## 1. Задания для лексического анализа

Во всех задачах приняты следующие обозначения:

- 1) слова, выделенные жирным шрифтом (например, **модуль**) являются ключевыми словами рассматриваемых языков;
- 2) всё, заключённое в квадратные скобки зелёного цвета ([...]) является необязательным;
- 3) то, что заключено в фигурные скобки зелёного цвета  $(\{...\})$ , может повторяться любое число раз, в том числе и ни разу;
- 4) круглые скобки зелёного цвета ((...)) служат для группировки конструкций;
- 5) знак | зелёного цвета означает "или";
- 6) запись ид обозначает идентификатор;

 $6bip_2 \rightarrow \{!\}6bip_3$ 

7) всё прочее, что имеет чёрный цвет, является либо знаком операции, либо разделителем языка из задачи.

```
Задача 1.1. С помощью генератора лексических анализаторов Мяука для приводимого ниже
языка напишите и протестируйте лексический анализатор.
программа \to модуль ид {[использует списид] [предоставляет списид] тело модуля}
cnucud \rightarrow uд{,uд}
тело модуля \rightarrow {переменные | munы | функция | прототип | константы }
nepeменные \rightarrowперем cnucud:npocmeŭuee onpedeление <math>muna\{;cnucud:npocmeŭuee onpedeлeниe <math>muna\}
munu \rightarrow \mathbf{TИП} \ \mathbf{ИД} = onpederenue \ muna\{; \mathbf{ИД} = onpederenue \ muna\}
onpedeление muna 	o npocme
uuee onpedеление muna|cocmaвное onpedеление muna|cocmasное onpedеление onpedелени
простейшее определение типа \rightarrow элементарный тип\mid
                                                               массив[[выр]{,[выр]}] простейшее определение типа
элементарный mun \rightarrow \{@\} (встроенный mun \mid \mu \mu)
6cmpoeнный mun \rightarrow \{большое|маленькое\} (цел|вещ|беззн|лог|симв|строка)|ничто|байт|цел8|
                                     цел16|цел32|цел64|цел128|беззн8|беззн16|беззн32|беззн64|беззн128|
                                     вещ32|вещ64|вещ128|лог8|лог16|лог32|лог64|симв8|симв16|симв32|
                                     строка8/строка16/строка32
cocmaвное определение muna \rightarrow onpedenenue kopme \rightarrow color pedenenue указателя на функцию
                                                           определение алгебраического muna
определение кортежа \rightarrow (:элементарный mun\{,элементарный mun\}:)
определение указателя на функцию 
ightarrow \mathbf{\phi} \mathbf{y} \mathbf{n} \mathbf{k} \mathbf{q} \mathbf{u} \mathbf{s} сигнатура функции
сигнатура функции \rightarrow (спис форм парам):элементарный тип
cnuc форм napam \rightarrow \phiорм napam \{, \phiорм napam \}
форм парам \rightarrowзначение списид:простейшее определение типа
                            ссылка списид:простейшее определение типа
                            конст ссылка списид:простейшее определение типа
определение алгебраического типа \rightarrow часть алгебр типа\{...часть алгебр типа\}
часть алгебр типа \rightarrowперечисление|структура
neperuc_nehue \rightarrow neperuc_nehue ид\{cnucud\}
cmpyкmypa \rightarrow crpyкrypa ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud:простейшее определение типа
\phiункция \to \phiункция ид сигнатура \phiункции\{тело блока\}
npomomun \rightarrow \phiункция ид сигнатура функции;
константы \toконст определение константы \{, определение константы \}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow{переменные|типы|функция|прототип|константы|операторы}}
ewp \rightarrow ewp_0 [?ewp_0:ewp_0]
\varepsilon \omega p_0 \rightarrow \varepsilon \omega p_1 \{ (|||!|||^{^{}}|!^{^{}}) \varepsilon \omega p_1 \}
sup_1 \rightarrow sup_2 \{ (\&\&/!\&\&) sup_2 \}
```

```
6bip_3 \rightarrow 6bip_4 \{ (< > < = > = | = | ! = ) 6bip_4 \}
6bip_4 \rightarrow 6bip_5 \{(||^{\sim}||^{\wedge}|^{\sim})6bip_5\}
eup_5 \rightarrow eup_6 \{ (\&|^{\sim}\&| \ll |\gg|) \otimes up_6 \}
eup_6 \rightarrow {^{\sim}}eup_7
eup_7 \rightarrow eup_8\{(+|-)eup_8\}
\varepsilon \omega p_8 \rightarrow \varepsilon \omega p_9 \{ (*//\%/.) \varepsilon \omega p_9 \}
eup_9 \rightarrow eup_{10}[(**|**.)eup_9]
выp_{10} \to [+/-/\#] выp_{11}
выp_{11} \rightarrow выp_{12} [##выp_{12}]
6 bip_{12} \to \text{целое} | \text{вещественное} | \textbf{истина} | \textbf{ложь} | \text{строковое} | (6 bip) | uma | cocm знач
cocm знач \rightarrowзнач cmpykm/знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nons\{, started ones \}\}
значение nоля \rightarrowид <- \epsilonыр
знач массива \to массив[:выр,элементарный тип:]{списвыр}
uмя \rightarrow [[:uмя модуля::][uмя типа::uмя части типа::]ид{([списвыр])|[списвыр]]@[.ид}
cnuceup \rightarrow eup\{,eup\}
uмя модуля \rightarrowид
uмя muna \rightarrow uд
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop 
ightarrow ukra | npucваивания | nepexoda | ввода | вывода | условный | разбора
условный \toесли выр \mathbf{ro}\{тело блока\}\{инес выр \mathbf{ro}\{тело блока\}\}\{иначе\{тело блока\}\}всё
uu\kappa na \rightarrow [<:uд:>] (для ид из uu выр\{meno\ блока\}|для ид := выр[,выр]\{meno\ блока\}|
           пока выр{тело блока}|повторяй{тело блока}покуда выр)
присваивания \to ums := выр
nepexoda \to \mathbf{выйди} [из ид] |\mathbf{продолжи} [ид] |\mathbf{возврат}| [выр]
ввода →ввод(списимён)
выводa \rightarrowвывод(cnuceup)
cnucum\ddot{e}н \rightarrowumя\{,umя\}
разбора \rightarrowразбор выр\{образец ->\{ тело блока\}\{,образец ->\{ тело блока\}\}\{ [ иначе\{ тело блока\}\}\}
образец \rightarrow []:имя модуля::]имя типа::имя части типа\{...\}
   Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
uenoe \rightarrow decsmuuhoe|восьмеричное|двоичное|шестнадиатеричное
\partial ecsmuчнoe \rightarrow \partial ec цифра('?\partial ec цифра)*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
\partial воичное \rightarrow 0(b|B)\partial воичн цифра('?\partial воичн цифра)*
uecmhaduamepuuhoe \rightarrow 0(x|X)uecmh uuppa('?uecmh uuppa)*
\partial ec \quad uu\phi pa \rightarrow 0/1/2/3/4/5/6/7/8/9
восьм иифра \rightarrow 0/1/2/3/4/5/6/7
\partialвоичн цифра \rightarrow 0/1
uecm + uu \phi pa \rightarrow 0 |1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F
вещественное \rightarrowцелая часть (дробная часть)?(символ порядка(+|-)?порядок)?
символ порядка \rightarrow E|e|S|s|D|d|Q|q
целая часть \rightarrow \partial e cятичное
дробная часть \rightarrow. десятичное
nopядoк \rightarrow decяmичноe
```

Если перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано и32. Если указано и16, то строковый литерал — в кодировке UTF-16. Наконец, если указано и8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

Задача 1.2. С помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор. программа  $\rightarrow$  module ид {[import cnucud] [export cnucud] тело модуля}  $cnucud \rightarrow uд{,uд}$ тело модуля  $\rightarrow$  {переменные | munы | функция | прототип | константы } nepeменные  $\rightarrow$  var cnucud:npocmeйшее onpedeneние muna  $\{;cnucud:$ npocmeйшее onpedeneние muna  $\}$  $munu \rightarrow type \, ud = onpedenehue \, muna \{; ud = onpedenehue \, muna\}$ onpederehue muna 
ightarrow npocmeŭwee onpederehue muna | cocmaвное onpederehue munaпростейшее определение типа  $\rightarrow$ элементарный тип $\mid$ array[[выр]{,[выр]}] простейшее определение типа элементарный  $mun \rightarrow \{@\}$  (встроенный  $mun \mid \mu \mu$ ) 6cmpoeнный  $mun \rightarrow \{long|short\} (int|float|uint|bool|char|string)|void|byte|int8 <math>t|int16$  t|int32 t|int64 t|int128 t|uint8 t|uint16 t|uint32 t|uint64 t|uint128 t|float32 t|float64 t|float128 t|bool8 t|bool16 t|bool32 t| $bool 64 \quad t | char 8 \quad t | char 16 \quad t | char 32 \quad t | string 8 \quad t | string 16 \quad t | string 32 \quad t$ cocmaвное определение  $muna \rightarrow onpederenue$   $kopme \rightarrow onpederenue$  kopmeопределение алгебраического типа определение кортежа  $\rightarrow$  (:элементарный  $mun\{$ ,элементарный  $mun\}$ :) определение указателя на функцию  $\rightarrow$ fn сигнатура функции сигнатура функции  $\rightarrow$  (спис форм парам):элементарный тип  $cnuc \not fopm napam \rightarrow \not fopm napam \{, \not fopm napam \}$ форм парам  $\rightarrow$  val списид:простейшее определение типа ref cnucud:npocmeйшее определение muna/ const ref списид:простейшее определение типа onpederenue алгебраического  $muna \rightarrow часть$  алгебр  $muna\{..., часть$  алгебр  $muna\}$ часть алгебр типа  $\rightarrow$ перечисление|структура nepeuceneue →**enum** ид $\{cnucud\}$  $cmpyкmypa \rightarrow struct$  ид $\{cnuc\ rpynn\ noneŭ\}$ cnuc групп полей  $\rightarrow$ группа полей $\{;$ группа полей $\}$ группа полей  $\rightarrow$  cnucud:простейшее определение типа функция  $\rightarrow$ **fn** ид сигнатура функции $\{$ тело блока $\}$  $npomomun \rightarrow \mathbf{fn}$  ид  $curhamypa\_ функции;$ константы  $\rightarrow$  const определение константы $\{$ , определение константы $\}$ определение константы  $\rightarrow$ ид:простейшее определение типа = выр mело блока  $\rightarrow$  { $nepemenhbe|munb||phyhkuux||npomomun||kohcmahmbe||onepamopb||}$  $eup \rightarrow eup_0$  [? $eup_0$ : $eup_0$ ]  $ewp_0 \rightarrow ewp_1\{(|||!|||^{^{-}}|!^{^{-}})ewp_1\}$  $ewp_1 \rightarrow ewp_2 \{ (\&\& | !\&\&) ewp_2 \}$  $6bip_2 \rightarrow \{!\}6bip_3$  $6bip_3 \rightarrow 6bip_4 \{ (<|>|<=|>=|=|!=) 6bip_4 \}$  $eup_4 \rightarrow eup_5 \{(I|~I|~|~~)eup_5\}$  $ebip_5 \rightarrow ebip_6 \{ (\&|^{\sim}\&| \ll |\gg|) \otimes bip_6 \}$  $6blp_6 \rightarrow {^{\sim}}6blp_7$  $sup_7 \rightarrow sup_8 \{(+/-)sup_8\}$  $\varepsilon \omega p_8 \rightarrow \varepsilon \omega p_9 \{ (*//\%/.) \varepsilon \omega p_9 \}$  $\varepsilon \omega p_9 \rightarrow \varepsilon \omega p_{10} [(**/**.)\varepsilon \omega p_9]$  $ewp_{10} \to [+/-/\#]ewp_{11}$ 

 $выp_{11} \rightarrow выp_{12}$  [## $выp_{12}$ ]

```
6 i i p_{12} \rightarrow целое/вещественное/true/false/строковое/(6 i i p)/имя/сост знач
cocm знач \rightarrowзнач cmpy\kappa m|знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nors\{, started or star
значение поля \rightarrowид <- выр
знач массива \rightarrowarray[:выр,элементарный тип:]{списвыр}
uмя \rightarrow [[:uмя\_моdуля::] [uмя\_muna::uмя uacmu umuna::]uд{([cnucвыp])|[cnucвыp]|@|.uд}
cnuceup \rightarrow eup\{,eup\}
имя модуля \rightarrowид
uмя muna \rightarrow uд
uмя чacmu muna \rightarrowuд
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop 
ightarrow ukna | npucваивания | nepexoda | ввода | вывода | условный | разбора
условный \rightarrowif выр then\{mело блока\}{elif выр then\{mело блока\}\}[else\{mело блока\}]endif
uu\kappa na \rightarrow [\langle :u\pi :\rangle] (for u\pi : eup\{meno\ блока\}|for u\pi := eup,eup[,eup]\{meno\ блока\}|
                    while выр\{mело блока\}|repeat\{mело блока\}until выр)
присваивания \to ums := выр
nepexoda \rightarrow break [ид]/continue [ид]/return [выр]
вво∂а \rightarrow \mathbf{read}(cnucum\ddot{e}H)
вывода \rightarrowprint(cnucвыр)
cnucum\ddot{e}_{H} \rightarrow ums\{,ums\}
разбора \rightarrow match выр {образец ->{mело блока}{,образец ->{mело блока}} [else{mело блока}]}
образец \rightarrow []:имя модуля::]имя типа::имя части типа\{..\}
      Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
yenoe \rightarrow decsmuuhoe|восьмеричное|deouvhoe|mecmhadyamepuuhoe
\partial e c s m u u ho e \rightarrow \partial e c u u \phi p a ('? \partial e c u u \phi p a)^*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
\partial воичное \rightarrow 0(b|B)\partial воичн цифра ('?\partial воичн цифра)*
uecmhaduamepuчhoe \rightarrow 0(x|X)uecmh uufpa('?uecmh uufpa)*
\partial ec \quad uu\phi pa \rightarrow 0|1|2|3|4|5|6|7|8|9
восьм иифра \rightarrow 0/1/2/3/4/5/6/7
\partialвоичн цифра \rightarrow 0/1
uecm h uu fpa \rightarrow 0 |1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F
вещественное \toцелая часть(дробная часть)?(символ порядка(+/-)?)порядок)?
символ порядка \rightarrow E|e|S|s|D|d|Q|q
целая часть \rightarrow \partial e cятичное
дробная часть \rightarrow. десятичное
nopядoк \rightarrow decятичноe
```

