M269

Exam Revision

Phil Molyneux

16 May 2015

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

M269 Exam Revision

Agenda & Aims

1. Welcome and introductions
2. Revision strategies
3. M269 Exam — Part 1 has 15 questions 60%
4. M269 Exam — Part 2 has 2 questions 40%
5. M269 Exam — 3 hours, Part 1 100 mins, Part 2 70 mins
6. M269 2013J exam — Part 1 in reverse order
7. M269 2013J exam — Part 2 in notes version
8. Topics and discussion for each question
9. Exam technique

M269 Exam

Revision strategies

- Organising your knowledge
- Each give one exam tip to the group
- TODO: add some more points

M269 Specimen Exam

Q15 Topics

- Unit 7
- Computability and ideas of computation
- Complexity
- P and NP
- NP-complete

Computability

Ideas of Computation

- The idea of an algorithm and what is effectively computable
- **Church-Turing thesis** Every function that would naturally be regarded as computable can be computed by a deterministic Turing Machine. (Unit 7 Section 4)
- See Phil Wadler on computability theory performed as part of the Bright Club at The Strand in Edinburgh, Tuesday 28 April 2015

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J
Q 15

- Which *two* of the following statements are true?

A. A Turing Machine is a mathematical model of computational problems.

B. If the lower bound for a computational problem is $O(n^2)$, then there is an algorithm that solves the problem and which has complexity $O(n^2)$.

C. Searching a sorted list is not in the class NP.

D. The decision Travelling Salesperson Problem is NP-complete.

E. There is no known tractable quantum algorithm for solving a known NP-complete problem.

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J
Q 15 Solution

- Only D and E are true.
- A Universal Turing Machine can compute any *computable* sequence but there are well defined problems that are not computable. (So not A)
- A lower bound may be lower than any actual algorithm. (So not B)
- Every problem in P is in NP — we just do not know if P == NP (So not C)

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Computability

Reducing one problem to another

- To reduce problem P_1 to P_2, invent a construction that converts instances of P_1 to P_2 that have the same answer. That is:
  - any string in the language P_1 is converted to some string in the language P_2
  - any string over the alphabet of P_1 that is not in the language of P_1 is converted to a string that is not in the language P_2
- With this construction we can solve P_1
  - Given an instance of P_1, that is, given a string w that may be in the language P_1, apply the construction algorithm to produce a string x
  - Test whether x is in P_2 and give the same answer for w in P_1

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

Computability
Direction of Reduction

- The direction of reduction is important
- If we can reduce P_1 to P_2 then (in some sense) P_2 is at least as hard as P_1 (since a solution to P_2 will give us a solution to P_1)
- So, if P_2 is decidable then P_1 is decidable
- To show a problem is undecidable we have to reduce from an known undecidable problem to it
- $\forall x(\mathrm{dp}_{P_1}(x) = \mathrm{dp}_{P_2}(\mathrm{reduce}(x)))$
- Since, if P_1 is undecidable then P_2 is undecidable

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Computability

Models of Computation

- In automata theory, a *problem* is the question of deciding whether a given string is a member of some particular language
- If Σ is an alphabet, and L is a language over Σ, that is $L \subseteq \Sigma^*$, where Σ^* is the set of all strings over the alphabet Σ then we have a more formal definition of *decision problem*
- Given a string $w \in \Sigma^*$, decide whether $w \in L$
- Example: Testing for a prime number — can be expressed as the language L_p consisting of all binary strings whose value as a binary number is a prime number (only divisible by 1 or itself)

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

Computability

Church-Turing Thesis & Quantum Computing

- **Church-Turing thesis** Every function that would naturally be regarded as computable can be computed by a deterministic Turing Machine.
- **physical Church-Turing thesis** Any finite physical system can be simulated (to any degree of approximation) by a Universal Turing Machine.
- **strong Church-Turing thesis** Any finite physical system can be simulated (to any degree of approximation) with polynomial slowdown by a Universal Turing Machine.
- Shor's algorithm (1994) — quantum algorithm for factoring integers — an NP problem that is not known to be P — also not known to be NP-complete and we have no proof that it is not in P

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Computability

Turing Machine

- **Finite control** which can be in any of a finite number of *states*
- **Tape** divided into cells, each of which can hold one of a finite number of symbols
- Initially, the **input**, which is a finite-length string of symbols in the *input alphabet*, is placed on the tape
- All other tape cells (extending infinitely left and right) hold a special symbol called *blank*
- A **tape head** which initially is over the leftmost input symbol
- A **move** of the Turing Machine depends on the state and the tape symbol scanned
- A move can change state, write a symbol in the current cell, move left, right or stay

