Универзитет „Св. Кирил и Методиј“ во Скопје
ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Алгоритми

Структурно програмирање

ФИНКИ 2014

Дефиниции

- Алгоритми ?
- Chamber's on-line dictionary "A set of prescribed computational procedures for solving a problem; a step-by-step method for solving a problem."
- Knuth, "The Art of Computer Programming" … " An algorithm is a finite, definite, effective procedure, with some input and some output."
  - ***Finiteness:*** *"An algorithm must always terminate after a* ***finite number of steps****"*
  - ***Definiteness:*** *"Each step of an algorithm must be* ***precisely defined;*** *the actions to be carried out must be rigorously and unambiguously specified for each case"*
  - ***Input:*** *"…quantities which are given to it initially before the algorithm begins. These inputs are taken from specified sets of objects"*
  - ***Output:*** *"…quantities which have a specified relation to the inputs"*
  - ***Effectiveness:*** *"… all of the operations to be performed in the algorithm must be sufficiently basic that they can in principle be done exactly and in a finite length of time by a man using paper and pencil"*

Алгоритми

- **Алгоритам –**
  - **постапка која се состои од конечно множество на точно дефинирани дејства (операции),**
  - **операции применети врз влезните податоци, по строго пропишан редослед, кои доведуваат до излезни резултати**
- **алгоритамски чекори** - дејства од кои се состои еден алгоритам
- Зависно од општоста на чекорите алгоритамот може да биде:

  - општ

  - детален

Пример за алгоритам

Да се подредат три броја a, b и c по големина.

Општ алгоритам:

чекор-1: Задавање на три броеви.
чекор-2: Подредување на првиот и вториот број по големина.
чекор-3: Подредување на првиот и третиот број по големина.
чекор-4: Подредување на вториот и третиот број по големина.
чекор-5: Печатење на броевите.

Подетален алгоритам:

чекор-1: Задавање на броевите a, b и c.
чекор-2: ако a>b тогаш a $\leftrightarrow$ b
чекор-3: ако a>c тогаш a $\leftrightarrow$ c
чекор-4: ако b>c тогаш b $\leftrightarrow$ c.
чекор-5: Печатење на броевите a, b и c.

Детален алгоритам

чекор-1: Задавање на броевите a, b и c.
чекор-2: ако a>b тогаш
чекор-2-1: pom ←a
чекор-2-2: a← b
чекор-2-3: b←pom
чекор-3: ако a>c тогаш
чекор-3-1: pom ← a
чекор-3-2: a ← c
чекор-3-3: c ← pom
чекор-4: ако b>c тогаш
чекор-4-1: pom ← b
чекор-4-2: b ← c
чекор-4-3: c ← pom
чекор-5: Печатење на броевите a, b и c.

Што е правилен алгоритам?

Правилен алгоритам е оној кој ги исполнува условите:

Во блок-дијаграмите

- има само една влезна линија
- има само една излезна линија
- за секој јазол постои пат од влезната до излезната линија кој минува низ него.

Значи: правилен алгоритам **не смее** да има недостапни сегменти.

Пример

Дали овој алгоритам е ПРАВИЛЕН?

```
алгоритам СоГрешка;
почеток
        читај m, n;
            ако m ≥ n
              тогаш
                печати m,'≥',n;
              инаку
                ако m < n
                  тогаш
                    печати m,'<',n;
                  инаку
                    печати m,'=',n;
                крај_ако {m<n}
            крај_ако {m ≥ n}
        печати 'Каде е грешката ?';
крај
```

Пример

Дали овој алгоритам е ПРАВИЛЕН?

Има недостапен сегмент

```
алгоритам СоГрешка;
почеток
        читај m, n;
            ако m ≥ n
              тогаш
                печати m,'≥',n;
              инаку
                ако m < n
                  тогаш
                    печати m,'<',n;
                  инаку
                    печати m,'=',n;
                крај_ако {m<n}
            крај_ако {m ≥ n}
        печати 'Каде е грешката ?';
крај
```

Пример

Дали овој алгоритам е ПРАВИЛЕН?

Има недостапен сегмент

```
алгоритам СоГрешка;
почеток
        читај m, n;
            ако m ≥ n
              тогаш
                печати m,'≥',n;
              инаку
                ако m < n
                  тогаш
                    печати m,'<',n;
                  инаку
                    печати m,'=',n;
                крај_ако {m<n}
            крај_ако {m ≥ n}
        печати 'Каде е грешката ?';
крај
```

Пример

Дали овој алгоритам е ПРАВИЛЕН?

Има недостапен сегмент

Што треба да се исправи за да биде правилен?

```
алгоритам СоГрешка;
почеток
        читај m, n;
            ако m ≥ n
              тогаш
                печати m,'≥',n;
              инаку
                ако m < n
                  тогаш
                    печати m,'<',n;
                  инаку
                    печати m,'=',n;
                крај_ако {m<n}
            крај_ако {m ≥ n}
        печати 'Каде е грешката ?';
крај
```

Пример

Дали овој алгоритам е ПРАВИЛЕН?