Eсли перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано u32. Eсли указано u16, то строковый литерал — в кодировке UTF-16. Наконец, если указано u8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

```
Задача 1.3. С помощью генератора лексических анализаторов Мяука для приводимого ниже
языка напишите и протестируйте лексический анализатор.
программа \to модуль ид {[использует списид] [предоставляет списид] тело модуля}
cnucud \rightarrow uд{,uд}
тело модуля \rightarrow {переменные | типы | функция | прототип | константы }
nepeменные \rightarrow перем cnucud: npocmeŭuee onpedeление muna\{; cnucud: npocmeŭuee onpedeление muna\}
munu \rightarrow \mathbf{TИП} \, \mathbf{ИД} = onpedenenue \, muna\{; \mathbf{ИД} = onpedenenue \, muna\}
onpedeление muna 	o npocme
uuee onpedеление muna|cocmaвное onpedеление muna|cocmasное onpedеление onpedелени
простейшее определение типа \rightarrowэлементарный тип\mid
                                                                                 массив[[выр]{,[выр]}] простейшее определение типа
элементарный mun \rightarrow \{@\} (встроенный mun \mid \mu \mu)
6cmpoeнный mun \rightarrow \{большое|маленькое\} (цел|вещ|беззн|лог|симв|строка)|ничто|байт|цел8|
                                                цел16|цел32|цел64|цел128|беззн8|беззн16|беззн32|беззн64|беззн128|
                                                вещ32|вещ64|вещ128|лог8|лог16|лог32|лог64|симв8|симв16|симв32|
                                                строка8/строка16/строка32
cocmaвноe определение muna \rightarrow onpedenenue kopme \rightarrow color pedenenue указателя на функцию
                                                                             определение алгебраического типа
определение кортежа \rightarrow (:элементарный mun\{,элементарный mun\}:)
определение _ указателя _ на _ функцию 
ightarrow\phiункция сигнатура _ функции
cuгнатура функции \rightarrow (cnuc форм napam):элементарный mun
cnuc \ \phi opM \ napaM \rightarrow \phi opM \ napaM \{, \phi opM \ napaM \}
форм парам \rightarrowзначение списид:простейшее определение типа
                                    ссылка списид:простейшее определение типа
                                    конст ссылка списид:простейшее определение типа
определение алгебраического типа \rightarrow часть алгебр типа\{...часть алгебр типа\}
часть алгебр типа \rightarrowперечисление|структура
neperuc_nehue \rightarrow neperuc_nehue ид\{cnucud\}
cmpyкmypa \rightarrow ctpyкtypa ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud: простейшее определение типа
функция \to \phiункция ид сигнатура функции\{тело блока\}
npomomun \rightarrow \phiункция ид сигнатура функции;
константы \rightarrowконст определение константы\{, определение константы\}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow {переменные | типы | функция | прототип | константы | операторы}
eup \rightarrow eup_0 [?eup_0:eup_0]
eup_0 \rightarrow eup_1\{(|||!|||^{^{-}}|!^{^{-}})eup_1\}
6bip_1 \rightarrow 6bip_2 \{ (\&\& | !\&\&) 6bip_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4 \{ (<|>|<=|>=|=|!=) 6bip_4 \}
eup_4 \rightarrow eup_5 \{(I|^{\sim}I|^{\wedge}|^{\sim})eup_5\}
eup_5 \rightarrow eup_6 \{ (\&|^{\sim}\&| \ll |\gg|) \otimes up_6 \}
eup_6 \rightarrow {^{\sim}}eup_7
sup_7 \rightarrow sup_8 \{(+|-)sup_8\}
sup_8 \to sup_9 \{ (*//\%/.)sup_9 \}
\varepsilon \omega p_9 \rightarrow \varepsilon \omega p_{10} [(**/**.)\varepsilon \omega p_9]
ewp_{10} \to [+/-/\#]ewp_{11}
eup_{11} \rightarrow eup_{12} [##eup_{12}]
6 wp_{12} \to \text{целое} | \text{вещественное} | \textbf{истина} | \textbf{ложь} | \text{строковое} | (6 wp) | ums | cocm знач
cocm знач \rightarrowзнач cmpykm|знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nors\{, started or star
значение nоля \to ид < - выр
знач массива \rightarrow массив[:выр,элементарный mun:]{cnucвыр}
uмя \rightarrow [[:uмя\_модуля::][uмв\_muna::uмя части типа::]ид{([cnucвыр])|[cnucвыр]]@].ид}
cnuceup \rightarrow eup\{,eup\}
```

```
uмя muna 
ightarrow uд
uмя uасmи mиnа \rightarrowид
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop \rightarrow uu\kappa \wedge a | npucaauahun | nepexoda | aboda | buboda | yc \wedge obhuŭ | pastopa
условный \rightarrowесли выр то\{тело блока\}\{инес выр то\{тело блока\}\}\{[иначе\{тело блока\}\}\}всё
uu\kappa \wedge a \to [<:u\pi:>] (для ид из выр\{me \wedge o \text{ блока}\}[завершение\{me \wedge o \text{ блока}\}]
           для ид := 6ыp, 6ыp [, 6ыp] { meno блока } [завершение { meno блока }] |
           пока выр{тело блока}|повторяй{тело блока}покуда выр)
присваивания \to ums := выр
nepexoda \to \mathbf{выйди} [из ид] |\mathbf{продолжи} [ид] |\mathbf{возврат}| [выр]
ввода \rightarrow \mathbf{ввод}(списимён)
выводa \rightarrowвывод(cnuceup)
cnucum\ddot{e}_{H} \rightarrow ums\{,ums\}
разбора \rightarrowразбор выр\{образец \rightarrow\{mело блока\}\{,образец \rightarrow\{mело блока\}\}[иначе\{mело блока\}]<math>\}
образец \rightarrow [[:имя модуля::]имя типа::имя части типа\{...\}
   Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
uenoe \rightarrow decsmuuhoe|восьмеричное|двоичное|шестнадцатеричное|
\partial e c s m u ч ho e \rightarrow \partial e c \quad u u \phi p a ('? \partial e c \quad u u \phi p a)^*
восьмеричное \rightarrow 0овосьм_ цифра('?восьм цифра)*
\partial воичное \rightarrow 0(b|B)\partial воичн цифра ('?\partial воичн цифра)*
uecmhaduamepuчhoe \rightarrow 0(x|X)uecmh uuфpa('?uecmh uuфpa)*
\partial ec \quad uu\phi pa \rightarrow 0|1|2|3|4|5|6|7|8|9
восъм цифра \rightarrow 0/1/2/3/4/5/6/7
\partialвоичн цифра \rightarrow 0/1
uecm H uu фра \rightarrow 0/1/2/3/4/5/6/7/8/9/a/b/c/d/e/f/A/B/C/D/E/F
вещественное \to целая_ часть (дробная_ часть)?(символ порядка (+/-)?порядок)?
символ порядка \rightarrow E|e|S|s|D|d|Q|q
целая часть \rightarrow \partial e cятичное
дробная часть \rightarrow. десятичное
nopядoк \rightarrow decятичноe
```

uмя модуля  $\rightarrow$ ид

Простой строковый литерал — это любая (в том числе и пустая) последовательность символов, заключённая в одинарные кавычки. Если в строковом литерале нужно указать саму одинарную кавычку, то эту кавычку нужно удвоить. Символьный литерал представляет собой либо заключённый в одинарные кавычки символ, либо символ, заданный своим кодом. Последнее представляет собой знак \$, после которого идёт целое число, являющееся кодом символа. Число это может быть двоичным, восьмеричным, шестнадцатеричным, или десятичным Внутреннее представление строковых и символьных литералов — в кодировке Unicode, а именно, в её четырёхбайтовом варианте. Текст программы — только в кодировке UTF-8. Строковый литерал может представлять собой как простой строковый литерал, так и чередование последовательности заданнных своими кодами символов и простых строковых литералов.

Если перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано и32. Если указано и16, то строковый литерал — в кодировке UTF-16. Наконец, если указано и8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

**Задача 1.4.** С помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор.

```
программа \rightarrow module ид {[import cnucud] [export cnucud] тело_модуля} списид \rightarrowид{,ид} тело_модуля \rightarrow{переменные|типы|функция|прототип|константы} переменные \rightarrow var cnucud:простейшее определение типа};
```

```
munu \rightarrow type ид = определение muna\{;ид = определение muna\}
onpederehue muna 
ightarrow npocmeŭwee onpederehue muna | cocmaвное onpederehue muna
простейшее определение типа \rightarrow элементарный тип\mid
                                                              аттау[[выр]{,[выр]}] простейшее определение типа
элементарный mun \rightarrow \{@\} (встроенный mun \mid \mu \mu)
gcmpoeнный mun \rightarrow \{long|short\} (int|float|uint|bool|char|string)|void|byte|int8 <math>t|int16 t|
                                     int32 t|int64 t|int128 t|uint8 t|uint16 t|uint32 t|uint64 t|
                                     uint128 t|float32 t|float64 t|float128 t|bool8 t|bool16 t|bool32 t|
                                     bool64\ t|char8\ t|char16\ t|char32\ t|string8\ t|string16\ t|string32\ t|
cocmaвное определение muna \rightarrow onpederenue kopme<math>cocmap eq onpederenue cocmap eq onpederenue 
                                                           определение алгебраического типа
определение кортежа \rightarrow (:элементарный mun\{,элементарный mun\}:)
onpedeление_указателя_на_функцию \rightarrow fn сигнатура функции
сигнатура функции \rightarrow (спис форм парам):элементарный тип
cnuc форм napaM \rightarrow \phiорм napaM \{, \phiорм napaM \}
форм парам \rightarrow val списид:простейшее определение типа
                            ref cnucud:npocmeйшее определение muna/
                            const ref cnucud:простейшее определение типа
onpedenetue алгебраического muna \rightarrow часть алгебр muna\{..., часть алгебр muna\}
часть алгебр типа \rightarrowперечисление|структура
nepeucsehue →enum ид\{cnucud\}
cmpyкmypa \rightarrow struct ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud:простейшее определение типа
функция \rightarrowfn ид сигнатура функции\{тело блока\}
npomomun \rightarrow fn ид curhamypa функции;
константы \rightarrowconst определение константы{,определение константы}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow{переменные|типы|функция|прототип|константы|операторы}}
eup \rightarrow eup_0 [?eup_0:eup_0]
eup_0 \rightarrow eup_1\{(II|!II|^{\hat{}}|!^{\hat{}})eup_1\}
sup_1 \rightarrow sup_2 \{ (\&\&/!\&\&) sup_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4 \{ (<|>|<=|>=|=|!=) 6bip_4 \}
6bip_4 \rightarrow 6bip_5 \{(I|^*I|^*|^*) 6bip_5\}
\operatorname{sup}_5 \to \operatorname{sup}_6 \{ (\& / \& | \ll | \gg | \gg) \operatorname{sup}_6 \}
eup_6 \rightarrow {^{\sim}}eup_7
sup_7 \rightarrow sup_8 \{(+|-)sup_8\}
\varepsilon \omega p_8 \rightarrow \varepsilon \omega p_9 \{ (*//\%/.) \varepsilon \omega p_9 \}
\varepsilon \omega p_9 \rightarrow \varepsilon \omega p_{10} [(**/**.)\varepsilon \omega p_9]
6bip_{10} \rightarrow [+/-/\#] 6bip_{11}
eup_{11} \rightarrow eup_{12} [##eup_{12}]
6ыp_{12} \rightarrow целое|вещественное|true|false|строковое|(6ыp)|имя|сост знач
cocm знач \rightarrowзнач cmpy\kappa m|знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nons\{, started{started}
значение поля \toид <- выр
знач массива \rightarrowarray[:выр,элементарный mun:]{cnucвыр}
uмя \rightarrow [[:uмя модуля::][uмя типа::uмя части типа::]ид{([cписвыр])|[cписвыр]]@[.ид}
cnuceup \rightarrow eup\{,eup\}
uмя модуля \rightarrowид
uмя muna \rightarrow uд
onepamop \omega \rightarrow [onepamop] \{; [onepamop] \}
onepamop \rightarrow uu\kappa \wedge a | npucaauahun | nepexoda | aboda | buboda | yc \wedge obhuŭ | pastopa
условный \rightarrowif выр then\{mело блока\}\{elif выр then\{mело блока\}\}\{else\{mело блока\}\}endif
```

```
uu\kappa \wedge a \rightarrow [\langle :u\pi : \rangle] (for u\pi : eup\{me \wedge o \wedge o \wedge a\}[else\{me \wedge o \wedge o \wedge a\}]]
           for ид := выр, выр[,выр]{meno\_блока}[else{meno\_блока}]]
           while выр{тело блока}|repeat{тело блока}until выр)
присваивания \to ums := выр
nepexoda \rightarrow break [ид]/continue [ид]/return [еыр]
ввода \rightarrowread(списимён)
вывода \rightarrowprint(cnucвыр)
cnucum\ddot{e}_{H} \rightarrow ums\{,ums\}
разбора \rightarrow match выр {образец ->{mело блока}{,образец ->{mело блока}} [else{mело блока}]}
образец \rightarrow []:имя модуля::]имя типа::имя части типа\{..\}
   Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
uenoe \rightarrow decsmuuhoe|восьмеричное|двоичное|шестнадцатеричное|
\partial e c я m u ч ho e \rightarrow \partial e c \quad u u \phi p a ('? \partial e c \quad u u \phi p a)^*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
\partial воичное \rightarrow 0(b|B)\partial воичн цифра ('?\partial воичн цифра)*
uecmhaduamepuчhoe \rightarrow 0(x|X)uecmh uuфpa('?uecmh uuфpa)*
\partial ec \quad uu\phi pa \rightarrow 0|1|2|3|4|5|6|7|8|9
восьм uu\phi pa \to 0|1|2|3|4|5|6|7
\partialвоичн цифра \rightarrow 0/1
uecm H uu фра \rightarrow 0/1/2/3/4/5/6/7/8/9/a/b/c/d/e/f/A/B/C/D/E/F
вещественное \rightarrowцелая часть(дробная часть)?(символ порядка(+|-)?порядок)?
символ порядка \rightarrow E|e|S|s|D|d|Q|q
целая часть \rightarrow \partial e cятичное
дробная часть \rightarrow. десятичное
nopядoк \rightarrow decятичноe
```

