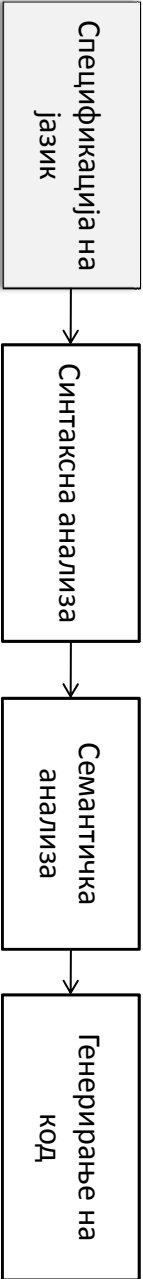


Спецификација на јазик I дел

Фази на изработка на компајлер

► Дијаграм



Дефиниција на алгоритам

- ▶ PASCAL User Manual and Report - Niklaus Wirth
- ▶ Алгоритам или компјутерска програма се состои од два основни дела: **опис на функциите** што треба да се извршат и **опис на податоците** кои се манипулираат со т.н. изрази и истите се опишани со помош на декларации и дефиниции.

▷

Јазикот Inger

- ▶ <http://inger.sourceforge.net/html/about.html>
- ▶ Именуван по Inger Vermeir
- ▶ Конструкција на јазикот Inger:
 - ▶ Дефинирање на податоци
 - ▶ Начини за манипулација на податоците

▷

Споредба Inger - C

- ▶ Основни разлики:
 - ▶ Inger е поедноставен од C (нема struct)
 - ▶ Во синтаксата на наредбите на Inger за повторување и за избор задолжително е употребата на заградите – без разлика на бројот на командните линии






Опис на синтаксата на програмската структура

- ▶ Синтаксен дијаграм
- ▶ Backus-Naur Form (BNF)
- ▶ Extended Backus-Naur Form (EBNF)



Синтаксен дијаграм

- ▶ Додатен (additional) синтаксен дијаграм 
- ▶ Краен симбол 
- ▶ Тек 

▶ ПРИМЕРИ



Backus-Naur Form (BNF)

- ▶ Составено од повеќе линии (production rule)
- ▶ Синтакса на линија :

име_на_синтаксен_дијаграм : содржина

- ▶ Содржината е низа од терминали (задебелени букви) и нетерминали (нормален фонт)



Extended Backus-Naur Form (EBNF)

- ▶ Проширување на BNF
- ▶ Се користат:
 - ▶ `()` – избор
 - ▶ `[и]` – опција
 - ▶ `((и))` – задолжително
 - ▶ `{ и }` – повторување од нула или повеќе пати



Пример за програма во lnger

- ▶ Програма во lnger за пресметување на факториел на бројот 6

```
module primer;
factor : int n → int
{
    int factor = 1;
    int i = 1;
    while ( i <= n ) do
    {
        factor = factor * n;
        n = n + 1;
    }
    return( factor );
}

start main: void → void
{
    int f ;
    f = factor ( 6 );
}
```