Има недостапен сегмент

Што треба да се исправи за да биде правилен?

ако m>n

```
алгоритам СоГрешка;
почеток
        читај m, n;
            ако m ≥ n
              тогаш
                  печати m,'≥',n;
              инаку
                  ако m < n
                     тогаш
                         печати m,'<',n;
                     инаку
                         печати m,'=',n;
                  крај_ако {m<n}
            крај_ако {m ≥ n}
        печати 'Каде е грешката ?';
крај
```

Правилно градење алгоритми

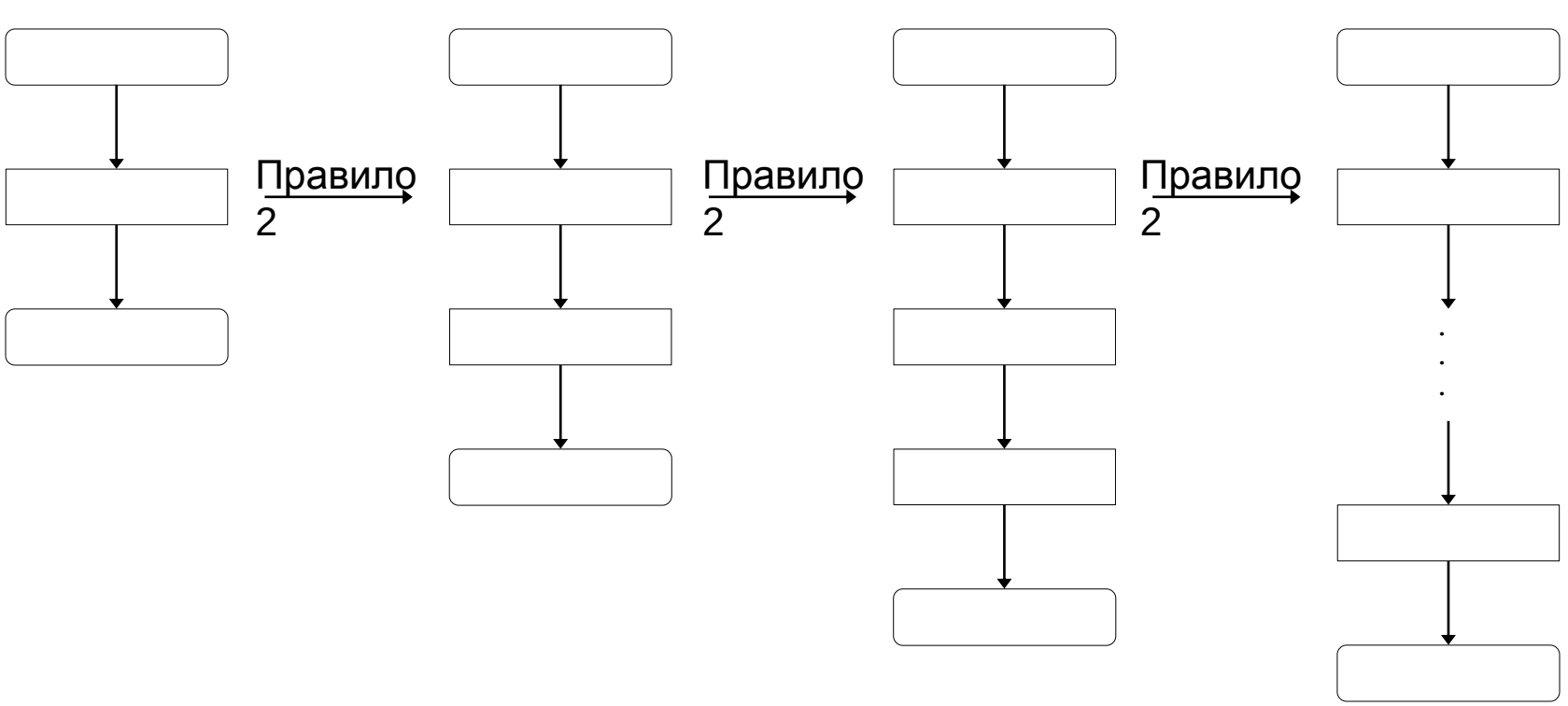
- Правила
  1. Започнете со наједноставниот блок-дијаграм
  2. Секој правоаголник (процес) може да се замени со два последователни правоаголника (процеси)
  3. Секој правоаголник (процес) може да се замени со која било контролна структура (секвенца, if, if/else, while, do/while, for)
  4. Правилата 2 и 3 може да се применат во кој било редослед и поголем број пати.