Если перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано u32. Если указано u16, то строковый литерал — в кодировке UTF-16. Наконец, если указано u8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

**Задача 1.5.** С помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор.

```
программа \to модуль ид {[использует списид] [предоставляет списид] тело_модуля} списид \toид{,ид} тело_модуля \to{переменные|типы|функция|прототип|константы} переменные \toперем списид:простейшее_определение_типа{;списид:простейшее_определение_типа} типы \toтип ид = определение_типа{;ид = определение_типа} определение_типа \toпростейшее_определение_типа|составное_определение_типа простейшее_определение_типа \toлементарный_тип|

массив[[выр]{,[выр]}]простейшее_определение_типа \to элементарный тип \to4@}(встроенный тип|ид)
```

```
6cmpoehhhiй mun \to \{6onbinoe| маленькое\} (цел| bein| bei
                                    цел16|цел32|цел64|цел128|беззн8|беззн16|беззн32|беззн64|беззн128|
                                    вещ32|вещ64|вещ128|лог8|лог16|лог32|лог64|симв8|симв16|симв32|
                                    строка8/строка16/строка32
cocmaвное определение muna \rightarrow onpedenenue kopme \rightarrow color pedenenue указателя на функцию
                                                          onpedeление_ алгебраического muna
определение кортежа \rightarrow (:элементарный mun\{,элементарный mun\}:)
определение указателя на функцию \rightarrowфункция сигнатура функции
cuгнатура\_функции \to (cnuc\_форм\_парам):элементарный_тип
cnuc форм napam \rightarrow \phiорм napam \{, \phiорм napam \}
форм парам \rightarrowзначение списид:простейшее определение типа
                           ссылка списид:простейшее определение типа
                           конст ссылка списид:простейшее определение типа
onpederenue алгебраического muna \rightarrow часть алгебр muna\{..., часть алгебр muna\}
часть алгебр типа \rightarrowперечисление|структура
neperuc_nehue \rightarrow neperuc_nehue ид\{cnucud\}
cmpyкmypa \rightarrow ctpyкtypa ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud:простейшее определение типа
функция \to \phiункция ид сигнатура функции\{тело блока\}
npomomun \rightarrow \phi yнкция ид сигнатура функции;
константы \toконст определение константы\{,определение константы\}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow {переменные | типы | функция | прототип | константы | операторы}
eup \rightarrow eup_0 [?eup_0:eup_0]
eup_0 \rightarrow eup_1\{(|||!|||^{^{-}}|!^{^{-}})eup_1\}
eup_1 \rightarrow eup_2 \{ (\&\&/!\&\&) eup_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4\{(<|>|<=|>=|=|!=)6bip_4\}
6bip_4 \rightarrow 6bip_5 \{(||^{\sim}||^{\wedge}|^{\sim})6bip_5\}
eup_5 \rightarrow eup_6 \{ (\& | ~\& | \ll | \gg | \gg) eup_6 \}
6blp_6 \rightarrow {^{\sim}}6blp_7
ewp_7 \rightarrow ewp_8\{(+|-)ewp_8\}
sup_8 \to sup_9 \{ (*//\%/.) sup_9 \}
6blp_9 \rightarrow 6blp_{10} [(**|**.)6blp_9]
6bip_{10} \rightarrow [+/-/\#] 6bip_{11}
6bip_{11} \rightarrow 6bip_{12} [##6bip_{12}]
выp_{12} \rightarrow целое|вещественное|истина|ложь|строковое|(eыp)|имя|сост знач
cocm знач \rightarrowзнач cmpy\kappa m|знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nons\{, statement statement
значение nоля \rightarrowид <- \epsilonыр
знач массива \to массив[:выр,элементарный тип:]{списвыр}
uмя \rightarrow [[:uмя модуля::][uмя типа::uмя части типа::]ид{([списвыр])|[списвыр]]@[.ид}
cnucвыр \rightarrow выр{,выр}
uмя мо\partial yля \rightarrowид
uмя muna \rightarrow uд
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop 
ightarrow ukna | npucваивания | nepexoda | ввода | вывода | условный | разбора
условный \rightarrowесли выр \mathbf{To} \{ тело блока\} \{ \mathbf{unec} выр \mathbf{To} \{ тело блока\}\} \{ \mathbf{unaue} \{ тело блока\}\} \} \mathbf{Ec}
пока выр{тело блока}|повторяй{тело блока}покуда выр)
присваивания \rightarrowимя (:=|+:=|-:=|*:=|/:=|/.:=|**:=|**.:=|%:=|&:=|~&:=|\ll:=|\ll:=||>:=|1:=|^-|:=|^-:=|
                                    ||:=|!||:=|^^:=|!^^:=|&&:=|!&&:=|>>>:=) выр
nepexoda \to \mathbf{выйди} [из ид] |\mathbf{npoдoлжu}| [ид] |\mathbf{возврат}| [выр]
660 \partial a \rightarrow \mathbf{ввод}(cnucum\ddot{e}H)
6ы60 \partial a \rightarrowвывод(cnuceup)
```

```
cnucum\ddot{e}h \rightarrow ums\{,ums\}
разбора \toразбор выр\{образец ->\{ тело блока\}\{,образец ->\{ тело блока\}\}\{ [ иначе\{ тело блока\}\}\}
образеu \to []:uмя модуля::]uмя muna::umя uacmu muna\{...\}
   Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
uenoe \rightarrow decsmuuhoe|bocьмерииное|deouuhoe|mecmhaduamepuuhoe
\partial ecsmuчнoe \rightarrow \partial ec цифра('?\partial ec цифра)*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
dвоичное \rightarrow 0(b|B)dвоичн_ цифра ('?dвоичн_ цифра)*
uecmhaduamepuчhoe \rightarrow 0(x|X)uecmh uuфpa('?uecmh uuфpa)*
\partial ec \quad uu \phi pa \rightarrow 0/1/2/3/4/5/6/7/8/9
восьм иифра \rightarrow 0/1/2/3/4/5/6/7
\partialвоичн ци\phiра \rightarrow 0/1
uecm h uu fpa \rightarrow 0 |1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F
вещественное \toцелая часть (дробная часть)?(символ порядка(+|-)?порядок)?
символ порядка \rightarrow E|e|S|s|D|d|Q|q
целая часть \rightarrow \partial e cятичное
дробная часть \rightarrow. десятичное
nopядoк \rightarrow decяmuчнoe
```

Если перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано u32. Если указано u16, то строковый литерал — в кодировке UTF-16. Наконец, если указано u8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

**Задача 1.6.** С помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор.

```
программа \rightarrow module ид {[import cnucud] [export cnucud] тело модуля}
cnucud \rightarrow uд{,uд}
тело модуля \rightarrow {переменные | munы | функция | прототип | константы }
nepeменные \rightarrow var списид:npocmeйшее onpeделение muna {;cnucud:npocmeйшее onpeделение muna}
munu \rightarrow type \, ud = onpedenenue \, muna \{; ud = onpedenenue \, muna\}
onpederehue muna 
ightarrow npocmeŭwee onpederehue muna | cocmaвное onpederehue muna
простейшее определение типа \rightarrow элементарный тип\mid
                                      array[[выр]{,[выр]}] простейшее определение типа
элементарный mun \rightarrow \{@\} (встроенный mun \mid \mu Д)
6cmpoeнный mun \rightarrow \{long|short\} (int|float|uint|bool|char|string)|void|byte|int8 <math>t|int16 t|
                      int 32 t | \text{int} 64 \ t | \text{int} 128 \ t | \text{uint} 8 \ t | \text{uint} 16 \ t | \text{uint} 32 \ t | \text{uint} 64 \ t |
                      uint128 t|float32 t|float64 t|float128 t|bool8 t|bool16 t|bool32 t|
                      bool64 t/char8 t/char16 t/char32 t/string8 t/string16 t/string32 t
cocmaвное определение muna \rightarrow onpederenue kopme \rightarrow calonpederenue указателя на функцию
                                    определение алгебраического типа
определение кортежа \rightarrow (:элементарный mun\{,элементарный mun\}:)
определение указателя на функцию \rightarrowfn сигнатура функции
```

```
cuгнатура функции \rightarrow (cnuc форм napam):элементарный mun
cnuc форм napaM \rightarrow \phi opM napaM \{, \phi opM napaM \}
форм парам \rightarrow val списид:простейшее определение типа
                  ref cnucud:npocmeйшее определение muna/
                  const ref cnucud:простейшее определение типа
onpedenehue\_aлгебрauческого\_muna 	oчасmb aлгебр muna \{...часmb aлгебр muna \}
часть алгебр типа \rightarrow перечисление |структура
nepeuucnehue \rightarrow enum ид\{cnucud\}
cmpyкmypa \rightarrow struct ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud:простейшее определение типа
функция \rightarrowfn ид сигнатура функции {тело блока}
npomomun \rightarrow \mathbf{fn} ид сигнатура функции;
\kappaонстанты \rightarrow const определение \kappaонстанты\{, определение \kappaонстанты\}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow{переменные|типы|функция|прототип|константы|операторы}}
eup \rightarrow eup_0 [?eup_0:eup_0]
eup_1 \rightarrow eup_2 \{ (\&\& | !\&\&) eup_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4 \{ (<|>|<=|>=|=|!=) 6bip_4 \}
eup_4 \rightarrow eup_5 \{(I|^{\sim}I|^{\wedge}|^{\sim})eup_5\}
eup_5 \rightarrow eup_6 \{ (\&/~\&/\ll/\gg/\gg) eup_6 \}
eup_6 \rightarrow {^{\sim}}eup_7
sup_7 \rightarrow sup_8 \{(+|-)sup_8\}
sup_8 \to sup_9 \{ (*//\%/.) sup_9 \}
\varepsilon \omega p_9 \rightarrow \varepsilon \omega p_{10} [(**|**.)\varepsilon \omega p_9]
ewp_{10} \to [+/-/\#]ewp_{11}
eup_{11} \rightarrow eup_{12} [##eup_{12}]
выp_{12} \rightarrowцелое|вещественное|true|false|строковое|(выp)|имя|сост знач
cocm знач \rightarrowзнач cmpykm/знач массива
знач cmpyкm \to []:ums modyns::]ums muna::ums uscmu muna{shavehue} nons{shavehue} nons{shavehue}
значение nоля \rightarrowид <- \epsilonыр
знач массива \rightarrowarray[:выр,элементарный mun:]{cnucвыр}
uмя \rightarrow [[:uмя модуля::][uмя типа::uмя части типа::]ид{([списвыр])|[списвыр]]@[.ид}
cnuceup \rightarrow eup\{,eup\}
uмя модуля \rightarrowид
uмя muna \rightarrow uд
uмя uacmu muna \rightarrowuд
onepamopы \rightarrow [onepamop] \{ : [onepamop] \}
onepamop \rightarrow uu\kappa \wedge a | npucaauahun | nepexoda | aboda | buboda | yc \wedge obhuŭ | pastopa
условный \rightarrowif выр then\{mело блока\}\{elif выр then\{mело блока\}\}\{else\{mело блока\}\}endif
uu\kappa na \rightarrow [\langle :u\pi : \rangle] (for u\pi : esp\{meno\ блока\}| for u\pi : = esp,esp[,esp]\{meno\ блока\}|
          while выр{тело блока}|repeat{тело блока}until выр)
npuc в a u в a u н x \rightarrow u н x (:=|+:=|-:=|*:=|/:=|/.:=|**:=|**:=|%:=|^* \& :=|^* \& :=| > :=||:=|^*|:=|^*:=|^* :=|
                        ||:=|!||:=|^^:=|!^^:=|&&:=|!&&:=|>>>:=) выр
nepexoda \rightarrow break [ид]/continue [ид]/return [выр]
ввода \rightarrowread(списимён)
вывода \rightarrowprint(cnucвыр)
cnucum\ddot{e}н \rightarrowumя\{,umя\}
разбора \rightarrow match выр{образец ->{mело блока}{,образец ->{mело блока}}[else{mело блока}]}
образеu \to []:имя модуля::]имя типа::имя части типа\{...\}
   Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
uenoe \rightarrow decsmuuhoe|восьмерииное|deouuhoe|шестнадцатерииное|
\partial ecsmuчнoe \rightarrow \partial ec цифра('?\partial ec цифра)*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
```

```
 \begin{tabular}{ll} $\operatorname{deouvhoe} \to 0(b|B)\operatorname{deouvh}\_uu\phi pa\end{tabular} (??deouvh\_uu\phi pa\end{tabular})^* \\ wecmhaduamepuvhoe \to 0(x|X)wecmh\_uu\phi pa\end{tabular} (??wecmh\_uu\phi pa\end{tabular})^* \\ dec\_uu\phi pa \to 0|1|2|3|4|5|6|7|8|9 \\ eochhau\_uu\phi pa \to 0|1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F \\ deouvh\_uu\phi pa \to 0|1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F \\ deuecmhauhoe \to uenah\_vacmb\end{tabular} (dpodhahauhoe)?(cumbon\_nophaha(+|-)?nophaha)? \\ cumbon\_nophaha \to E|e|S|s|D|d|Q|q \\ uenah\_vacmb \to dechmuvhoe \\ dpodhahauhoe \to dechmuvhoe \\ nophaha \to dechmuvhoe \\ \end{tabular}
```

Eсли перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано и32. Eсли указано и16, то строковый литерал — в кодировке UTF-16. Наконец, если указано и8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

