляционными данными
   1. Встроенные типы: table, row, column
   2. Встроенные функции задания структур, изменения данных, поиска

***1.4 Варианты целевого кода***

1. Байт-код JVM, формат промежуточного кода ассемблер для JVM
2. Байт-код .NET, формат промежуточного кода ассемблер для .NET (CIL)
3. Исполняемый файл (.exe), для генерации целевого кода использовать LLVM (http://llvm.org)
4. Исполняемый файл (.exe), для генерации целевого код использовать GCC Front end (входит в состав GNU GCC)

***Варианты заданий***

| Вариант | Язык(1.3) | Свойства(1.2) | | | | | | | | | Целевой код(1.4) |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 1 | 2 |
| 3 | 3 | 1 | 2 | 1 | 1 | 1 | 2 | 1 | 3 | 2 | 3 |
| 4 | 4 | 2 | 2 | 1 | 1 | 2 | 2 | 1 | 1 | 1 | 4 |
| 5 | 2 | 1 | 1 | 2 | 1 | 1 | 1 | 2 | 1 | 1 | 1 |
| 6 | 3 | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 3 | 1 | 2 |
| 7 | 4 | 1 | 2 | 2 | 1 | 1 | 2 | 2 | 1 | 1 | 3 |
| 8 | 5 | 2 | 2 | 2 | 1 | 2 | 2 | 2 | 1 | 2 | 4 |
| 9 | 3 | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 3 | 1 | 1 |
| 10 | 4 | 2 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 1 | 2 |
| 11 | 5 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 3 |
| 12 | 6 | 2 | 2 | 1 | 2 | 2 | 2 | 1 | 3 | 2 | 4 |
| 13 | 4 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 2 | 1 | 1 |
| 14 | 5 | 2 | 1 | 2 | 2 | 2 | 1 | 2 | 2 | 1 | 2 |
| 15 | 6 | 1 | 2 | 2 | 2 | 1 | 2 | 2 | 3 | 2 | 3 |
| 16 | 7 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 4 |
| 17 | 5 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 18 | 6 | 2 | 1 | 1 | 1 | 2 | 1 | 1 | 3 | 1 | 2 |
| 19 | 7 | 1 | 2 | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 3 |
| 20 | 8 | 2 | 2 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 4 |
| 21 | 6 | 1 | 1 | 2 | 1 | 1 | 1 | 2 | 3 | 1 | 1 |
| 22 | 7 | 2 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 1 | 2 |
| 23 | 8 | 1 | 2 | 2 | 1 | 1 | 2 | 2 | 1 | 1 | 3 |
| 24 | 1 | 2 | 2 | 2 | 1 | 2 | 2 | 2 | 3 | 2 | 4 |
| 25 | 7 | 1 | 1 | 1 | 2 | 1 | 1 | 1 | 2 | 1 | 1 |
| 26 | 8 | 2 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 1 | 2 |
| 27 | 1 | 1 | 2 | 1 | 2 | 1 | 2 | 1 | 3 | 1 | 3 |
| 28 | 2 | 2 | 2 | 1 | 2 | 2 | 2 | 1 | 2 | 2 | 4 |
| 29 | 8 | 1 | 1 | 2 | 2 | 1 | 1 | 2 | 2 | 1 | 1 |
| 30 | 1 | 2 | 1 | 2 | 2 | 2 | 1 | 2 | 3 | 1 | 2 |