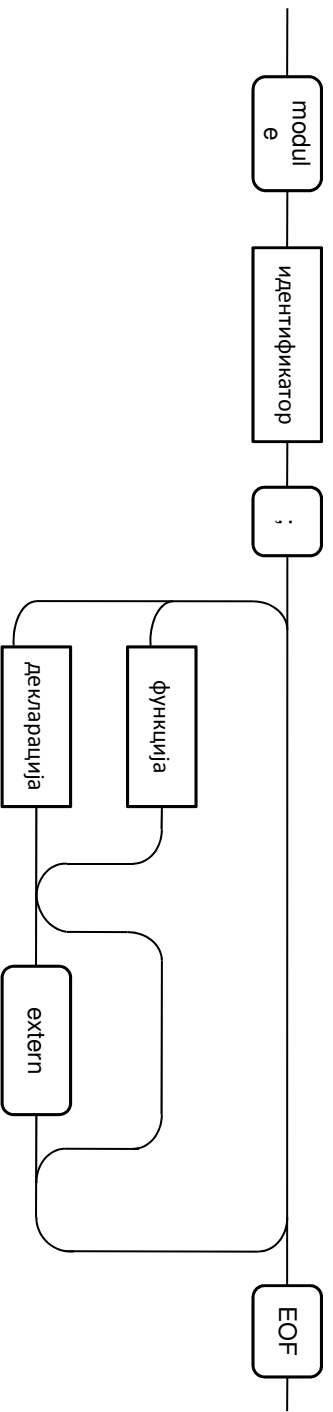
Структура на програма во Inger

- ▶ Еден или повеќе модули, сместени во посебни изворни датотеки
- ▶ Секој модул:
 - ▶ Започнува со име
 - ▶ Содржи нула или повеќе функции
 - ▶ Содржи нула или повеќе глобални променливи



Модул

- ▶ Синтаксен дијаграм



Модул - граматика

► BNF

модул : **module** идентификатор ; глобални

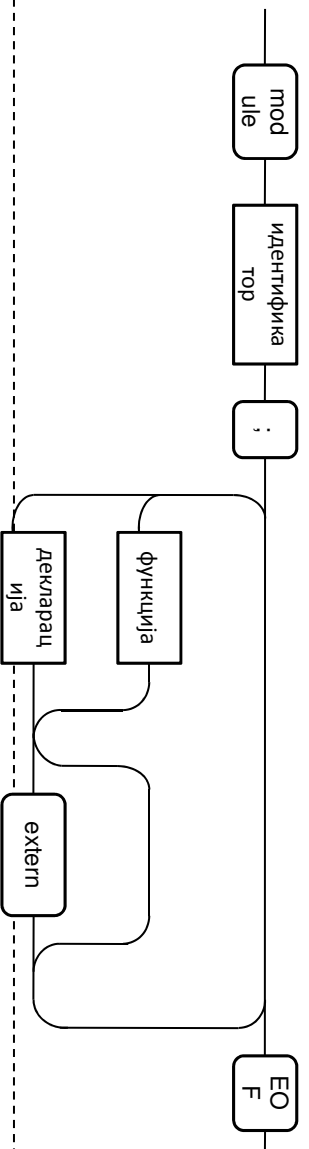
глобални : €

глобални : глобално глобални

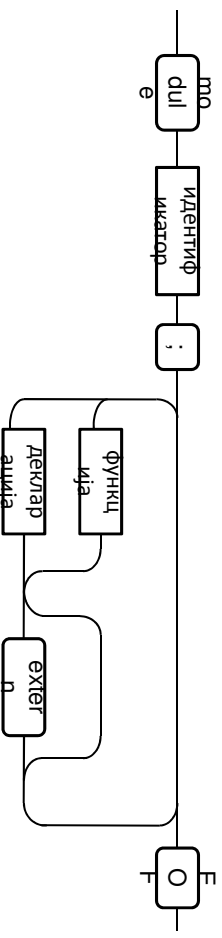
глобални : **extern** глобално глобални

глобално : функција

глобално : декларација



Модул - граматика



EBNF

модул : **module** идентификатор ; { [**extern**]

(функција | декларација) }

- | - или
- [] - опциони загради, изразот внатре може но не мора да се појави (еднаш или нула пати)
- {} - изразот внатре може да се појави нула или повеќе пати (*)
- () - изразот внатре мора да се појави еднаш или повеќе пати (+)



Пример

► Валиден пример:

module Program;
extern функција
декларација
функција

► Невайдэн прымер

```
module Program;
extern функција
декларација
extern
```



Генерирање на програма

▼ BNF

модул: **module** идентификатор ; глобални

глобални : € | глобално глобални | **extern** глобално глобални

глобално : функција | декларација

▼
MODYJI

→ **module** идентификатор ; глобални

- **module** Програм; глобални

► **→ module** Програм; **extern** глобальни

→ **module** Програм; **extern** глобално глобални

→ **module** Program; **extern** функция глобальной

→ **module** Програм; **extern** функција глобално глобални

→ **module** Program; **extern** функција декларација глобални

→ **module** Program: **extern** функція декларативно глобально

→ **module** Program: **extern** функція декларація функції глобальні

→ **module** Program; **extern** функција декларација функција



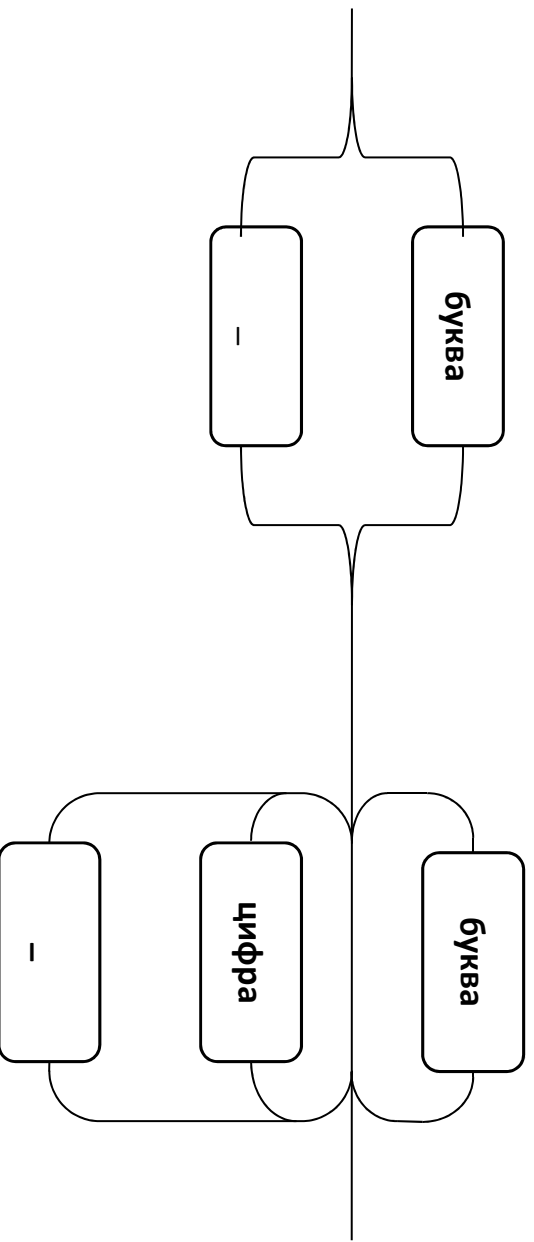
Нотација

- ▶ Нотацијата опфаќа:
 - ▶ Резервирани зборови
 - ▶ `extern, if, while, do, else, break, int, bool, true ...`
 - ▶ Оператори
 - ▶ `+, -, *, >, <, ~, !, &, |, &&, ||, ?, <= ...`
 - ▶ Разделувачи
 - ▶ `{, }, ;, (,), -, >, [,], : и ,`
 - ▶ Коментари
 - ▶ `/* */` - коментарот може да се протега на повеќе линии
 - ▶ `//` - коментарот се протега во една линија



Идентификатор

- ▶ Синтаксен дијаграм



Идентификатор

► BNF

идентификатор : **буква** додаток

идентификатор : _ додаток

додаток: €

додаток : **буква** додаток

додаток : **цифра** додаток

додаток : _ додаток

► EBNF

идентификатор: (_ | буква) { буква | цифра | _ }

буква: **A** | ... | **Z** | **a** | ... | **z**

цифра: **0** | ... | **9**



Идентификатор

► Валидни ідентифікатори

- _Promeliva_D

- Promeliva_D

- Promeliva25D

► Неваідні ідентифікатори

- 25Promeliva_D

- Promeliva&D

- *Promeliva_25D



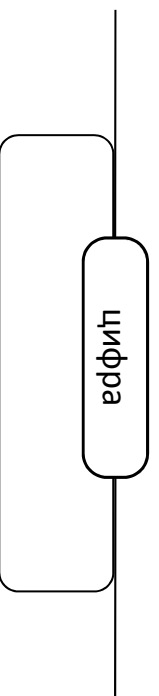
Нумерички информации

- I. Целобројни
 - ▶ Декадни
 - ▶ Бинарни
 - ▶ Хексадецимални
- II. Броеви со подвижна запирка