**Задача 1.7.** С помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор.

```
программа \to модуль ид {[использует списид] [предоставляет списид] тело модуля}
cnucud \rightarrow uд{.uд}
тело модуля \rightarrow {переменные | munы | функция | прототип | константы }
nepeменные \rightarrowперем cnucud:npocmeŭuee onpedeление <math>muna\{;cnucud:npocmeŭuee onpedeлeниe <math>muna\}
munu \rightarrow \mathbf{TИП} \ \mathbf{ИД} = onpederenue \ muna \{; \mathbf{ИД} = onpederenue \ muna \}
onpedeление muna 	o npocme
uuee onpedеление muna|cocmaвное onpedеление muna|cocmasное onpedеление onpedелени
простейшее определение типа \rightarrow элементарный тип\mid
                                                                         массив[[выр]{,[выр]}] простейшее определение типа
элементарный mun \rightarrow \{@\} (встроенный mun \mid \mu \mu)
gcmpoehhhi mun 	o \{большое|маленькое\} (цел|вещ|беззн|лог|симв|строка)|ничто|байт|цел|8|
                                           цел16|цел32|цел64|цел128|беззн8|беззн16|беззн32|беззн64|беззн128|
                                           вещ32|вещ64|вещ128|лог8|лог16|лог32|лог64|симв8|симв16|симв32|
                                           строка8/строка16/строка32
cocmaвноe определение muna \rightarrow onpedenenue kopme \rightarrow color pedenenue указателя на функцию
                                                                     определение алгебраического muna
определение кортежа \rightarrow (:элементарный mun\{,элементарный mun\}:)
определение указателя на функцию 
ightarrow \mathbf{\phi} \mathbf{y} \mathbf{n} \mathbf{k} \mathbf{u} \mathbf{n} сигнатура функции
cuгнатура функции \rightarrow (cnuc форм napam):элементарный mun
cnuc \not fopm napam \rightarrow \not fopm napam \{, \not fopm napam \}
\phiорм парам \rightarrowзначение списид:простейшее определение типа
                                ссылка списид:простейшее определение типа
                                 конст ссылка списид:простейшее определение типа
определение алгебраического типа \rightarrow часть алгебр типа\{...часть алгебр типа\}
часть алгебр типа \rightarrowперечисление|структура
перечисление \rightarrowперечисление ид\{cnucud\}
```

```
cmpyкmypa \rightarrow ctpyкtypa ид\{cnuc\ rpynn\ noneŭ\}
cnuc\_групn\_полей \to группа\_полей \{; группа\_полей \}
группа полей \rightarrow cnucud:простейшее определение типа
\phiункция \to \phiункция ид сигнатура \phiункции\{тело \phiока\}
npomomun \rightarrow \mathbf{dyhkuu} ид сигнатура функции;
константы \toконст определение константы \{, определение константы\}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow{переменные|типы|функция|прототип|константы|операторы}
eup \rightarrow eup_0 [?eup_0:eup_0]
esip_0 \rightarrow esip_1\{(II|!II|^{\hat{}})|esip_1\}
eup_1 \rightarrow eup_2 \{ (\&\& | !\&\&) eup_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4 \{ (<|>|<=|>=|=|!=) 6bip_4 \}
eup_4 \rightarrow eup_5 \{(I | ~I | ~I | ~I \sim )eup_5 \}
eup_5 \rightarrow eup_6 \{ (\&| \tilde{\&}| \ll | \gg | \gg) eup_6 \}
eup_6 \rightarrow {^{\sim}}eup_7
ewp_7 \rightarrow ewp_8\{(+|-)ewp_8\}
\varepsilon \omega p_8 \rightarrow \varepsilon \omega p_9 \{ (*//\%/.) \varepsilon \omega p_9 \}
\varepsilon \omega p_9 \rightarrow \varepsilon \omega p_{10} [(**/**.)\varepsilon \omega p_9]
выp_{10} \to [+/-/\#] выp_{11}
выp_{11} \rightarrow выp_{12} [##выp_{12}]
6 bip_{12} \to \text{целое} | \text{вещественное} | \textbf{истина} | \textbf{ложь} | \text{строковое} | (6 bip) | uma | cocm знач
cocm знач \rightarrowзнач cmpykm/знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nons\{.значение nons\}\}
значение поля \rightarrowид <- выр
знач массива \rightarrow массив[:выр,элементарный mun:]{cnucвыр}
uмя \rightarrow [[:uмя модуля::][uмя типа::uмя части типа::]ид{([списвыр])|[списвыр]]@[.ид}
cnuceup \rightarrow eup\{,eup\}
uмя модуля \rightarrowид
uмя muna \rightarrow uд
one pamop \uplus \rightarrow [one pamop] \{; [one pamop] \}
onepamop \rightarrow uu\kappa na | npuc ва uвания | nepexoda | ввода | вывода | условный | разбора
условный \rightarrowесли выр \mathbf{To} \{ тело блока\} \{ \mathbf{unec} выр \mathbf{To} \{ тело блока\}\} \{ \mathbf{unaue} \{ тело блока\}\} \} \mathbf{Ec}
uu\kappa \Lambda A \rightarrow [<:u\pi:>] (для ид из u\pi A \cap \{u\pi\} [завершение \{u\pi\} блока \{u\pi\}]
           для ид := выр, выр[, выр] \{ mело блока \} [ завершение\{ mело блока \} ] [
           пока выр{тело блока}|повторяй{тело блока}покуда выр)
npuc в a u в a u н x \rightarrow u н x (:=|+:=|-:=|*:=|/:=|/::=|**:=|%:=|%:=|^*\&:=|%:=|%:=|%:=||:=|^*:=|^*:=|
                          |\ |\ |:=|!\ |\ |:=|^{-}:=|!\ ^{-}:=|\&\&:=|!\&\&:=|\ggg:=|
nepexoda \to \mathbf{выйди} [из ид] продолжи [ид] возврат [еыр]
660 \partial a \rightarrow \mathbf{ввод}(cnucum\ddot{e}H)
вывод a \rightarrowвывод(cnuceup)
cnucum\ddot{e}H \rightarrow ums\{,ums\}
разбора \rightarrowразбор выр\{образец ->\{ тело блока\}\{,образец ->\{ тело блока\}\}[ иначе\{ тело блока\}]<math>\}
образец \rightarrow []:имя модуля::]имя типа::имя части типа\{...\}
    Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
uenoe \rightarrow decsmuuhoe|восьмеричное|двоичное|шестнадцатеричное|
\partial e c s m u u ho e \rightarrow \partial e c u u \phi p a ('? \partial e c u u \phi p a)^*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
\partial воичное \rightarrow 0(b|B)\partial воичн цифра ('?\partial воичн цифра)*
uecmhaduamepuчhoe \rightarrow 0(x|X)uecmh uuфpa('?uecmh uuфpa)*
\partial ec \quad uu \phi pa \rightarrow 0/1/2/3/4/5/6/7/8/9
восъм цифра \rightarrow 0|1|2|3|4|5|6|7
\partialвоичн цифра \rightarrow 0/1
uecm H uu фра \rightarrow 0/1/2/3/4/5/6/7/8/9/a/b/c/d/e/f/A/B/C/D/E/F
вещественное \rightarrowцелая часть (дробная часть)?(символ порядка(+|-)?порядок)?
```

```
символ_ порядка \rightarrow E|e|S|s|D|d|Q|q целая_ часть \rightarrow десятичное дробная_ часть \rightarrow десятичное порядок \rightarrow десятичное
```

Если перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано и32. Если указано и16, то строковый литерал — в кодировке UTF-16. Наконец, если указано и8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

**Задача 1.8.** С помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор.

```
программа \rightarrow module ид {[import cnucud] [export cnucud] тело модуля}
cnucud \rightarrow uд{,uд}
meno модуля \rightarrow{nepemenhue|munu|pyhkuun|npomomun|kohcmahmu}
nepementue \rightarrow var списид:npocmeйшee onpedenetue muna {; cnucud:npocmeйшee onpedenetue muna }
munu \rightarrow type ид = onpedenenue muna{;ид = onpedenenue muna}
определение muna \rightarrow npocmeŭuee определение muna/cocmaвное определение muna
простейшее определение типа \rightarrowэлементарный тип\mid
                                   array[[выр]{,[выр]}] простейшее определение типа
элементарный mun \rightarrow \{@\} (встроенный mun \mid \mu \mu)
встроенный mun \rightarrow \{long|short\} (int|float|uint|bool|char|string)|void|byte|int8 t|int16 t|
                     int32 t|int64 t|int128 t|uint8 t|uint16 t|uint32 t|uint64 t|
                     uint128 t|float32 t|float64 t|float128 t|bool8 t|bool16 t|bool32 t|
                     bool64 t|char8 t|char16 t|char32 t|string8 t|string16 t|string32 t
cocmaвное определение muna \rightarrow onpederenue kopme \rightarrow calonpederenue указателя на функцию
                                 определение_ алгебраического типа
определение кортежа \rightarrow (:элементарный mun\{,элементарный mun\}:)
определение указателя на функцию \rightarrowfn сигнатура функции
cuгнатура\_функции \to (cnuc\_форм\_парам):элементарный тип
cnuc \not fopm napam \rightarrow \not fopm napam \{, \not fopm napam \}
форм парам \rightarrow val списид:простейшее определение типа
                ref cnucud:npocmeйшее определение muna/
                const ref cnucud:простейшее определение типа
определение алгебраического типа \rightarrow часть алгебр типа\{...часть алгебр типа\}
часть алгебр типа \rightarrowперечисление|структура
nepeuceneque → enum \muд{cnucud}
cmpyкmypa \rightarrow struct ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud:простейшее определение типа
функция \rightarrowfn ид сигнатура функции{тело блока}
npomomun \rightarrow fn ид curhamypa функции;
константы \rightarrow const определение константы\{, определение константы\}
определение константы \rightarrowид:простейшее определение типа = выр
```

```
тело блока \rightarrow {переменные | типы | функция | прототип | константы | операторы}
eup \rightarrow eup_0 [?eup_0:eup_0]
estp_0 \rightarrow estp_1\{(|||!|||^{^{-}}|!^{^{-}})estp_1\}
eup_1 \rightarrow eup_2 \{ (\&\& | !\&\&) eup_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4 \{ (<\!\! > \mid < = \mid > = \mid = \mid ! = ) 6bip_4 \}
eup_4 \rightarrow eup_5 \{(I|^{\sim}I|^{\wedge}|^{\sim})eup_5\}
eup_5 \rightarrow eup_6 \{ (\&/~\&/\ll/\gg/\gg) eup_6 \}
6blp_6 \rightarrow {\red}{sblp_7}
sup_7 \rightarrow sup_8 \{(+|-)sup_8\}
sup_8 \to sup_9 \{ (*//\%/.) sup_9 \}
\varepsilon \omega p_9 \rightarrow \varepsilon \omega p_{10} [(**/**.)\varepsilon \omega p_9]
eup_{10} \to [+/-/\#] eup_{11}
eup_{11} \rightarrow eup_{12} [##eup_{12}]
6bp_{12} \rightarrowцелое|вещественное|true|false|строковое|(6bp)|имя|сост знач
cocm знач \rightarrowзнач cmpykm/знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nons\{, started{started}
значение поля \toид <- выр
знач массива \rightarrowarray[:выр,элементарный mun:]{cnucвыр}
uмя \rightarrow [[:uмя модуля::][uмя типа::uмя части типа::]ид{([списвыр])|[списвыр]]@[.ид}
cnuceup \rightarrow eup\{,eup\}
имя модуля \rightarrowид
uмя muna \rightarrow uд
uмя чacmu muna \rightarrowuд
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop 
ightarrow ukra | npucваивания | nepexoda | ввода | вывода | условный | разбора
условный \rightarrow if выр then\{mело блока\}\{elif выр then\{mело блока\}\}\{else\{mело блока\}\}\}endif
uu\kappa na \rightarrow [<:uд:>] (for ug:выр\{meno блока\}[else\{meno блока\}]]
           for ид := выр, выр[, выр] \{ mело блока \} [else \{ mело блока \}] |
           while выр{тело блока}|repeat{тело блока}until выр)
npuc в a u в a u н x \rightarrow u н x (:=|+:=|-:=|*:=|/:=|/.:=|**:=|**:=|%:=|^* \& :=|^* \& :=| > :=||:=|^*|:=|^*:=|^* :=|
                          ||:=|!||:=|^^:=|!^^:=|&&:=|!&&:=|>>>:=) выр
nepexoda \rightarrow break [ид]/continue [ид]/return [выр]
вво∂а \rightarrow read(cnucum\"eн)
вывода \rightarrow print(cnucвыр)
cnucum\ddot{e}_{H} \rightarrow ums\{,ums\}
разбора \rightarrow match выр{образец ->{mело блока}{,образец ->{mело блока}}[else{mело блока}]}
образец \rightarrow []:имя модуля::]имя типа::имя части типа\{...\}
   Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
uenoe \rightarrow decsmuuhoe|bocьмеричное|deouuhoe|mecmhaduamepuuhoe
\partial e c s m u u ho e \rightarrow \partial e c u u \phi p a ('? \partial e c u u \phi p a)^*
восьмеричное \rightarrow 0овосьм_ uu\phi pa('?восьм_ uu\phi pa)^*
dвоичное \rightarrow 0(b|B)dвоичн цифра('?dвоичн цифра)*
uecmhaduamepuчhoe \rightarrow 0(x|X)uecmh uuфpa('?uecmh uuфpa)*
\partial ec \quad uu \phi pa \rightarrow 0/1/2/3/4/5/6/7/8/9
восьм иифра \rightarrow 0/1/2/3/4/5/6/7
\partialвоичн цифра \rightarrow 0/1
uecm h uu fpa \rightarrow 0 |1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F
вещественное \toцелая часть (дробная часть)?(символ порядка(+|-)?порядок)?
символ порядка \rightarrow E|e|S|s|D|d|Q|q
иелая часть \rightarrow \partial e cятичное
дробная часть \rightarrow. десятичное
nopядoк \rightarrow decяmuчнoe
   Простой строковый литерал — это любая (в том числе и пустая) последовательность символов,
```

Простой строковый литерал — это любая (в том числе и пустая) последовательность символов, заключённая в одинарные кавычки. Если в строковом литерале нужно указать саму одинарную кавычку, то эту кавычку нужно удвоить. Символьный литерал представляет собой либо заключённый

в одинарные кавычки символ, либо символ, заданный своим кодом. Последнее представляет собой знак \$, после которого идёт целое число, являющееся кодом символа. Число это может быть двоичным, восьмеричным, шестнадцатеричным, или десятичным Внутреннее представление строковых и символьных литералов — в кодировке Unicode, а именно, в её четырёхбайтовом варианте. Текст программы — только в кодировке UTF-8. Строковый литерал может представлять собой как простой строковый литерал, так и чередование последовательности заданнных своими кодами символов и простых строковых литералов.