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

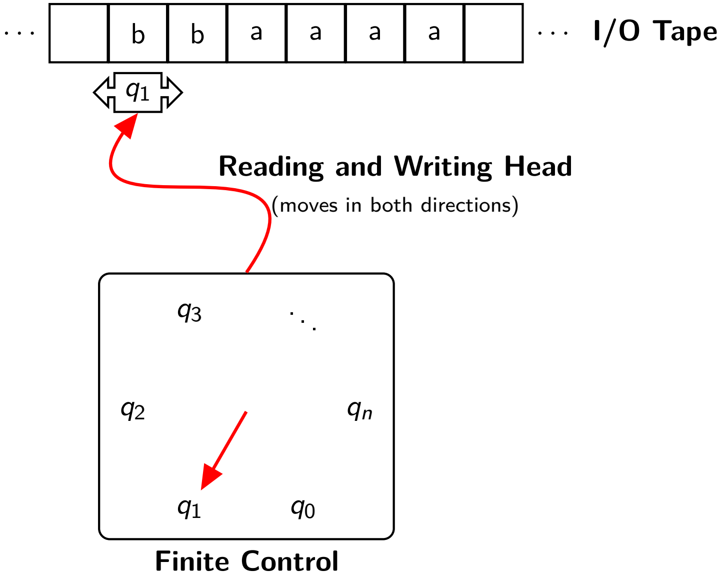
Units 1 & 2

M269 Exam Section 2

Exam Techniques

Turing Machine Diagram

Turing Machine Diagram



M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

Computability

Turing Machine notation

- Q finite set of states of the finite control
- Σ finite set of *input symbols* (M269 S)
- Γ complete set of *tape symbols* $\Sigma \subset \Gamma$
- δ Transition function (M269 instructions, I)
  $\delta :: Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, S\}$
  $\delta(q, X) \mapsto (p, Y, D)$
- $\delta(q, X)$ takes a state, q and a tape symbol, X and returns (p, Y, D) where p is a state, Y is a tape symbol to overwrite the current cell, D is a direction, Left, Right or Stay
- q_0 *start state* $q_0 \in Q$
- B *blank symbol* $B \in \Gamma$ and $B \notin \Sigma$
- F set of *final or accepting states* $F \subseteq Q$

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Computability

Decidability

- **Decidable** — there is a TM that will halt with yes/no for a decision problem — that is, given a string w over the alphabet of P the TM with halt and return yes.no the string is in the language P (same as *recursive* in Recursion theory — old use of the word)
- **Semi-decidable** — there is a TM will halt with yes if some string is in P but may loop forever on some inputs (same as *recursively enumerable*) — *Halting Problem*
- **Highly-undecidable** — no outcome for any input — *Totality, Equivalence Problems*

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Computability

Undecidable Problems

- **Halting problem** — the problem of deciding, given a program and an input, whether the program will eventually halt with that input, or will run forever — term first used by Martin Davis 1952
- **Entscheidungsproblem** — the problem of deciding whether a given statement is provable from the axioms using the rules of logic — shown to be undecidable by Turing (1936) by reduction from the *Halting problem* to it
- **Type inference and type checking** in the second-order lambda calculus (important for functional programmers, Haskell, GHC implementation)
- Undecidable problem — see link to list

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

Computability

Why undecidable problems must exist

- A *problem* is really membership of a string in some language
- The number of different languages over any alphabet of more than one symbol is uncountable
- Programs are finite strings over a finite alphabet (ASCII or Unicode) and hence countable.
- There must be an infinity (big) of problems more than programs.

Computability

Computability and Terminology (1)

- The idea of an *algorithm* dates back 3000 years to Euclid, Babylonians...
- In the 1930s the idea was made more formal: which *functions* are *computable*?
- A *function* a set of pairs $f = \{(x, f(x)) : x \in X \wedge f(x) \in Y\}$ with the *function property*
- *Function property:* $(a, b) \in f \wedge (a, c) \in f \Rightarrow b == c$
- *Function property:* Same input implies same output
- Note that maths notation is deeply inconsistent here — see Function and History of the function concept
- *What do we mean by computing a function — an algorithm ?*

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Computability

Computability and Terminology (2)

- In the 1930s three definitions:
- λ-Calculus, simple semantics for computation — Alonzo Church
- General recursive functions — Kurt Gödel
- Universal (Turing) machine — Alan Turing
- Terminology:
  - Recursive, recursively enumerable — Church, Kleene
  - Computable, computably enumerable — Gödel, Turing
  - Decidable, semi-decidable, highly undecidable
  - In the 1930s, *computers* were *human*
  - Unfortunate choice of terminology
- Turing and Church showed that the above three were equivalent
- Church-Turing thesis — function is intuitively computable if and only if Turing machine computable

