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

ФАКУЛТЕТ ЗА ИНФОРМАТИЧКИ НАУКИ И КОМПЈУТЕРСКО ИНЖЕНЕРСТВО

Вовед

дел 1 (вовед)

Структурно програмирање

ФИНКИ 2014

Цел

- Цели на предметот
  - Да се воведат студентите во парадигмата на структурното програмирање
  - Да го разберат концептот на алгоритми и да се оспособат да развиваат алгоритми, да кодират, тестираат и компајлираат програми
  - Ќе бидат воведени податочните типови, контролните структури,функциите, полињата, датотеките.

Настава

- Предавања: 2 часа неделно
- Вежби:
  - аудиториски - 2 часа неделно
  - лабораториски - 2 часа неделно
- Самостојна работа

  xy часови неделно
- Консултации
  - термините ќе бидат дополнително објавени ...

Услови за полагање

- Услов за полагање на испитот
  - потпис од наставникот
- Услов за добивање потпис од наставникот
  - потпис од асистентот
- Услов за добивање потпис од асистентот
  - успешно изработени лабораториски вежби (дозволени се најмногу 2 отсуства)

Оценување

- Завршен писмен испит (75%), или
  - Два парцијални испити (35% + 40%)
- 2 теста (15%)
- Лабораториски вежби (10%)

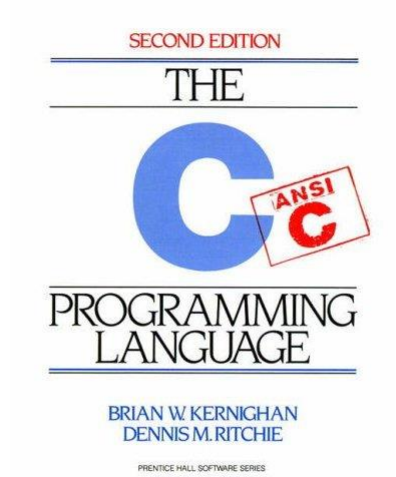
- Услов за положување – минимум 50%

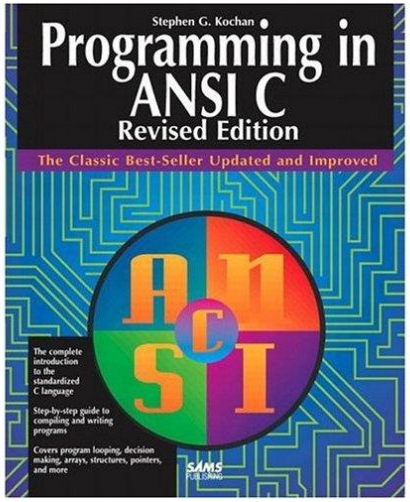
- Прашање: Доколку од лаб. вежби добиете само 1% и не сте биле на тестовите, колку поени (од 100) од писмениот испит треба да освоите за да го положите испитот?

Содржина на предметот

- Техники за дизајнирање програми: како да се моделира реалниот свет и да се менаџира комплексноста
  - Структурно програмирање
- Вештини на решавање проблеми (problem-solving)
  - Како да се реши некој проблем ефикасно и елегантно
  - дебагирање
  - Посебна вештина во програмирањето: рекурзија (раздели па владеј)
- Стил на добро програмирање
  - Робусно, ефикасно програмирање, читлив код, документираност
- Користење на важни податочни структури
  - Променливи, низи, матрици, датотеки
- Програмски алатки
  - Програмски јазик C