Если перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано u32. Если указано u16, то строковый литерал — в кодировке UTF-16. Наконец, если указано u8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

```
Задача 1.9. С помощью генератора лексических анализаторов Мяука для приводимого ниже
языка напишите и протестируйте лексический анализатор.
программа \to модуль ид {[использует списид] [предоставляет списид] тело модуля}
cnucud \rightarrow uд{,uд}
тело модуля \rightarrow {переменные | munы | функция | прототип | константы }
nepeменные \rightarrowперем cnucud:npocmeŭuee onpedeление <math>muna\{;cnucud:npocmeŭuee onpedeлeниe <math>muna\}
munu \rightarrow \mathbf{TИП} \, \mathbf{И}\mathbf{Д} = onpedenenue \, muna\{; \mathbf{И}\mathbf{Д} = onpedenenue \, muna\}
определение muna \rightarrow npocmeŭuee определение muna \mid cocmaвноe определение muna
простейшее определение типа \rightarrow элементарный тип\mid
                                     массив[[выр]{,[выр]}] простейшее определение типа
элементарный mun \rightarrow \{@\} (встроенный mun \mid \mu \mu)
6cmpoeнный mun \rightarrow \{большое|маленькое\} (цел|вещ|беззн|лог|симв|строка)|ничто|байт|цел8|
                      цел16|цел32|цел64|цел128|беззн8|беззн16|беззн32|беззн64|беззн128|
                      вещ32|вещ<math>64|вещ<math>128|лог<math>8|лог<math>16|лог<math>32|лог<math>64|симв<math>8|симв<math>16|симв<math>32|
                      строка8/строка16/строка32
cocmaвное определение muna \rightarrow onpederenue kopme \rightarrow ca|onpederenue указателя на функцию|
                                   определение алгебраического типа
определение кортежа \rightarrow (:элементарный mun\{,элементарный mun\}:)
определение указателя на функцию \rightarrow \phiункция сигнатура функции
cuгнатура функции \rightarrow (cnuc форм napam):элементарный mun
cnuc \phi opM napaM \rightarrow \phi opM napaM \{, \phi opM napaM \}
\phiорм парам \rightarrowзначение списид:простейшее определение типа
                ссылка списид:простейшее определение типа
                конст ссылка списид:простейшее определение типа
onpedenetue алгебраического muna \rightarrow часть алгебр muna\{.\}. часть алгебр muna\}
часть алгебр типа \rightarrow перечисление |структура
nеречисление →перечисление ид\{cnucud\}
cmpyкmypa \rightarrow ctpyкtypa ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud:простейшее определение типа
\phiункция \rightarrow \phiункция ид cигнатура \phiункции\{тело \deltaлока\}
npomomun \to \phiункция ид сигнатура функции;
константы \toконст определение константы\{,определение константы\}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow{переменные|типы|функция|прототип|константы|операторы}}
eup \rightarrow eup_0 [?eup_0:eup_0]
sup_1 \rightarrow sup_2 \{ (\&\&/!\&\&) sup_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4 \{ (<|>|<=|>=|=|!=) 6bip_4 \}
```

 $eup_4 \rightarrow eup_5 \{(I|^{\sim}I|^{\wedge}|^{\sim})eup_5\}$ 

```
\operatorname{Bup}_5 \to \operatorname{Bup}_6 \{ (\&/~\&/\ll/\gg/\gg) \operatorname{Bup}_6 \}
6blp_6 \rightarrow {^{\sim}}6blp_7
sup_7 \rightarrow sup_8 \{(+|-)sup_8\}
eup_8 \rightarrow eup_9 \{ (*//\%//.) eup_9 \}
\epsilon \omega p_9 \rightarrow \epsilon \omega p_{10} [(**/**.)\epsilon \omega p_9]
выp_{10} \to [+/-/\#] выp_{11}
eup_{11} \rightarrow eup_{12} [##eup_{12}]
6 bip_{12} \to \text{целое} | \text{вещественное} | \textbf{истина} | \textbf{ложь} | \text{строковое} | (6 bip) | uma | cocm знач
cocm знач \rightarrowзнач cmpykm|знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nors\{, statement{}
значение nоля \rightarrowид <- \epsilonыр
знач массива \to массив[:выр,элементарный тип:]{списвыр}
uмя \rightarrow [[:uмя\_моdуля::] [uмя\_muna::uмя\_части\_muna::]uд{([cnucвыp])|[cnucвыp]|@[.uд}
cnuceup \rightarrow eup\{,eup\}
имя модуля \rightarrowид
uмя muna \rightarrow uд
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop \rightarrow uu\kappa na | npuc ва uвания | nepexoda | ввода | вывода | условный | разбора
условный \toесли выр \mathbf{тo}{mело_блока}{uнес выр \mathbf{тo}{mело блока}}[иначе{mело блока}]всё
uuкла \rightarrow [<:uд:>] (для ид из uuran = uuran
                   пока выр{тело блока}|повторяй{тело блока}покуда выр|
                   \mathbf{mayk}{случай {,случай} [иначе{mело блока}]})
случай \to \mathbf{при} \ выр\{ mело \ блока \}
присваивания \to ums := выр
nepexoda \to \mathbf{выйди} [из ид] |\mathbf{npoдoлжu}| [ид] |\mathbf{возврат}| [выр]
660 \partial a \rightarrow \mathbf{ввод}(cnucum\ddot{e}H)
вывод a \rightarrowвывод(cnucsup)
cnucum\ddot{e}_{H} \rightarrow ums\{,ums\}
разбора \rightarrowразбор выр\{образец ->\{ тело блока\}\{,образец ->\{ тело блока\}\}\{ [ иначе\{ тело блока\}\}\}
образец \rightarrow []:имя модуля::]имя типа::имя части типа\{...\}
      Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
uenoe \rightarrow decsmuuhoe|восьмеричное|двоичное|шестнадиатеричное
\partial ecsmuчнoe \rightarrow \partial ec цифра('?\partial ec цифра)*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
\partial воичное \rightarrow 0(b|B)\partial воичн цифра('?\partial воичн цифра)*
uecmhaduamepuuhoe \rightarrow 0(x|X)uecmh uuppa('?uecmh uuppa)*
\partial ec \quad uu\phi pa \rightarrow 0/1/2/3/4/5/6/7/8/9
восьм иифра \rightarrow 0/1/2/3/4/5/6/7
\partialвоичн цифра \rightarrow 0/1
uecm + uu \phi pa \rightarrow 0 |1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F
вещественное \rightarrowцелая часть (дробная часть)?(символ порядка(+|-)?порядок)?
символ порядка \rightarrow E|e|S|s|D|d|Q|q
целая часть \rightarrow \partial e cятичное
дробная часть \rightarrow. десятичное
nopядoк \rightarrow decяmичноe
```

Если перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано и32. Если указано и16, то строковый литерал — в кодировке UTF-16. Наконец, если указано и8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

Задача 1.10. С помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор. программа  $\rightarrow$  module ид {[import cnucud] [export cnucud] тело модуля}  $cnucud \rightarrow uд{,uд}$ тело модуля  $\rightarrow$  {переменные | munы | функция | прототип | константы } nepeменные  $\rightarrow$  var cnucud:npocmeйшее onpedeneние muna  $\{;cnucud:$ npocmeйшее onpedeneние muna  $\}$  $munu \rightarrow type \, ud = onpedenehue \, muna \{; ud = onpedenehue \, muna\}$ onpederehue muna 
ightarrow npocmeŭwee onpederehue muna | cocmaвное onpederehue munaпростейшее определение типа  $\rightarrow$ элементарный тип $\mid$ array[[выр]{,[выр]}] простейшее определение типа элементарный  $mun \rightarrow \{@\}$  (встроенный  $mun \mid \mu \mu$ ) 6cmpoeнный  $mun \rightarrow \{long|short\} (int|float|uint|bool|char|string)|void|byte|int8 <math>t|int16$  t|int 32  $t | \text{int} 64 \ t | \text{int} 128 \ t | \text{uint} 8 \ t | \text{uint} 16 \ t | \text{uint} 32 \ t | \text{uint} 64 \ t |$ uint128 t|float32 t|float64 t|float128 t|bool8 t|bool16 t|bool32 t| $bool 64 \quad t | char 8 \quad t | char 16 \quad t | char 32 \quad t | string 8 \quad t | string 16 \quad t | string 32 \quad t$ cocmaвное определение  $muna \rightarrow onpederenue$   $kopme \rightarrow onpederenue$  kopmeопределение алгебраического типа определение кортежа  $\rightarrow$  (:элементарный  $mun\{$ ,элементарный  $mun\}$ :) определение указателя на функцию  $\rightarrow$ fn сигнатура функции сигнатура функции  $\rightarrow$  (спис форм парам):элементарный тип  $cnuc \not fopm napam \rightarrow \not fopm napam \{, \not fopm napam \}$ форм парам  $\rightarrow$  val списид:простейшее определение типа ref cnucud:npocmeйшее определение muna/ const ref списид:простейшее определение типа onpederenue алгебраического  $muna \rightarrow часть$  алгебр  $muna\{..., часть$  алгебр  $muna\}$ часть алгебр типа  $\rightarrow$ перечисление|структура nepeuceneue →**enum** ид $\{cnucud\}$  $cmpyкmypa \rightarrow struct$  ид $\{cnuc\ rpynn\ noneŭ\}$ cnuc групп полей  $\rightarrow$ группа полей $\{;$ группа полей $\}$ группа полей  $\rightarrow$  cnucud:простейшее определение типа функция  $\rightarrow$ **fn** ид сигнатура функции $\{$ тело блока $\}$  $npomomun \rightarrow \mathbf{fn}$  ид  $curhamypa\_ функции;$ константы  $\rightarrow$  const определение константы $\{$ , определение константы $\}$ определение константы  $\rightarrow$ ид:простейшее определение типа = выр mело блока  $\rightarrow$  { $nepemenhbe|munb||phyhkuux||npomomun||kohcmahmbe||onepamopb||}$  $eup \rightarrow eup_0$  [? $eup_0$ : $eup_0$ ]  $ewp_0 \rightarrow ewp_1\{(|||!|||^{^{-}}|!^{^{-}})ewp_1\}$  $ewp_1 \rightarrow ewp_2 \{ (\&\& | !\&\&) ewp_2 \}$  $6bip_2 \rightarrow \{!\}6bip_3$  $6bip_3 \rightarrow 6bip_4 \{ (<|>|<=|>=|=|!=) 6bip_4 \}$  $eup_4 \rightarrow eup_5 \{(I|~I|~|~~)eup_5\}$  $ebip_5 \rightarrow ebip_6 \{ (\&|^{\sim}\&| \ll |\gg|) \otimes bip_6 \}$  $6blp_6 \rightarrow {^{\sim}} 6blp_7$  $sup_7 \rightarrow sup_8 \{(+/-)sup_8\}$  $\varepsilon \omega p_8 \rightarrow \varepsilon \omega p_9 \{ (*//\%/.) \varepsilon \omega p_9 \}$  $\varepsilon \omega p_9 \rightarrow \varepsilon \omega p_{10} [(**/**.)\varepsilon \omega p_9]$  $6bip_{10} \rightarrow [+/-/\#] 6bip_{11}$ 

 $выp_{11} \rightarrow выp_{12}$  [## $выp_{12}$ ]

```
6 i i p_{12} \rightarrow целое/вещественное/true/false/строковое/(6 i i p)/имя/сост знач
cocm знач \rightarrowзнач cmpykm/знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nors\{, started{started}
значение поля \rightarrowид <- выр
знач массива \rightarrowarray[:выр,элементарный тип:]{списвыр}
uмя \rightarrow [[:uмя\_моdуля::] [uмя\_muna::uмя uacmu umuna::]uд{([cnucвыp])|[cnucвыp]|@|.uд}
cnuceup \rightarrow eup\{,eup\}
имя модуля \rightarrowид
uмя muna \rightarrow uд
uмя чacmu muna \rightarrowuд
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop 
ightarrow ukna | npucваивания | nepexoda | ввода | вывода | условный | разбора
условный \rightarrowif выр then\{mело блока\}{elif выр then\{mело блока\}\}[else\{mело блока\}]endif
uu\kappa na \rightarrow [\langle :u\pi :\rangle] (for u\pi : eup\{meno\ блока\}|for u\pi := eup,eup[,eup]\{meno\ блока\}|
          while выр{тело блока}|repeat{тело блока}until выр|
          spider{случай {,случай} [else{mело блока}]})
cлучай \rightarrowcase выр\{тело блока\}
присваивания \to ums := выр
nepexoda \rightarrow break [ ид] | continue [ид] | return [выр]
ввода \rightarrowread(списимён)
вывода \rightarrowprint(cnucвыр)
cnucum\ddot{e}_{H} \rightarrow ums\{,ums\}
разбора \rightarrow match выр{образец ->{mело блока}{,образец ->{mело блока}}[else{mело блока}]}
образец \rightarrow []:имя модуля::]имя типа::имя части типа\{..\}
   Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
uenoe \rightarrow decsmuuhoe|восьмеричное|deouuhoe|mecmhaduamepuuhoe|
\partial e c я m u ч ho e \rightarrow \partial e c \quad u u \phi p a ('? \partial e c \quad u u \phi p a)^*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
dвоичное \rightarrow 0(b|B)dвоичн цифра('?dвоичн цифра)*
uecmhaduamepuчhoe \rightarrow 0(x|X)uecmh uuфpa('?uecmh uuфpa)*
\partial ec \quad uu\phi pa \rightarrow 0|1|2|3|4|5|6|7|8|9
восьм иифра \rightarrow 0/1/2/3/4/5/6/7
\partialвоичн ии\phiра \rightarrow 0/1
uecm h uu fpa \rightarrow 0 |1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F
вещественное \rightarrowцелая часть(дробная часть)?(символ порядка(+|-)?порядок)?
символ порядка \rightarrow E|e|S|s|D|d|Q|q
целая часть \rightarrow \partial e cятичное
дробная часть \rightarrow.десятичное
nopядoк \rightarrow decятичноe
```