Нумерички информации

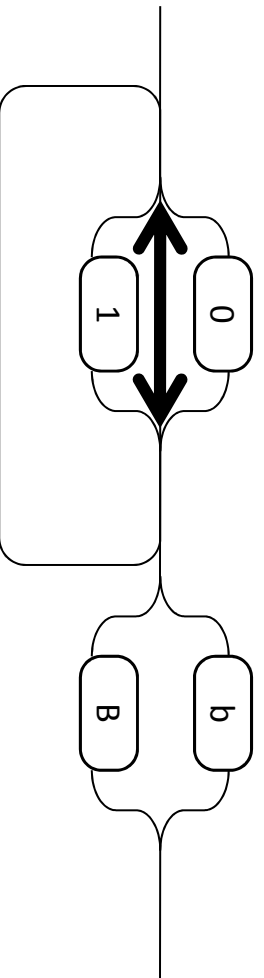
- I. Целобројни
 - ▶ Декадни
 - ▶ се состојат само од цифри
 - ▶ се 32 битни
 - ▶ Приказ со синтаксен дијаграм



Нумерички информации

I. Целобројни

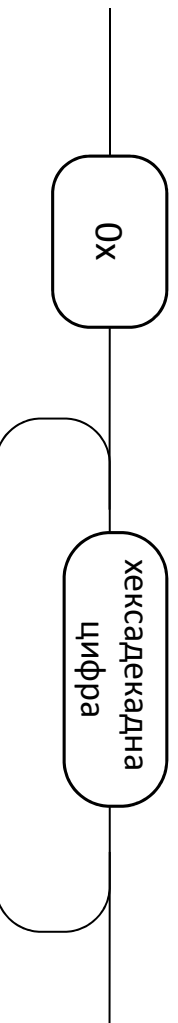
- ▶ Бинарни
 - ▶ се состојат само од цифрите 0 и 1
 - ▶ мора да завршуваат со В или в
 - ▶ Приказ со синтаксен дијаграм



Нумерички информации

I. Целобројни

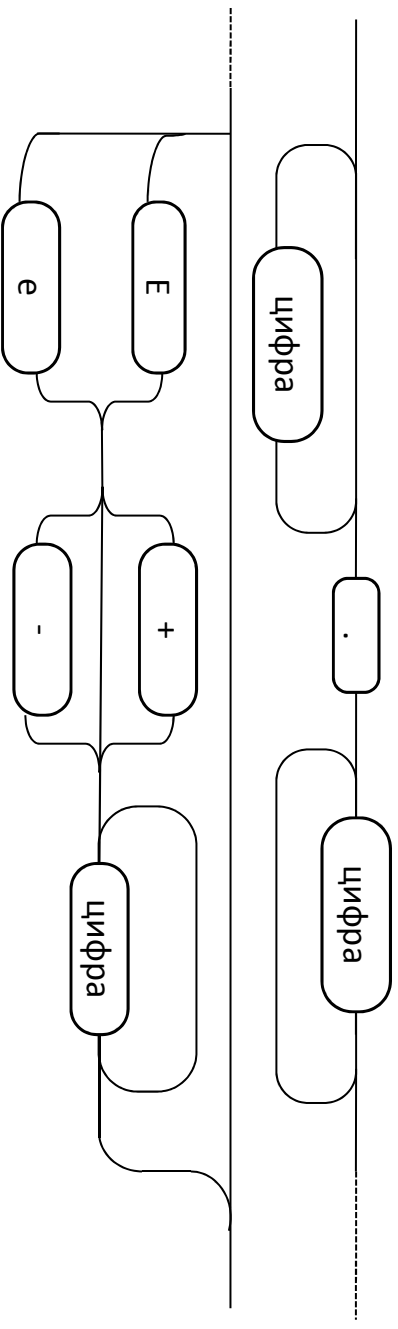
- ▶ Хексадецимални
 - ▶ се состојат од цифрите од 0 до 9 и буквите од А до F
 - ▶ како префикс се додава 0x
 - ▶ Приказ со синтаксен дијаграм