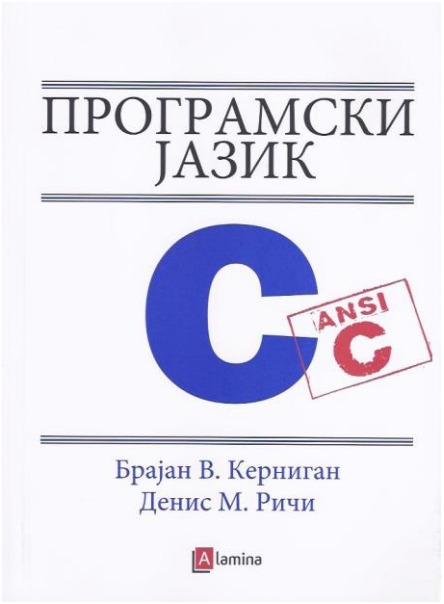
Литература





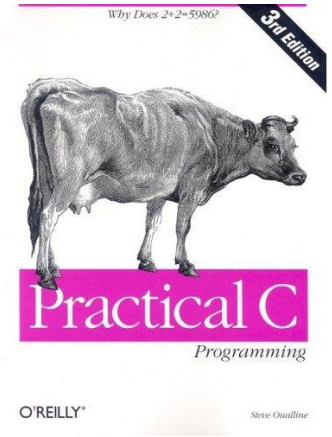
- Kernighan B., Ritchie D., **The C Programming Language**, (2nd edition), Prentice Hall.

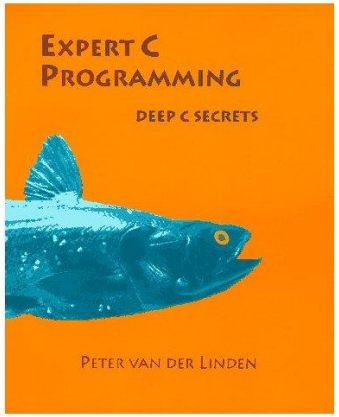
- **Преведена на Македонски:** ПРОГРАМСКИ ЈАЗИК С, Брајан В. Керингам, Денис М. Ричи

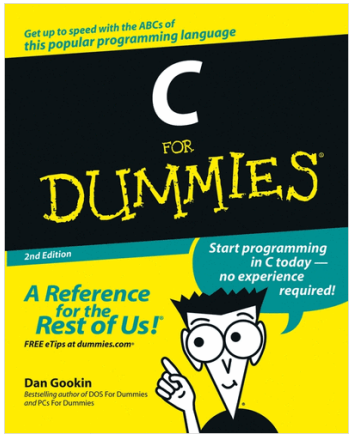


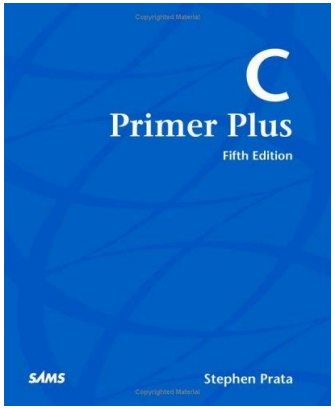
- Kochan C., **Programming in ANSI C**, SAMS Publishing 1994