Eсли перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано u32. Eсли указано u16, то строковый литерал — в кодировке UTF-16. Наконец, если указано u8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

```
Задача 1.11. С помощью генератора лексических анализаторов Мяука для приводимого ниже
языка напишите и протестируйте лексический анализатор.
программа \to модуль ид {[использует списид] [предоставляет списид] тело модуля}
cnucud \rightarrow uд{,uд}
тело модуля \rightarrow {переменные | типы | функция | прототип | константы }
nepeменные \rightarrow перем cnucud: npocmeŭuee onpedeление muna\{; cnucud: npocmeŭuee onpedeление muna\}
munu \rightarrow \mathbf{TИП} \, \mathbf{ИД} = onpedenenue \, muna\{; \mathbf{ИД} = onpedenenue \, muna\}
onpedenehue muna 
ightarrow npocmeŭwee onpedenehue muna | cocmaвное onpedenehue muna
простейшее определение типа \rightarrow элементарный тип\mid
                                       массив[[выр]{,[выр]}] простейшее определение типа
элементарный mun \rightarrow \{@\} (встроенный mun \mid \mu \mu)
6cmpoehhhiй mun \to \{большое|маленькое\} (цел|вещ|беззн|лог|симв|строка)|ничто|байт|цел8|
                       цел16|цел32|цел64|цел128|беззн8|беззн16|беззн32|беззн64|беззн128|
                       вещ32|вещ64|вещ128|лог8|лог16|лог32|лог64|симв8|симв16|симв32|
                       строка8/строка16/строка32
cocmaвноe определение muna \rightarrow onpedenenue kopme \rightarrow color pedenenue указателя на функцию
                                     определение алгебраического типа
определение кортежа \rightarrow (:элементарный mun\{,элементарный mun\}:)
определение _ указателя _ на _ функцию 
ightarrow\phiункция сигнатура _ функции
cuгнатура функции \rightarrow (cnuc форм napam):элементарный mun
cnuc \ \phiopm \ napam \rightarrow \phiopm \ napam \{, \phiopm \ napam \}
форм парам \rightarrowзначение списид:простейшее определение типа
                 ссылка списид:простейшее определение типа
                 конст ссылка списид:простейшее определение типа
определение алгебраического типа \rightarrow часть алгебр типа\{...часть алгебр типа\}
часть алгебр типа \rightarrowперечисление|структура
neperuc_nehue \rightarrow neperuc_nehue ид\{cnucud\}
cmpyкmypa \rightarrow ctpyкtypa ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud: простейшее определение типа
функция \to \phiункция ид сигнатура функции\{тело блока\}
npomomun \rightarrow \phiункция ид сигнатура функции;
константы \rightarrowконст определение константы\{, определение константы\}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow {переменные | типы | функция | прототип | константы | операторы}
ewp \rightarrow ewp_0 [?ewp_0:ewp_0]
eup_0 \rightarrow eup_1\{(|||!|||^{^}|!^{^})eup_1\}
6bip_1 \rightarrow 6bip_2 \{ (\&\& | !\&\&) 6bip_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4 \{ (<|>|<=|>=|=|!=) 6bip_4 \}
eup_4 \rightarrow eup_5 \{(I|^{\sim}I|^{\wedge}|^{\sim})eup_5\}
eup_5 \rightarrow eup_6 \{ (\&|^{\sim}\&| \ll |\gg|) \otimes up_6 \}
eup_6 \rightarrow {^{\sim}}eup_7
ewp_7 \rightarrow ewp_8\{(+|-)ewp_8\}
выp_8 \rightarrow выp_9 \{ (*//%//.) выp_9 \}
\epsilon \omega p_9 \rightarrow \epsilon \omega p_{10} [(**|**.)\epsilon \omega p_9]
ewp_{10} \to [+/-/\#]ewp_{11}
eup_{11} \rightarrow eup_{12} [##eup_{12}]
6 wp_{12} \to \text{целое} | \text{вещественное} | \textbf{истина} | \textbf{ложь} | \text{строковое} | (6 wp) | ums | cocm знач
cocm знач \rightarrowзнач cmpykm|знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nors\{, started{started}
значение nоля \to ид < - выр
знач массива \rightarrow массив[:выр,элементарный mun:]{cnucвыр}
uмя \rightarrow [[:uмя\_модуля::][uмв\_muna::uмя части типа::]ид{([cnucвыр])|[cnucвыр]]@].ид}
cnuceup \rightarrow eup\{,eup\}
```

```
имя модуля \rightarrowид
uмя muna 
ightarrow uд
uмя uасmи mиnа \rightarrowид
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop \rightarrow uu\kappa \wedge a | npucaauahun | nepexoda | aboda | buboda | yc \wedge obhuŭ | pastopa
условный \rightarrowесли выр \mathbf{ro}\{тело \{блока\}\{инес выр \mathbf{ro}\{тело \{блока\}\}\{иначе\{тело \{блока\}\}\}всё
uu\kappa \wedge a \to [<:u\pi:>] (для ид из выр\{me\wedge o \text{ блока}\}[ завершение\{me\wedge o \text{ блока}\}]/
                         для ид := выр, выр[, выр] \{ mело блока \} [ завершение\{ mело блока \} ] [
                          пока выр{тело блока}|
                         \max \{c_{ny} \in \{
cлучай \rightarrow \mathbf{при} \ выр\{mело \ блока\}
присваивания \to ums := выр
nepexoda \to \mathbf{выйди} [из ид] |\mathbf{продолжи} [ид] |\mathbf{возврат}| [eыp]
ввода \rightarrow \mathbf{ввод}(cnucum\ddot{e}H)
выводa \rightarrowвывод(cnuceup)
cnucum\ddot{e}н \rightarrowumя\{,umя\}
разбора \rightarrowразбор выр\{образец ->\{ тело блока\}\{,образец ->\{ тело блока\}\}\{ [ иначе\{ тело блока\}\}\}
образец \rightarrow [:имя модуля::]имя типа::имя части типа\{...\}
         Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
yenoe \rightarrow decsmuuhoe|восьмеричное|deouvhoe|mecmhadyamepuuhoe
\partial e c s m u ч ho e \rightarrow \partial e c \quad u u \phi p a ('? \partial e c \quad u u \phi p a)^*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
\partial воичное \rightarrow 0(b|B)\partial воичн цифра ('?\partial воичн цифра)*
uecmhaduamepuчhoe \rightarrow 0(x|X)uecmh uuфpa('?uecmh uuфpa)*
\partial ec \quad uu\phi pa \rightarrow 0/1/2/3/4/5/6/7/8/9
восьм цифра \to 0|1|2|3|4|5|6|7
\partialвоичн цифра \rightarrow 0/1
uecm h uu fpa \rightarrow 0 |1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F
вещественное \to целая_ часть (дробная_ часть)?(символ порядка (+/-)?порядок)?
символ порядка \rightarrow E|e|S|s|D|d|Q|q
целая часть \rightarrow \partial e cятичное
дробная часть \rightarrow. десятичное
nopядoк \rightarrow decятичноe
```

Eсли перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано u32. Eсли указано u16, то строковый литерал — в кодировке UTF-16. Наконец, если указано u8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

**Задача 1.12.** С помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор.

```
программа \rightarrow module ид {[import cnucud] [export cnucud] тело_модуля} списид \rightarrowид{,ид}
```

```
тело модуля \rightarrow {переменные | munы | функция | прототип | константы }
nepeменные \rightarrow var списид:npocmeйшее onpeделение muna {;cnucud:npocmeйшее onpeделение muna}
munu \rightarrow type \ ud = onpedenenue \ muna \{; ud = onpedenenue \ muna \}
onpedeление muna 	o npocme
uuee onpedеление muna|cocmaвное onpedеление muna|cocmasное onpedеление onpedелени
простейшее определение типа \rightarrow элементарный тип\mid
                                                           array[[выр]{,[выр]}] простейшее определение типа
элементарный mun \rightarrow \{@\} (встроенный mun \mid \mu \mu)
gcmpoeнный mun \rightarrow \{long|short\} (int|float|uint|bool|char|string)|void|byte|int8 t|int16 t|
                                   int32 t|int64 t|int128 t|uint8 t|uint16 t|uint32 t|uint64 t|
                                   uint128 t|float32 t|float64 t|float128 t|bool8 t|bool16 t|bool32 t|
                                   bool64 t/char8 t/char16 t/char32 t/string8 t/string16 t/string32 t
cocmaвное определение muna \rightarrow onpederenue кортежа|onpederenue указателя на функцию|
                                                        определение алгебраического muna
определение кортежа \rightarrow (:элементарный mun\{,элементарный mun\}:)
определение указателя на функцию \rightarrowfn сигнатура функции
сигнатура функции \rightarrow (спис форм парам):элементарный тип
cnuc \not fopm napam \rightarrow \not fopm napam \{, \not fopm napam \}
форм парам \rightarrow val списид:простейшее определение типа
                          ref cnucud:простейшее определение типа
                          const ref списид:простейшее определение типа
определение алгебраического типа \rightarrow часть алгебр типа\{...часть алгебр типа\}
часть алгебр типа \rightarrowперечисление|структура
nepeuucлehue →enum ид\{cnucud\}
cmpyкmypa \rightarrow struct ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud:простейшее определение типа
\phiункция \rightarrowfn ид сигнатура \phiункции\{тело \phiока\}
npomomun \rightarrow \mathbf{fn} ид curhamypa функции;
константы \rightarrowconst определение константы \{, определение константы\}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow {переменные | типы | функция | прототип | константы | операторы}
ewp \rightarrow ewp_0 [?ewp_0:ewp_0]
eup_0 \rightarrow eup_1\{(|||!|||^{^{-}}|!^{^{-}})eup_1\}
ewp_1 \rightarrow ewp_2 \{ (\&\& | !\&\&) ewp_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4 \{ (< > | < = | > = | = | ! = ) 6bip_4 \}
eup_4 \rightarrow eup_5 \{(I|^{\sim}I|^{\wedge}|^{\sim})eup_5\}
eup_5 \rightarrow eup_6 \{ (a/~a/\ll \gg ) up_6 \}
eup_6 \rightarrow {^{\sim}}eup_7
eup_7 \rightarrow eup_8\{(+|-)eup_8\}
sup_8 \to sup_9 \{ (*//\%/.) sup_9 \}
\varepsilon \omega p_9 \rightarrow \varepsilon \omega p_{10} [(**|**.)\varepsilon \omega p_9]
ewp_{10} \to [+/-/\#] ewp_{11}
выp_{11} \rightarrow выp_{12} [##выp_{12}]
выp_{12} \rightarrow целое|вещественное|true|false|строковое|(выp)|имя|сост знач
cocm знач \rightarrowзнач cmpykm/знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nons\{, started ones \}\}
значение nоля \rightarrowид <- \epsilonыр
знач массива \rightarrowarray[:выр,элементарный mun:]{cnucвыр}
uмя \rightarrow [[:uмя модуля::][uмя типа::uмя части типа::]ид{([списвыр])|[списвыр]]@[.ид}
cnuceup \rightarrow eup\{,eup\}
uмя модуля \rightarrowид
uмя muna \rightarrow uд
uмя uacmu muna \rightarrowuд
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop \rightarrow uu\kappa \wedge a | npucaauahun | nepexoda | aboda | buboda | yc \wedge obhuŭ | pastopa
условный \rightarrowif выр then{meло блока}{elif выр then{meло блока}}[else{meло блока}]endif
```

```
uu\kappa \wedge a \rightarrow [\langle :u\pi : \rangle] (for u\pi : eup\{me \wedge o \wedge o \wedge a\}[else\{me \wedge o \wedge o \wedge a\}]]
           for ид := eup,eup[,eup]\{meno\ блока\}[else\{meno\ блока\}]|
           while выр{тело блока}|repeat{тело блока}until выр
           spider{случай {,случай} [else{meло блока}]})
cлучай \rightarrowcase выр\{тело блока\}
npuceaueahuя \to umя := eыp
nepexoda \rightarrow break [ид]/continue [ид]/return [еыр]
ввода \rightarrowread(списимён)
вывода \rightarrowprint(cnucвыр)
cnucum\ddot{e}_{H} \rightarrow ums\{,ums\}
разбора \rightarrow match выр{образец ->{mело блока}{,образец ->{mело блока}}[else{mело блока}]}
образец \rightarrow []:имя модуля::]имя типа::имя части типа\{...\}
   Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
иелое \rightarrow decsmuvhoe|восьмеричное|deouvhoe|шестнадиатеричное|
\partial e c я m u ч ho e \rightarrow \partial e c \quad u u \phi p a ('? \partial e c \quad u u \phi p a)^*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
\partial Bouчнoe \rightarrow O(b|B)\partial Bouчн и цифра ('? \partial Bouчн и цифра)*
wecmhaduamepuчное \rightarrow 0(x|X)wecmh uuфpa('?wecmh uuфpa)*
\partial ec \quad uu\phi pa \rightarrow 0|1|2|3|4|5|6|7|8|9
восьм цифра \to 0|1|2|3|4|5|6|7
\partialвоичн цифра \rightarrow 0/1
uecm + uu\phi pa \rightarrow 0|1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F
вещественное \to целая_ часть (дробная часть)?(символ порядка (+|-)?порядок)?
символ порядка \rightarrow E|e|S|s|D|d|Q|q
целая часть \rightarrow \partial e cятичное
дробная часть \rightarrow. десятичное
nopядок \rightarrow десятичное
```

Если перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано u32. Если указано u16, то строковый литерал — в кодировке UTF-16. Наконец, если указано u8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