Нумерички информации

II. Броеви со подвижна запирка

- Примери
 - ▶ 0.2, 56.7 , 12e-3
- Приказ со синтаксен дијаграм



Нумерички информации

- | ▶ Правилно | ▶ Неправилно |
|------------|--------------|
| ▶ 5 | ▶ 5a |
| ▶ 0x2e | ▶ 0x2h |
| ▶ 1001b | ▶ 1001a |
| ▶ .34e-2 | ▶ 34e-2 |



Алфанумерички информации

- ▶ Карактер (‘ ’)
 - ▶ Пр: ‘b’, ‘&’, ‘7’, ‘ ’, ‘ ’, ‘ ’
- ▶ Низа карактери-стринг (“ ”)
 - ▶ Пр: “Ova e string.”,
 - ▶ “5 7 89”, “\r\n\t”, “\”zdravo!\””

▷

Алфанумерички информации

▶ Специјални знаци

Escape Sequence	Специјален карактер
\"	"
\'	'
\\	\
\a	Звучен сигнал
\b	Место наазад
\Bbbbbbbb	Конвертирање на бинарна вредност во карактер
\f	Form feed
\n	Нов ред
\oпп	Конвертирање на октална вредност во карактер
\r	Carriage return
\t	Хоризонтален таб
\v	Вертикален таб
\хпп	Конвертирање на хексадецимална вредност во карактер

▷

Податоци

- ▶ Inger програмите работат на повисоко ниво и нудат повеќе податочни апстракции:
 - ▶ bool
 - ▶ char
 - ▶ float
 - ▶ int
 - ▶ untyped



Boolean

- ▶ Вредности:
 - ▶ True
 - ▶ False
- ▶ Оператори

Оператор	Операција
&&	Логичко И
	Логичко ИЛИ
!	Негација
==	Еднаквост
!=	Нееднаквост
=	Доделување на вредност



Int

- Вредности
- Поддржува позитивни и негативни (signed) целобројни вредности
- Целобројни вредности:

Големина на зборот	Ранг
8 бита	-128,..., 127
16 бита	-32768, ..., 32767
32 бита	-2147483648, ..., 2147483647

- Overflow

Int

- Оператори

Оператор	Операција	Асоцијативност	Оператор	Операција	Асоцијативност
+	Унарен плус	Десно (унарен)	<	Помало	Лево
-	Унарен минус	Десно (унарен)	<=	Помало или еднакво	Лево
~	Комплемент на битовите	Десно (унарен)	>	Поголемо	Лево
*	Множење	Лево	>=	Поголемо или еднакво	Лево
/	Делење	Лево	==	Еднаквост	Лево
%	Модул	Лево	!=	Нееднаквост	Лево
+	Собирање	Лево	&	И меѓу битови	Лево
-	Одземање	Лево		ИЛИ меѓу битови	Лево
>>	Поместување на битовите во десно	Лево	^	Ексклузивно ИЛИ меѓу битови	Лево
<<	Поместување на битовите во лево	Лево	=	Доделување	Десно

Float

- ▶ Вредности
- ▶ Релани броеви
- ▶ 8 бајти
- ▶ Оператори :
- ▶ Подмножество од оператори кај int

Оператор	Операција	Оператор	Операција
+	Унарен плус	<=	Помало или еднакво
-	Унарен минус	>	Поголемо
*	Множење	>=	Поголемо или еднакво
/	Делење	==	Еднаквост
+	Собирање	!=	Нееднаквост
-	Одземање	=	Доделување
<	Помало		

Char

- ▶ Вредности
- ▶ Се зачувуваат единични неозначени (unsigned) бајтови (8 бита)
- ▶ Оператори
- ▶ Сите операции кои можат да се извршат на податоците од тип int

▶ Пример на иницијализација

```
char prva_bukva = 'a';  
char nov_red = '\n';
```