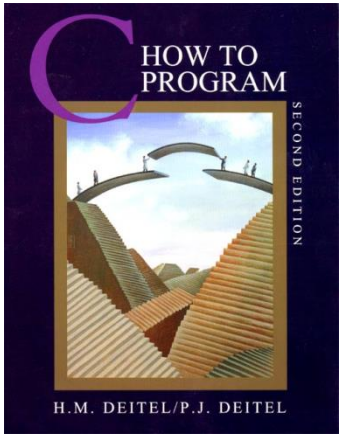
Помошна литература

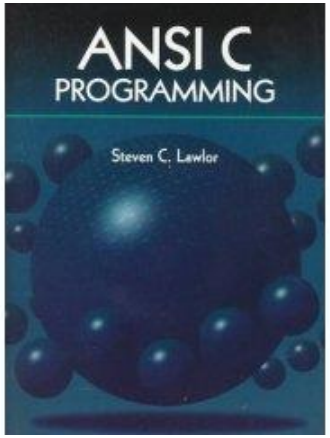


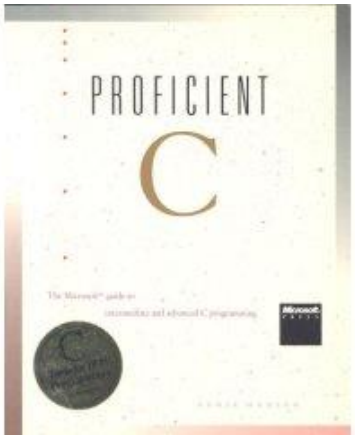


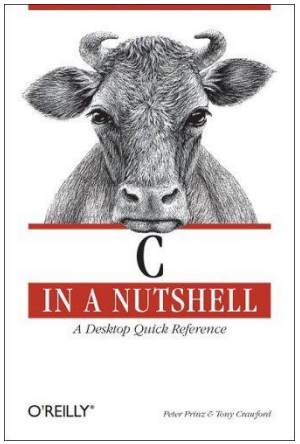












Информатика

- Информатиката е наука за информациите, практиката на обработка на информации и инженерството на информациски системи.
- Информатиката ги истражува структурите, алгоритамите, однесувањето и интеракциите на природни и вештачки системи кои складираат, обработуваат, споделуваат и пристапуваат до **информации**.
- Како да се дизајнираат системи кои ја испорачуваат вистинската информација, на вистинското лице, на вистинско место, во вистинско време, и на вистински начин.

Што е информатика?

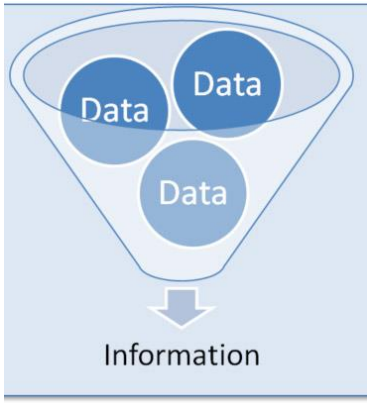
Разни дефиниции, но воглавно 3 дела:

- Наука за информацијата
- Практицирање на процесирање информации
- Градење и користење на информациски систем

Компјутерите, луѓето и организациите процесираат информации

- информатиката има
  - Пресметковен
  - Когнитивен
  - Социјален аспект

Информациска хиерархија (пирамида на знаење)



- Податоци
  - факти, симболи, сигнали
- Информации - кој, што, каде, колку, кога?
  - Податоци на кои им е дадена смисла
- Знаење – како?
  - Збир на корисни информации
- Разбирање – зошто?
  - Когнитивен, аналитички, интерполирачки и пробабилистички процес
- Мудрост - Продлабочено разбирање
  - Екстраполирачки, недетерминистички, непробабилистички процес

DIKW Hierarchy

Податоци и информации

- Податоците се запишуваат во форма разбирлива за човекот:
  - текст, број, слика, звук, ...

John Brown	23	55
William Smith	24	55
Robert Jones	20	40
David Wilson	19	20

Од податок до информација

- Човекот разменува податоци со надворешниот свет, а тие податоци носат информации.
- **Информацијата е содржина на сè она што го разменуваме (даваме и примаме) со надворешниот свет.**
- Во **податоците** често има „скриена“ **информација** која за да се добие истите треба да се **обработат**
- **Обработка на податоци**
  - **процес што се состои од низа дејства што се извршуваат врз податоците за да се добијат информации.**
- Добиените информации треба да се **организираат** за од нив да се извлече **знаење**

Зошто да се изучува развојот на софтвер?

- За да го положите испитот ☺
- Ја задоволите вашата љубопитност
- Подобро разбирање на програмските јазици
- Корисно за студирањето / работата / истражувањето
- Ја унапредува способноста за **прецизно** размислување и решавање проблеми

Што е тоа програмирање?

- Програмирањето на компјутер во најширокото значење е едноставно кажување на компјутерот што треба да стори.
- Програма претставува детален и прецизен опис на некоја појава или процес кој е разбирлив и извршлив од страна на компјутер.
- Програма е низа на точно дефинирани инструкции напишани во некој **програмски јазик** со кои се извршуваат конкретни задачи.