**Задача 1.13.** С помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор.

```
программа \rightarrow модуль ид {[использует списид] [предоставляет списид] тело_модуля} списид \rightarrowид{,ид} тело_модуля \rightarrow{переменные|типы|функция|прототип|константы} переменные \rightarrowперем списид:простейшее_определение_типа{;списид:простейшее_определение_типа} типы \rightarrowтип ид = определение_типа{;ид = определение_типа} определение_типа \rightarrowпростейшее_определение_типа | составное_определение_типа простейшее_определение_типа \rightarrowэлементарный_тип| массив[[выр]{,[выр]}]простейшее_определение_типа
```

```
элементарный mun \rightarrow \{@\} (встроенный mun | \mu д)
6cmpoehhhiй mun \to \{большое|маленькое\} (цел|вещ|беззн|лог|симв|строка)|ничто|байт|цел8|
                                       _{\rm цел}16|_{\rm цел}32|_{\rm цел}64|_{\rm цел}128|_{\rm беззh}8|_{\rm беззh}16|_{\rm беззh}32|_{\rm беззh}64|_{\rm беззh}128|_{\rm fessh}8|_{\rm fessh}8|_
                                       вещ32|вещ64|вещ128|лог8|лог16|лог32|лог64|симв8|симв16|симв32|
                                       строка8/строка16/строка32
cocmaвное определение muna \rightarrow onpedenenue kopme \rightarrow color pedenenue указателя на функцию
                                                               определение алгебраического типа
определение кортежа \to (:элементарный mun\{,элементарный mun\}:)
определение _ указателя _ на _ функцию 
ightarrow\phiункция сигнатура _ функции
сигнатура функции \rightarrow (спис форм парам):элементарный тип
cnuc \not fopm napam \rightarrow \not fopm napam \{, \not fopm napam \}
форм парам \rightarrowзначение списид:простейшее определение типа
                             ссылка списид:простейшее определение типа
                             конст ссылка списид:простейшее определение типа
onpedeеление алгебраического muna 	o часть алгебр muna \{...\}иасть алгебр muna \}
часть алгебр типа \rightarrowперечисление|структура
neperuc_nehue \rightarrow neperuc_nehue ид\{cnucud\}
cmpyкmypa \rightarrow ctpyкtypa ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud:простейшее определение типа
\phiункция \to \phiункция ид сигнатура \phiункции\{тело \phiока\}
npomomun \rightarrow \phiункция ид сигнатура функции;
константы \toконст определение константы\{,определение константы\}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow{переменные|типы|функция|прототип|константы|операторы}
ewp \rightarrow ewp_0 [?ewp_0:ewp_0]
eup_0 \rightarrow eup_1\{(|||!|||^{^{-}}|!^{^{-}})eup_1\}
eup_1 \rightarrow eup_2 \{ (\&\& | !\&\&) eup_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4 \{ (< > | < = | > = | = | ! = ) 6bip_4 \}
eup_4 \rightarrow eup_5 \{(I|^{\sim}I|^{\wedge}|^{\sim})eup_5\}
eup_5 \rightarrow eup_6 \{ (\&| \tilde{\&}| \ll | \gg | \gg) eup_6 \}
6bip_6 \rightarrow {^{\sim}} 6bip_7
eup_7 \rightarrow eup_8\{(+|-)eup_8\}
6blp_8 \rightarrow 6blp_9 \{(*//\%/.)6blp_9\}
\varepsilon \omega p_9 \rightarrow \varepsilon \omega p_{10} [(**|**.)\varepsilon \omega p_9]
выp_{10} \to [+/-/\#] выp_{11}
выp_{11} \rightarrow выp_{12} [##выp_{12}]
6 bip_{12} \to \text{целое} | \text{вещественное} | \textbf{истина} | \textbf{ложь} | \text{строковое} | (6 bip) | uma | cocm знач
cocm знач \rightarrowзнач cmpykm/знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nons\{.значение nons\}\}
значение nоля \to ид < - выр
знач массива \rightarrow массив[:выр,элементарный mun:]{cnucвыр}
uмя \rightarrow [[:uмя модуля::][uмя типа::uмя части типа::]ид{([списвыр])|[списвыр]]@[.ид}
cnuceup \rightarrow eup\{,eup\}
uмя модуля \rightarrowид
uмя muna \rightarrow uд
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop \rightarrow uu\kappa na | npuc ва uвания | nepexoda | ввода | вывода | условный | разбора
условный \rightarrowесли выр \mathbf{To} \{ тело блока\} \{ \mathbf{unec} выр \mathbf{To} \{ тело блока\}\} \{ \mathbf{unaue} \{ тело блока\}\} \} \mathbf{Ec}
пока выр{тело блока}|повторяй{тело блока}покуда выр
                паук{случай {,случай} [иначе{тело_блока}]})
случай \toпри выр\{тело блока\}
npuceaueahuя 	o umя \ (:=|+:=|-:=|*:=|/:=|/.:=|**:=|**.:=|\%:=|\&:=|^*\&:=|\ll:=|\gg:=|I:=|^*I:=|^*:=|^*:=|
                                       ||:=|!||:=|^^:=|!^^:=|&&:=|!&&:=|>>>:=) выр
```

```
nepexoda \to \mathbf{выйди} [из ид] |\mathbf{продолжи} [ид] |\mathbf{возврат}| [eыp]
ввода \rightarrow \mathbf{ввод}(списимён)
вывод a \rightarrowвывод(cnuceup)
cnucum\ddot{e}н \rightarrowumя\{,umя\}
разбора \rightarrow разбор выр\{образец ->\{ тело блока\}\{,образец ->\{ тело блока\}\}\{ [ иначе\{ тело блока\}\}\}
образеu \to []:uмя модуля::]uмя muna::umя uacmu muna\{...\}
   Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
yenoe \rightarrow decsmuuhoe|bocьмерииное|deouuhoe|mecmhadyamepuuhoe|
\partial e c s m u u ho e \rightarrow \partial e c u u \phi p a ('? \partial e c u u \phi p a)^*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
\partial воичное \rightarrow 0(b|B)\partial воичн цифра ('?\partial воичн цифра)*
uecmhaduamepuчноe \rightarrow 0(x|X)uecmh uuфpa('?uecmh uuфpa)*
\partial ec \quad uu \phi pa \rightarrow 0/1/2/3/4/5/6/7/8/9
восьм цифра \to 0|1|2|3|4|5|6|7
\partialвоичн цифра \rightarrow 0/1
uecm h uu fpa \rightarrow 0 |1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F
вещественное \rightarrowцелая часть(дробная часть)?(символ порядка(+|-)?порядок)?
символ порядка \rightarrow E|e|S|s|D|d|Q|q
целая часть \rightarrow \partial e cятичное
дробная часть \rightarrow.десятичное
nopядoк \rightarrow decятичноe
```

Если перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано u32. Если указано u16, то строковый литерал — в кодировке UTF-16. Наконец, если указано u8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

**Задача 1.14.** С помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор.

```
программа → module ид {[import cnucud] [export cnucud] тело_модуля}

списид →ид{,ид}

тело_модуля →{переменные|munы|функция|прототип|константы}

переменные →var cnucud:простейшее_определение_muna{;cnucud:простейшее_определение_muna}

типы →type ид = oпределение_muna{;ид = oпределение_muna}

определение_muna →простейшее_oпределение_muna|cocтавное_oпределение_muna

простейшее_onpedеление_muna →элементарный_mun|

array[[выр]{,[выр]}]простейшее_oпределение_muna

элементарный_mun →{@}(встроенный_mun|ид)

встроенный_mun →{long|short}(int|float|uint|bool|char|string)|void|byte|int8_t|int16_t|

int32_t|int64_t|int128_t|uint8_t|uint16_t|uint32_t|uint64_t|

uint128_t|float32_t|float64_t|float128_t|bool8_t|bool16_t|bool32_t|

bool64_t|char8_t|char16_t|char32_t|string8_t|string16_t|string32_t|
```

```
cocmaвнoe\_onpederenue\_muna 	oonpederenue\_кортежа | onpederenue\_указателя на функцию|
                                      определение алгебраического muna
определение кортежа \rightarrow (:элементарный mun\{,элементарный mun\}:)
определение указателя на функцию \rightarrowfn сигнатура функции
curhamypa функции \rightarrow (cnuc форм napam):элементарный mun
cnuc \ \phiopm \ napam \rightarrow \phiopm \ napam \{, \phiopm \ napam \}
форм парам \rightarrow val списид:простейшее определение типа
                 ref cnucud:простейшее определение типа
                  const ref cnucud:npocmeйшее_onpedeление_muna
определение алгебраического типа\rightarrowчасть алгебр типа\{...\}часть алгебр типа\}
часть алгебр типа \rightarrowперечисление|структура
nepeuucлehue →enum ид\{cnucud\}
cmpyкmypa \rightarrow struct ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud: простейшее определение типа
функция \rightarrowfn ид сигнатура функции{тело блока}
npomomun \rightarrow \mathbf{fn} ид curhamypa функции;
константы \rightarrowconst определение константы \{, определение константы\}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow{переменные|типы|функция|прототип|константы|операторы}
ewp \rightarrow ewp_0 [?ewp_0:ewp_0]
eup_1 \rightarrow eup_2 \{ (\&\& | \&\&) eup_2 \}
eup_2 \rightarrow \{!\}eup_3
6bip_3 \rightarrow 6bip_4 \{ (<|>|<=|>=|=|!=) 6bip_4 \}
eup_4 \rightarrow eup_5 \{(I|^{\sim}I|^{\wedge}|^{\sim})eup_5\}
ebip_5 \rightarrow ebip_6 \{ (\&| \tilde{k}| \ll | \gg | \gg) ebip_6 \}
6blp_6 \rightarrow {^{\sim}}6blp_7
eup_7 \rightarrow eup_8\{(+|-)eup_8\}
eup_8 \to eup_9 \{ (*//\%/.)eup_9 \}
\epsilon \omega p_9 \rightarrow \epsilon \omega p_{10} [(**/**.)\epsilon \omega p_9]
ewp_{10} \rightarrow [+/-/\#]ewp_{11}
eup_{11} \rightarrow eup_{12} [##eup_{12}]
выp_{12} \rightarrowцелое|вещественное|true|false|строковое|(выp)|имя|сост знач
cocm знач \rightarrowзнач cmpykm/знач массива
знач структ \to [!:имя модуля::]имя типа::имя части типа\{значение поля\{,значение поля\}\}
значение nоля \rightarrowид <- \epsilonыр
знач массива \rightarrowarray[:выр,элементарный mun:]{cnucвыр}
uмя \rightarrow [[:uмя модуля::][uмя типа::uмя части типа::]ид{([списвыр])|[списвыр]]@[.ид}
cnuceup \rightarrow eup\{,eup\}
uмя модуля \rightarrowид
uмя muna \rightarrow uд
uмя uacmu muna \rightarrowuд
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop 
ightarrow ukna | npucваивания | nepexoda | ввода | вывода | условный | разбора
условный \rightarrowif выр then\{mело блока\}\{elif выр then\{mело блока\}\}\{else\{mело блока\}\}endif
uu\kappa na \rightarrow [\langle :u\pi :\rangle] (for u\pi : eup\{meno\ блока\}|for u\pi := eup,eup[,eup]\{meno\ блока\}|
          while выр{тело блока}|repeat{тело блока}until выр
          spider{случай {,случай} [else{mело блока}]})
cлучай \rightarrowcase выр\{тело блока\}
npucваивания \toимя (:=|+:=|-:=|*:=|/:=|/.:=|**:=|**:=|%:=|^{\sim}:=|=|-:=|^{\sim}:=|
                       ||:=|!||:=|^^:=|&&:=|!&&:=|>>>:=) 6blp
nepexoda \rightarrow break [ид]/continue [ид]/return [выр]
ввода \rightarrowread(списимён)
вывода \rightarrowprint(cnucвыр)
cnucum\ddot{e}_{H} \rightarrow ums\{,ums\}
разбора \rightarrow match выр{образец ->{mело блока}{,образец ->{mело блока}}[else{mело блока}]}
```

```
образец \rightarrow [|:имя_модуля::] имя_типа::имя_части_типа {..} 
Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие регулярные определения):
```

```
уверов — десятичное | восьмеричное | двоичное | шестнадцатеричное десятичное — дес_цифра ('?дес_цифра)* восьмеричное \rightarrow 0овосьм_цифра ('?восьм_цифра)* двоичное \rightarrow 0(b|B) двоичн_цифра ('?двоичн_цифра)* шестнадцатеричное \rightarrow 0(x|X) шестн_цифра ('?шестн_цифра)* дес_цифра \rightarrow 0|1|2|3|4|5|6|7|8|9 восьм_цифра \rightarrow 0|1|2|3|4|5|6|7 двоичн_цифра \rightarrow 0|1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F вещественное \rightarrow целая_часть (дробная_часть)?(символ_порядка (+|-)?порядок)? символ_порядка \rightarrow E|e|S|s|D|d|Q|q целая_часть \rightarrow десятичное дробная_часть \rightarrow десятичное порядок \rightarrow десятичное
```

