**Варианты заданий**

**на лабораторные работы и РГР**

 В качестве задания на выполнение лабораторных работ и РГР является описание синтаксиса и семантических соглашений некоторого учебного языка программирования.

 Синтаксис учебного языка разрабатывается студентом самостоятельно на основе некоторого языка-прототипа.

 Общие требования к учебному языку:

* язык должен быть со строгой явной статической типизацией, т.е. типы всех объектов должны быть объявлены в специальном разделе описаний и не допускается неявное преобразование типов;
* должны быть ключевые слова, обозначающие начало и конец программы;
* программа должна состоять из раздела описаний и раздела (последовательности) операторов;
* должно быть, как минимум, три простых предопределенных базовых типа (целый, вещественный, логический);
* для арифметических выражений должны быть определены, как минимум, операции сложения, вычитания, умножения, деления (унарные плюс и минус – по желанию);
* для логических выражений должны быть определены операции дизъюнкции (ИЛИ), конъюнкции (И), отрицания (НЕ) и шесть операций отношения;
* обязательным оператором является оператор присваивания;
* текст программы должен допускать использование комментариев.

 Приоритетность арифметических и логических операций определяется языком-прототипом.

 Если в языке-прототипе отсутствуют какие-либо пункты перечисленных выше общих требований, следует разработать их самостоятельно.

 К общим требованиям добавляются производные типы и другие операторы в зависимости от номера варианта.

 Номер варианта состоит из двух цифр. Первая цифра означает выбор оператора, вторая цифра – выбор языка-прототипа.

 **Операторы:**

 0) цикл с предусловием типа **while**

 1) цикл с постусловием типа **repeat**

 2) цикл с параметром типа **for**

 3) условный оператор типа **if**

 4) оператор варианта типа **case**

 5) процедуры (описание и вызов)

 6) функции (описание и вызов)

 **Языки-прототипы:**

 0) Ada 5) Fortran

 1) Simula 6) Ruby

 2) C, C++, C#, Java 7) Python

 3) Delphi, Pascal 8) Oberon

 4) Swift 9) Scala

 **Производные типы данных**: для четных номеров – массив, для нечетных – запись. **Для заданий с процедурами и функциями нет производных типов.**

 Студент должен описать синтаксис языка в нормальной или расширенной форме Бэкуса-Наура (БНФ или РБНФ). Поскольку БНФ не позволяет задавать контекстные условия, раскрывающие особенности семантики языка, их можно записать в словесной форме в виде перечня неформальных семантических соглашений.

 Существуют различные модификации синтаксиса РБНФ. Можно рекомендовать следующий вариант РБНФ.

 Металингвистическая переменная (нетерминал) обозначается произвольной символьной строкой (без использования угловых скобок как в БНФ). Если нетерминал состоит из нескольких смысловых слов, то они записываются слитно или разделяются символом подчеркивания. При этом для удобства восприятия целесообразно каждое ее слово начинать с прописной буквы.

 Терминальные символы изображаются словами, написанными буквами латинского алфавита (ключевые слова) или цепочками символов, заключенными в одиночные (′) или двойные (″) кавычки. Для удобства восприятия ключевые слова дополнительно можно выделить жирным шрифтом.

 Левая и правая части правила разделяются метасимволом "=" (вместо "::=" в БНФ), альтернативные варианты разделяются метасимволом "**|**". Каждое правило заканчивается точкой.

 Квадратные скобки "[" и "]" означают, что заключенная в них синтаксическая конструкция может отсутствовать. Фигурные скобки "{" и "}" означают нуль или более повторений заключенной в них синтаксической конструкции.

 Ниже приведен пример описания синтаксиса учебного языка с использованием РБНФ.

1. Модуль = "**module**" Идент "**;**" Блок "**.**".

2. Идент = Буква {Буква **|** Цифpа }.

3. Блок = { Описание } "**begin**" ПоследОператоров "**end**".

4. Описание = "**type**" { ОписаниеТипа "**;**" } **|**

 "**var**" { ОписаниеПерем "**;**" }.

5. ОписаниеТипа = Идент "**=**" Тип.

6. ОписаниеПерем = СписокИдент "**:**" Тип.

7. СписокИдент = Идент { "**,**" Идент }.

8. Тип = ПростойТип **|** ТипМассив.

9. ПростойТип = Идент **|** Диапазон.

10. Диапазон = Константа "**..**" Константа.

11. ТипМассив = "**array**" "**[**" ПростойТип { "**,**" ПростойТип } "**]**"

 "**of**" Тип.

12. ПоследОператоров = Оператор { "**;**" Оператор }.

13. Оператор = Присваивание **|** Цикл.

14. Присваивание = Переменная "**:=**" Выражение.

15. Переменная = Идент **|** ИндексПеpеменная.

16. ИндексПеpеменная = Идент"**[**"ПростоеВыраж

 { "**,**" ПростоеВыраж } "**]**".

17. Цикл = "**while**" Выражение "**do**" ПоследОператоров "**end**".

18. Выражение = ПростоеВыраж [ Отношение ПростоеВыраж ].

19. Отношение = "**<**" **|** "**<=**" **|** "**>**" **|** "**>=**" **|** "**=**" **|** "**#**".

20. ПростоеВыраж = ["**+**" **|** "**–**" ] Терм { АддитОперация Терм }.

21. АддитОперация = "**+**" **|** "**–**" **|** "**or**".

22. Терм = Фактор { МультОперация Фактор }.

23. МультОперация = "**\***" **|** "**/**" **|** "**and**" **|** "**div**".

24. Фактор = Константа **|** Пеpеменная **|** "**(**" Выражение "**)**" **|**

 "**not**" Фактор.

25. Константа = Целое [ "**.**" Целое ].

26. Целое = Цифpа { Цифpа }.

27. Цифра = "0"**|**"1"**|**"2"**|**"3"**|**"4"**|**"5"**|**"6"**|**"7"**|**"8"**|**"9".

 Определение нетерминала «Буква» здесь не приведено ввиду его очевидности – определяется выбранным алфавитом (обычно строчные и прописные буквы латинского алфавита).

Краткая характеристика языка и семантические соглашения:

* Язык удовлетворяет семантическим соглашениям, характерным для многих языков программирования (единственность именования различных объектов программы, необходимость описания идентификатора до его использования и т.п.).
* В ключевых словах и идентификаторах прописные и строчные буквы не различаются.
* Ключевые слова языка зарезервированы, их нельзя использовать в качестве идентификаторов.
* Отсутствуют именованные константы.
* Предопределенные типы: integer, float, Boolean, char.
* В конструкции «Диапазон» (правило 10) константы должны быть целого типа, кроме того, значение первой константы должно быть меньше значения второй константы.
* Если в конструкции «ТипМассив» (правило 11) в качестве «ПростойТип» используется идентификатор («Идент»), он должен представлять тип «Диапазон».
* В операторе цикла конструкция «Выражение» после ключевого слова while должно быть типа Boolean (правило 17).
* Комментарий представляет собой любую последовательность символов, заключенную в фигурные скобки "{" и "}".