А, како се програмира?

- Програмирањето се состои од два чекора

Дизајнирање (осмислување) на програмата (архитектонско решение)

Кодирање (конструкција, градење)

Програмски јазици

- Програмските јазици се **вештачки јазици** кои се употребуваат за да се контролираат машини, најчесто компјутери.

- Програмските јазици претставуваат точно дефинирани системи од правила за опишување на **задачите на компјутерот** напишани во форма разбирлива и **за компјутерот** и **за програмерот.**

Поделба на програмските јазици

Може да се изведе од неколку аспекти:

- Генерации на програмски јазици
- Јазици со општа и јазици со специјална намена
- Декларативни и императивни јазици
  - **Како треба** операции - императивни јазици
  - **Што треба** резултати - декларативни јазици

Видови програмски јазици

- **Машински јазици** - множества инструкции напишани со 0 и 1
  - Природен јазик на компјутерите.
  - Макотрпно програмирање подложно на грешки.
  - Машински код тежок за разбирање и измени.
  - Програмирањето е долго и скапо.
- Јазици од **првата генерација** - асемблерски јазици
  - Едноставна модификација на машинските јазици.
  - Зависат од компјутерот на кој работат.
- Јазици од **втората и третата генерација** - имплементациски јазици
  - Слични со говорните јазици.
  - Наредбите се послични на говорниот јазик,
  - Структурирано програмирање.
  - Јазици со општа намена.
  - Едноставност, униформност и преносливост на програмите.
- Јазици од **четвртата и петтата генерација** -апликативни јазици
  - Јазици со специјална намена.

Генерации на програмски јазици

- Компјутерска праисторија - програмирање со хардверски спојки.
- Нулта генерација (до 1954 година) - асемблер
- Прва генерација (1954-1958) - базирани на математички изрази
  - FORTRAN, ALGOL 58, Flowmatic
- Втора генерација (1959-1961)
  - FORTRAN II, ALGOL 60, COBOL, APL, LISP
- Трета генерација (1962-1970)
  - PL/I, ALGOL 68, PASCAL, SIMUAL, BASIC
- Четврта генерација (1970-1980)
  - Smalltalk, C, Prolog, Ada
- Петта генерација (1980-) - објектно ориентирани јазици
  - Modula-3, C++, Object Pascal, Java

Поделба на програмските јазици

Алгоритамски	Декларативни		Објектно ориентирани
Machine code	Функционално апликативни	Логичко програмирање	Smalltalk
Assembly			Simula
FORTRAN			C++
COBOL	Haskell	PROLOG и деривати	Objective-C
ALGOL	ML		Oberon
PL/I	LISP		Modula-3
Pascal	Miranda		Java
Modula-2	Logo		
BASIC	APL		
C			
C++, Java, Modula-3, Oberon			

Синтакса и семантика на програмските јазици

- Правила за пишување на програмите - синтакса.
- Пример:
  - Песот ја прескокна оградата.
  - Оградата прескокна песот ја.
- Синтаксата е студија за значењето, репрезентацијата, граматиката и структурата на еден јазик.
  - Синтаксички точни реченици мора да бидат разбирливи за учесниците во разговорот.
  - Значењето на синтаксички точна реченица - семантика.
  - Пример:
    - Песот го гризна човекот.
    - Човекот го гризна песот.
  - Семантика е студија за значењето, акциите, функциите и однесувањето на еден јазик.

Програмски парадигми

- Компјутерите се исклучително „глупави“ машини кои сè сфаќаат буквално и се неспособни да ги разберат двосмисленостите и недореченостите.

- Затоа компјутерските програми **мора** да бидат **прецизни и недвосмислени.** Токму заради ова, организацијата на овие детали е клучна за успешноста на една програма.

- Различните пристапи кон организацијата на овие детали ги претставува различните начини на програмирање (програмски парадигми).