Если перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано u32. Если указано u16, то строковый литерал — в кодировке UTF-16. Наконец, если указано u8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

```
Задача 1.15. С помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор.
```

```
программа \to модуль ид {[использует списид] [предоставляет списид] тело модуля}
cnucud \rightarrow uд{,uд}
тело модуля \rightarrow {переменные | типы | функция | прототип | константы }
nepeмeнныe \rightarrow \mathbf{nepem} cnucud:npocmeŭwee onpedenetue <math>muna\{;cnucud:npocmeŭwee onpedenetue <math>muna\}
munu \rightarrow \mathbf{TИП} \ \mathbf{ИД} = onpederenue \ muna\{; \mathbf{ИД} = onpederenue \ muna\}
onpedenehue muna 
ightarrow npocmeŭwee onpedenehue muna | cocmaвное onpedenehue muna
npocmeйшее onpedenenue muna 	o 	 элементарный mun|
                                      массив[[выр]{,[выр]}] простейшее определение типа
элементарный mun \rightarrow \{@\} (встроенный mun \mid \mu \mu)
gcmpoehhhii mun 	o \{большое|маленькое\} (цел|вещ|беззн|лог|симв|строка)|ничто|байт|цел|8|
                       | \text{цел} 16 | \text{цел} 32 | \text{цел} 64 | \text{цел} 128 | \text{беззн} 8 | \text{беззн} 16 | \text{беззн} 32 | \text{беззн} 64 | \text{беззн} 128 |
                       вещ32|вещ64|вещ128|лог8|лог16|лог32|лог64|симв8|симв16|симв32|
                       строка8/строка16/строка32
cocmaвное определение muna \rightarrow onpederenue кортежа|onpederenue указателя на функцию|
                                    определение алгебраического типа
определение кортежа \rightarrow (:элементарный mun\{.элементарный mun\}:)
определение _ указателя _ на _ функцию 
ightarrow\phiункция сигнатура _ функции
cuгнатура функции \rightarrow (cnuc форм napam):элементарный mun
cnuc форм napaM \rightarrow \phiорм napaM \{, \phiорм napaM \}
```

```
форм парам \rightarrowзначение списид:простейшее определение типа
                   ссылка списид:простейшее определение типа
                   конст ссылка списид:простейшее определение типа
определение алгебраического типа \rightarrow часть алгебр типа\{...часть алгебр типа\}
часть алгебр типа \rightarrow перечисление |структура
nеречисление \rightarrow перечисление ид{cnucud}
cmpyкmypa \rightarrow ctpyкtypa ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа\_полей \to cnucud: простейшее\_определение\_типа
\phiункция \to \phiункция ид сигнатура \phiункции\{тело \phiока\}
npomomun \rightarrow \phiункция ид сигнатура функции;
константы \toконст определение константы\{,определение константы\}
определение константы \rightarrowид:простейшее определение типа = выр
тело блока \rightarrow{переменные|типы|функция|прототип|константы|операторы}}
выр \rightarrow выр_0 [?выp_0:выp_0]
esip_0 \rightarrow esip_1 \{(II|!II|^{\hat{}}|I|^{\hat{}}) esip_1\}
eup_1 \rightarrow eup_2 \{ (\&\&/!\&\&) eup_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4 \{ (< > | < = | > = | = | ! = ) 6bip_4 \}
eup_4 \rightarrow eup_5 \{(I|^{\sim}I|^{\wedge}|^{\sim})eup_5\}
ebip_5 \rightarrow ebip_6 \{ (\&| \tilde{k}| \ll | \gg | \gg) ebip_6 \}
eup_6 \rightarrow {^{\sim}}eup_7
eup_7 \rightarrow eup_8\{(+|-)eup_8\}
sup_8 \to sup_9 \{ (*//\%/.) sup_9 \}
\varepsilon \omega p_9 \rightarrow \varepsilon \omega p_{10} [(**|**.)\varepsilon \omega p_9]
6bip_{10} \rightarrow [+/-/\#] 6bip_{11}
выp_{11} \rightarrow выp_{12} [##выp_{12}]
6 wp_{12} \to \text{целое} | \text{вещественное} | \textbf{истина} | \textbf{ложь} | \text{строковое} | (6 wp) | ums | cocm знач
cocm знач \rightarrowзнач cmpykm/знач массива
знач cmpykm \to []:ums модуля::]ums muna::ums части muna\{значение nons\{, started ones \}\}
значение nоля \rightarrowид <- eыр
знач массива \rightarrow массив[:выр,элементарный тип:]{cnucвыр}
uмя \rightarrow [[:uмя модуля::][uмя типа::uмя части типа::]ид{([списвыр])|[списвыр]]@[.ид}
cnuceup \rightarrow eup\{.eup\}
uмя модуля \rightarrowид
uмя muna \rightarrow uд
uмя u acmu muna \rightarrowuд
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop 
ightarrow ukra | npucваивания | nepexoda | ввода | вывода | условный | разбора
условный \rightarrowесли выр то\{тело блока\}\{инес выр то\{тело блока\}\}\{[иначе\{тело блока\}\}\}всё
uu\kappa \wedge a \to [<:u\pi:>] (для ид из выр\{me \wedge o \text{ блока}\}[завершение\{me \wedge o \text{ блока}\}]
          для ид := выр, выр[, выр]\{mело блока\}[ завершение\{mело блока\}][
          пока выр{тело блока}|повторяй{тело блока}покуда выр
          \mathbf{паук} \{ cлучай \{ , cлучай \} [ \mathbf{иначe} \{ meло блока \} ] \} )
случай \to \mathbf{при} \ выр\{ mело \ блока \}
npuceaueahuя \to ums (:=|+:=|-:=|*:=|/:=|/.:=|**:=|**.:=|%:=|&:=|^&:=|\ll:=|\ll:=|>:=||:=|^-|:=|^-:=|^-:=|
                         ||:=|!||:=|^^:=|!^^:=|&&:=|!&&:=|>>>:=) выр
nepexoda \to \mathbf{выйди} [из ид] |\mathbf{продолжи} [ид] |\mathbf{возврат}| [eыp]
вво∂а \rightarrowввод(списимён)
вывод a \rightarrowвывод(cnucsup)
cnucum\ddot{e}н 
ightarrow ums\{,ums\}
разбора \toразбор выр\{образец ->\{ тело блока\}\{,образец ->\{ тело блока\}\}\{ [ иначе\{ тело блока\}\}\}
образец \rightarrow []:имя модуля::]имя типа::имя части типа\{..\}
   Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
uenoe \rightarrow decsmuuhoe|восьмеричное|deouuhoe|mecmhaduamepuuhoe
\partial e c s m u u ho e \rightarrow \partial e c u u \phi p a ('? \partial e c u u \phi p a)^*
```

```
восьмеричное \to 0овосьм_ цифра ('?восьм_ цифра)* двоичное \to 0(b|B) двоичн_ цифра ('?двоичн_ цифра)* шестнадцатеричное \to 0(x|X) шестн_ цифра ('?шестн_ цифра)* двосьм_ цифра \to 0|1|2|3|4|5|6|7|8|9 восьм_ цифра \to 0|1|2|3|4|5|6|7 двоичн_ цифра \to 0|1|2|3|4|5|6|7|8|9|a|b|c|d|e|f|A|B|C|D|E|F вещественное \to целая_ часть (дробная_ часть)?(символ_ порядка (+|-)?порядок)? символ_ порядка \to \to десятичное дробная_ часть \to десятичное порядок \to десятичное
```

Eсли перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано u32. Eсли указано u16, то строковый литерал — в кодировке UTF-16. Наконец, если указано u8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.

```
Задача 1.16. C помощью генератора лексических анализаторов Мяука для приводимого ниже языка напишите и протестируйте лексический анализатор.

программа \rightarrow module ид {[import cnucud] [export cnucud] meло_модуля} 

списид \rightarrowид{,ид} 

тело_модуля \rightarrow{переменные|munы|функция|npomomun|константы} 

переменные \rightarrow var cnucud:npocmeйшее_onpedenenue_muna{;cnucud:npocmeйшее_onpedenenue_muna} 

типы \rightarrow type ид = onpedenenue_muna{;ид = onpedenenue_muna} 

определение_muna \rightarrow npocmeйшее_onpedenenue_muna|cocmaвное_onpedenenue_muna 

простейшее_onpedenenue_muna \rightarrow элементарный_mun| 

аггау[[выр]{,[выр]}] простейшее_onpedenenue_muna 

элементарный_mun \rightarrow{@} (встроенный_mun|ид) 

встроенный mun \rightarrow{long|short} (int|float|uint|bool|char|string)|void|byte|int8 t|int16 t|
```

```
\begin{array}{c} \textbf{uint} 128\_t | \textbf{float} 32\_t | \textbf{float} 64\_t | \textbf{float} 128\_t | \textbf{bool} 8\_t | \textbf{bool} 16\_t | \textbf{bool} 32\_t | \\ \textbf{bool} 64\_t | \textbf{char} 8\_t | \textbf{char} 16\_t | \textbf{char} 32\_t | \textbf{string} 8\_t | \textbf{string} 16\_t | \textbf{string} 32\_t \\ \textbf{cocmashoe\_onpedenehue\_muna} \rightarrow \textbf{onpedenehue\_kopmeska} | \textbf{onpedenehue\_ykasamens\_ha\_fyhkuuo} | \\ \textbf{onpedenehue\_anzeбpauчeckoro\_muna} \end{array}
```

int32 t|int64 t|int128 t|uint8 t|uint16 t|uint32 t|uint64 t|

```
определение_кортежса → (:элементарный_тип {,элементарный_тип }:)
определение_указателя_на_функцию → fn сигнатура_функции
сигнатура_функции → (спис_форм_парам):элементарный_тип
спис_форм_парам → форм_парам {,форм_парам}
форм_парам → val списид:простейшее_определение_типа |
ref списид:простейшее_определение_типа |
const ref списид:простейшее определение типа
```

onpedeneнue\_ алгебраического\_  $muna \to uacmb\_ a$ лгебр\_  $muna \{.|.uacmb\_ a$ лгебр\_  $muna \}$   $uacmb\_ a$ лгебр\_  $muna \to ne$  peuucne ue  $uacmb\_ a$ лгебр\_  $uacmb\_$ 

```
nepeucenehue →enum ид\{cnucud\}
cmpyкmypa \rightarrow struct ид\{cnuc\ rpynn\ noneŭ\}
cnuc групп полей \rightarrowгруппа полей\{;группа полей\}
группа полей \rightarrow cnucud:простейшее определение типа
функция \rightarrow fn ид сигнатура функции \{ mело блока \}
npomomun \rightarrow fn ид curhamypa функции;
константы \rightarrow const определение константы \{, определение константы\}
определение константы \rightarrowид:простейшее определение типа = выр
mело блока \rightarrow {nepemenhbe|munb||phyhkuux||npomomun||kohcmahmbe||onepamopb||}
eup \rightarrow eup_0 [?eup_0:eup_0]
6bip_0 \rightarrow 6bip_1 \{(|||!|||^{^{-}}|!^{^{-}}) 6bip_1\}
ewp_1 \rightarrow ewp_2 \{ (\&\& | !\&\&) ewp_2 \}
6bip_2 \rightarrow \{!\}6bip_3
6bip_3 \rightarrow 6bip_4 \{ (<|>|<=|>=|=|!=) 6bip_4 \}
eup_4 \rightarrow eup_5 \{(I|^{\sim}I|^{\wedge}|^{\sim})eup_5\}
ebip_5 \rightarrow ebip_6 \{ (\&|^{\sim}\&| \ll |\gg|) \otimes bip_6 \}
eup_6 \rightarrow {^{\sim}}eup_7
6blp_7 \rightarrow 6blp_8 \{(+/-)6blp_8\}
sup_8 \rightarrow sup_9 \{ (*//\%/.)sup_9 \}
\varepsilon \omega p_9 \rightarrow \varepsilon \omega p_{10} [(**/**.)\varepsilon \omega p_9]
eup_{10} \rightarrow [+/-/\#]eup_{11}
выp_{11} \rightarrow выp_{12} [##выp_{12}]
6ыp_{12} \rightarrow целое|вещественное|true|false|строковое|(6ыp)|имя|сост знач
cocm знач \rightarrowзнач cmpykm/знач массива
знач структ \to [!:имя модуля::]имя типа::имя части типа\{значение поля\{,значение поля\}\}
значение nоля \rightarrowид <- \epsilonыр
знач\_массива \rightarrowarray[:выр,элементарный mun:]{списвыр}
uмя \rightarrow [[:uмя модуля::][uмя типа::uмя части типа::]ид{([списвыр])|[списвыр]]@[.ид}
cnuceup \rightarrow eup\{.eup\}
uмя модуля \rightarrowид
uмя muna \rightarrow uд
uмя uасmи mиnа \rightarrowид
onepamopы \rightarrow [onepamop] \{; [onepamop] \}
onepamop \rightarrow uu\kappa na | npuc ва uвания | nepexoda | ввода | вывода | условный | разбора
yсловный \rightarrowif выр then{meло блока}{elif} выр then{meло блока}}[else{meло блока}]endif
uu\kappa \wedge a \rightarrow [<:u\pi:>] (for u\pi: выр\{me \wedge a\}[else\{me \wedge a\}]]
           for ид := выр, выр[,выр] \{ mело блока \} [else \{ mело блока \}] |
           while выр{тело блока}|repeat{тело блока}until выр
           spider{случай {,случай} [else{meло блока}]})
cлучай \rightarrowcase выр\{тело блока\}
npuceaueahua \to uma \ (:=|+:=|-:=|*:=|/:=|/.:=|**:=|**:=|%:=|^*\&:=|\ll:=|\gg:=||:=|^*|:=|^*:=|
                         ||:=|!||:=|^^:=|!^^:=|&&:=|!&&:=|>>>:=) выр
nepexoda \rightarrow break [ид]/continue [ид]/return [выр]
ввода \rightarrowread(списимён)
вывода \rightarrow print(cnucвыр)
cnucum\ddot{e}_{H} \rightarrow ums\{,ums\}
разбора \rightarrow match выр{образец ->{mело блока}{,образец ->{mело блока}}[else{mело блока}]}
образец \rightarrow [:имя модуля::]имя типа::имя части типа\{...\}
   Здесь целочисленные и вещественные литералы выглядят так (записаны соответствующие ре-
гулярные определения):
uenoe \rightarrow decsmuuhoe|восьмеричное|двоичное|шестнадиатеричное
\partial ecsmuчнoe \rightarrow \partial ec uuфра('?\partial ec uuфра)*
восьмеричное \rightarrow 0овосьм цифра('?восьм цифра)*
\partial Bouvehoe \rightarrow O(b|B)\partial Bouveh uuppa('?\partial Bouveh uuppa)*
uecmhaduamepuuhoe \rightarrow 0(x|X)uecmh uuppa('?uecmh uuppa)*
\partial ec \quad uu\phi pa \rightarrow 0|1|2|3|4|5|6|7|8|9
восьм uu\phi pa \to 0|1|2|3|4|5|6|7
```

```
d b o u u u u u d p a \rightarrow 0 | 1
u e c m u u u d p a \rightarrow 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | a | b | c | d | e | f | A | B | C | D | E | F
b e u e c m b e u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c m b u e c
```

Eсли перед строковым или символьным литералом не указано ничего, то считается, что это строковый литерал в UTF-32. То же происходит, если перед строковым литералом указано и32. Eсли указано и16, то строковый литерал — в кодировке UTF-16. Наконец, если указано и8 — то в кодировке UTF-8.

Идентификатор представляет собой последовательность букв, цифр, и знаков подчёркивания, и может начинаться либо с буквы, либо со знака подчёркивания. Под буквой понимается либо русская буква, либо латинская буква.

При реализации сканера для хранения целочисленного значения лексемы следует использовать значение типа unsigned \_\_int128, а для хранения вещественного — значение типа \_\_float128.