Untyped

- ▶ Нема фиксна големина
- ▶ Полиморфистички тип
 - ▶ Се користи како покажувач на секаков тип на податоци
- ▶ Мора да се користи како покажувач
- ▶ Нивоа на индирекција
 - ▶ Се означува со една или повеќе ѕвездички (*), кои индицираат на едно или повеќе нивоа на индирекција



Untyped

- ▶ Валидни примери
 - ▶ `untyped *a;`
 - ▶ `untyped ***b;`
- ▶ Невалидни примери
 - ▶ `untyped a;`
 - ▶ `int *b;`



Декларација

- ▶ Дефиниција
 - ▶ **Декларација** на променлива или функција е:
 - ▶ доделување на тип на променилива на променливата и влезно и излезни типови на параметрите на функцијата
 - ▶ доделување на име на променливата или функцијата
- ▶ Правила
 - ▶ Променливите мора да се декларираат пред да се користат
 - ▶ функциите можат да се користат и пред да се дефинираат, ако претходно се декларирани



Декларација на променлива

- ▶ Глобални променливи
 - ▶ Декларирани надвор од сите функции
- ▶ Локални променливи
 - ▶ Декларирани во внатрешноста на функција

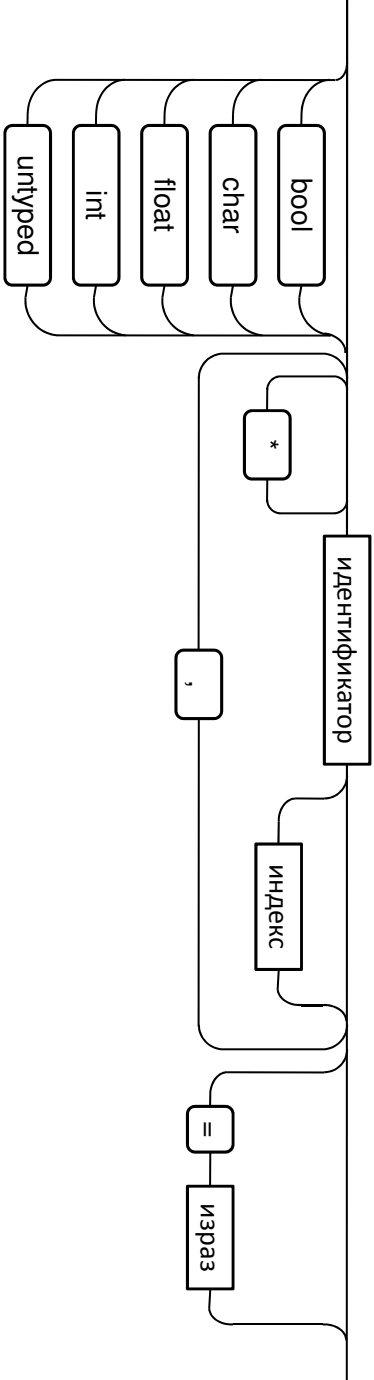
▶ Пример

```
module primer;
int n ; //декларација на глобална променлива
start funksiја: void → void
{
    bool b = false ; //декларација и иницијализација на локална
    променлива
    untured *u; // декларација на локална променлива
    //...кодот продолжува
}
```



Декларација на променлива

► Синтаксен дијаграм



► EBNF

блоказадекларација: тип декларација { , декларација }
декларација: { * } идентификатор { [број] } [= израз]
тип: **bool** | **char** | **float** | **int** | **untyped**
број : (0 | 1 | 2 | ... | 9) { 0 | 1 | 2 | ... | 9 }

Декларација на променлива

► BNF

блоказадекларација: тип декларација додаток
додаток: €
додаток: , декларација додаток
декларација: * декларација
декларација: идентификатор индекс
декларација: идентификатор индекс = израз
индекс: €
индекс: [број] индекс
тип: bool
тип: char
тип: float
тип: int
тип: untyped
број: **цифра** број
број: €



Декларација на променлива

- ▶ Валидни примери
 - ▶ `char a, b = 'Q', *c = 0x0;`
 - ▶ `int number = 0;`
 - ▶ `bool completed = false, found = true;`
 - ▶ `float niza[5];`



Прашања