_____
парадигма – шаблон или шема која сугерира начин или постапка за решавање на проблеми

Програмски парадигми (примери)

- Procedural programming <-> Functional programming
- Structured programming <-> Unstructured programming
- Imperative programming <-> Declarative programming
- Object-oriented programming

- Message passing programming <-> Imperative programming
- Flow-driven programming <-> Event-driven programming
- Scalar programming <-> Array programming
- Class-based programming <-> Prototype-based programming
- Constraint programming <-> Logic programming
- Rule-based programming
- Component-oriented programming
- Aspect-oriented programming
- Table-Oriented Programming
- Pipeline Programming
- Literate programming

- Post-object programming
- Subject-oriented programming
- Reflective programming
- Dataflow programming
- Policy-based programming
- Parallel programming
- Interactive programming

Структурирано програмирање (1)

- Структурирано програмирање е една од програмските парадигми
- Структурираното програмирање претставува техника за организација на програмата во логичка структура на хиерархиски модули.
- Големите рутини се разбиваат на помали модули.
- Употребата на наредбата за безусловен скок (GOTO) не се препорачува.
- Ова овозможува програмата да се разгледува во изолирани блокови меѓу кои интеракцијата е (релативно) контролирана.

Структурирано програмирање (2)

Историски, постојат три главни дефиниции за структурираното програмирање:

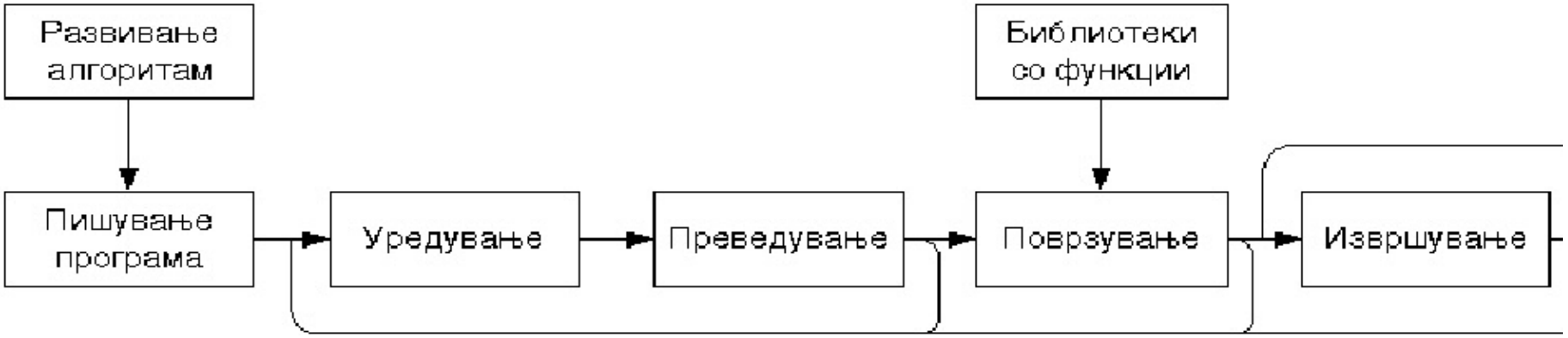
1. Порамнување на податочните структури со програмските структури (Jackson)
2. Логиката на програмата претставува структура составена од слични подструктури кои се комбинираат на строго ограничен број начини (Dijkstra)
3. Разложување на програмата на програмски подструктури со единствен влез (и единствен излез).