Правило 1

(започни со наједноставниот Блок-дијаграм)

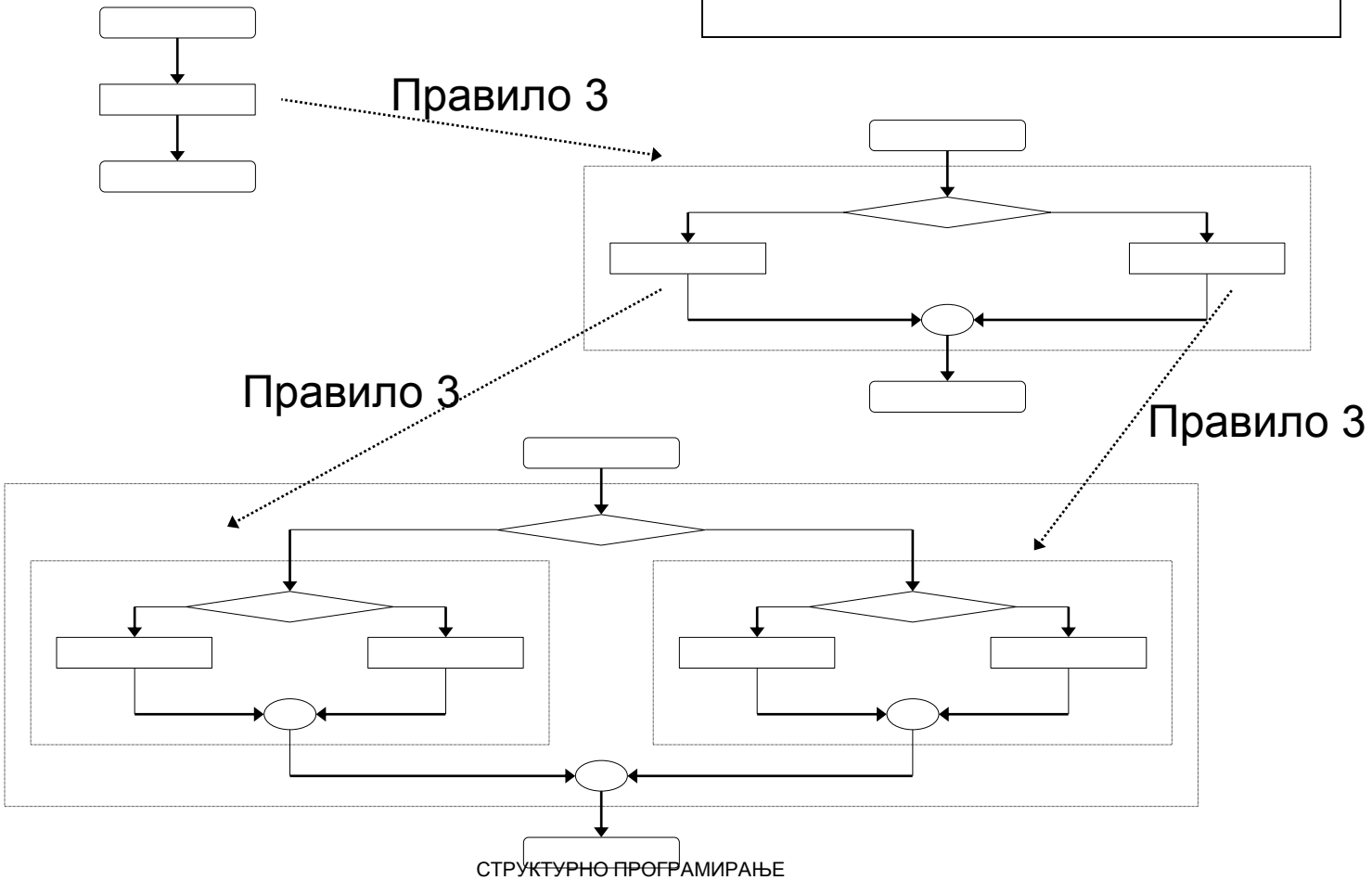
Правило 2

Секој правоаголник може да се замени со два последователни правоаголници



Правило 3

Секој правоаголник може да се замени со контролна структура за избор



Анализа на алгоритми

- Што може да се анализира?

- Може:
  - да се одреди времето на извршување на алгоритамот како функција од неговите влезни податоци
  - да се одреди максималното побарување на меморија што е потребна за податоците
  - да се одреди точната големина на програмскиот код
  - да се одреди дали програмата точно го пресметува посакуваниот резултат
  - да се одреди комплексноста на алгоритамот
  - да се види колку добро алгоритамот се соочува со неочекуваните и погрешни влезни податоци

Анализа на сложеност

Анализата на сложеност на алгоритмите овозможува:

- Предвидува кои ресурси се потребни за преработка од страна на дадениот алгоритам
- Сложеноста на алгоритмите се изразува како математичка функција.
- Функцијата ја одредува:
  - зависноста на бројот на влезни податоци со време на извршување на алгоритамот – временска сложеност или
  - зависноста на бројот на податоци со меморискиот простор потребен за извршување на алгоритамот- мемориска сложеност

ДЕФ: Сложеност на алгоритам претставува ниво на тежина при решавање на одреден проблем мерен во време, број на чекори или аритметички операции, меморија.

ПОВЕЌЕ за алгоритми – ТРЕТ семестар...

Прашања?