Основни конструкции

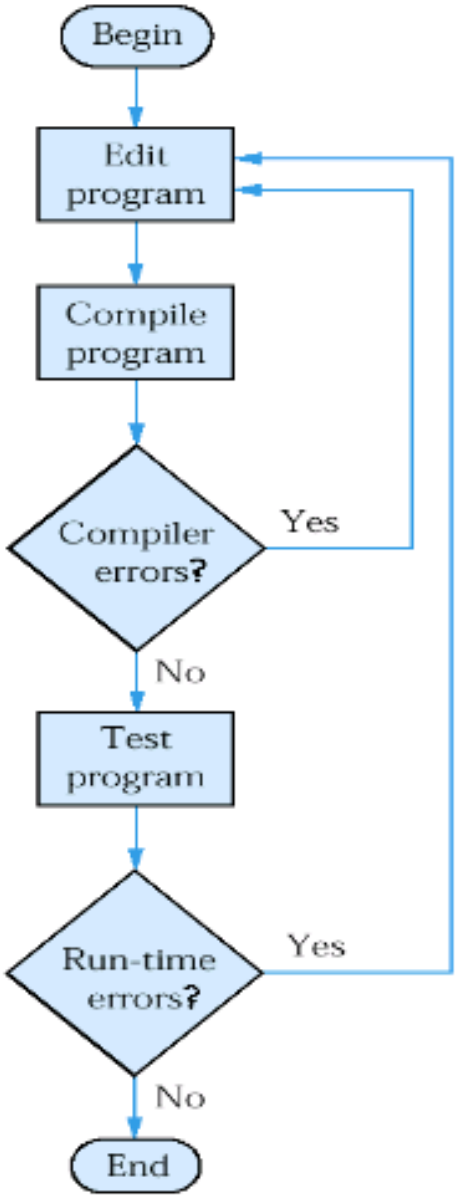
- Основни конструкции:
  - секвенца (sequence)
  - условно гранење (condition)
  - повторување (repetition)
- Структурен дизајн:
  - намалена комплексност
  - подобрена читливост
  - подобрено тестирање
  - подобрено одржување
  - chunking (препознавање на цели процедурални елементи)

Пишување, преведување, поврзување, извршување

- Изворна програма
- Преведувач е програма која програмите напишани во еден програмски јазик ги преведува во друг програмски јазик.
  - синтаксни грешки
- Објектна програма
- Извршна програма
- Интерпретер & Компајлер (преведувач)



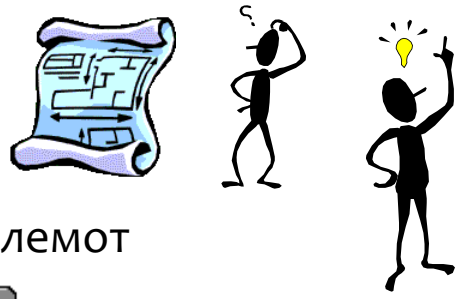
Програмирање: The Edit/Compile/Run Loop



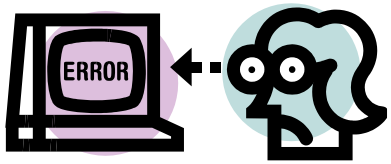
- *Грешки при компајлирање* (*Compile-time errors*)
  - Компајлерот може да најде проблеми со синтаксни грешки
  - Ако постојат грешки при компајлирање, НЕ се формира извршна верзија на програмата
- *Грешки при извршување* (*Run-time errors*)
  - Проблем при извршување на програмата
    - Делење со нула – програмата завршува НЕрегуларно
- *Логички грешки (Logical errors)*
  - Програмата „работи“, но неправилно и дава погрешни резултати

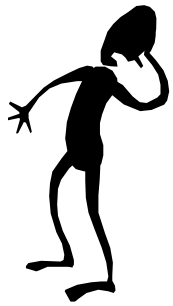
Развој на програма

Проблем кој треба да се реши

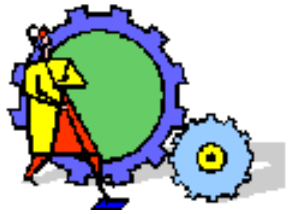


1. Идеја за програма која го решава проблемот
2. Пишување на изворниот код





3. Преведување на изворниот код
4. Плачи поради грешките при преведувањето (опција)



5. Извршување на програмата и тестирање

6. Корни коса поради грешките при извршувањето (опција)
7. Почни одново (задолжително)

Прашања?