Complexity

P and NP

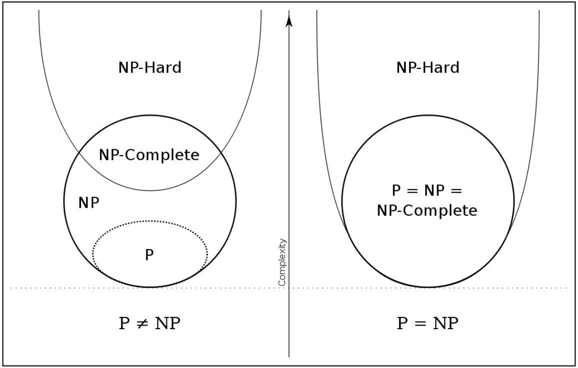
- P, the set of all decision problems that can be solved in polynomial time on a deterministic Turing machine
- NP, the set of all decision problems whose solutions can be verified (certificate) in polynomial time
- Equivalently, NP, the set of all decision problems that can be solved in polynomial time on a non-deterministic Turing machine
- A *decision problem*, *dp* is *NP-complete* if
  1. *dp* is in NP and
  2. Every problem in NP is reducible to *dp* in polynomial time
- *NP-hard* — a problem satisfying the second condition, whether or not it satisfies the first condition. Class of problems which are at least as hard as the hardest problems in NP. NP-hard problems do not have to be in NP and may not be decision problems

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Complexity

P and NP — Diagram

Euler diagram for P, NP, NP-complete and NP-hard set of problems



Source: Wikipedia NP-complete entry

Complexity

NP-complete problems

- Boolean satisfiability (SAT) Cook-Levin theorem
- Conjunctive Normal Form 3SAT
- Hamiltonian path problem
- Travelling salesman problem
- NP-complete — see list of problems

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

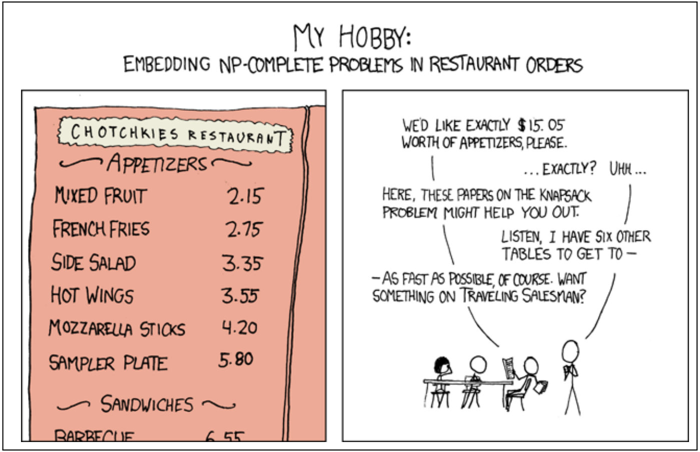
Units 1 & 2

M269 Exam Section 2

Exam Techniques

Complexity

Knapsack Problem



Source & Explanation: XKCD 287

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

NP-Completeness and Boolean Satisfiability

Points on Notes

- The *Boolean satisfiability problem (SAT)* was the first decision problem shown to be *NP-Complete*
- This section gives a sketch of an explanation
- **Health Warning** different texts have different notations and there will be some inconsistency in these notes
- **Health warning** these notes use some formal notation *to make the ideas more precise* — computation requires precise notation and is about manipulating strings according to precise rules.

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

NP-Completeness and Boolean Satisfiability

Alphabets, Strings and Languages

- Notation:
- Σ is a set of symbols — the alphabet
- Σ^k is the set of all string of length k, which each symbol from Σ
- Example: if $\Sigma = \{0, 1\}$
  - $\Sigma^1 = \{0, 1\}$
  - $\Sigma^2 = \{00, 01, 10, 11\}$
- $\Sigma^0 = \{\epsilon\}$ where ϵ is the empty string
- Σ^* is the set of all possible strings over Σ
- $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \ldots$
- A *Language*, L, over Σ is a subset of Σ^*
- $L \subseteq \Sigma^*$

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

NP-Completeness and Boolean Satisfiability

Language Accepted by a Turing Machine

- Language accepted by Turing Machine, M denoted by $L(M)$
- $L(M)$ is the set of strings $w \in \Sigma^*$ accepted by M
- For *Final States* $F = \{Y, N\}$, a string $w \in \Sigma^*$ is accepted by $M \Leftrightarrow$ (if and only if) M starting in q_0 with w on the tape halts in state Y
- Calculating a function (function problem) can be turned into a decision problem by asking whether $f(x) = y$

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

NP-Completeness and Boolean Satisfiability

The NP-Complete Class

- If we do not know if P $\neq$ NP, what can we say ?
- A language L is *NP-Complete* if:
  - $L \in$ NP and
  - for all other $L' \in$ NP there is a *polynomial time transformation* (Karp reducible, reduction) from L' to L
- Problem P_1 *polynomially reduces* (Karp reduces, transforms) to P_2, written $P_1 \propto P_2$ or $P_1 \leq_p P_2$, iff $\exists f : \mathrm{dp}_{P_1} \to \mathrm{dp}_{P_2}$ such that
  - $\forall I \in \mathrm{dp}_{P_1} [I \in Y_{P_1} \Leftrightarrow f(I) \in Y_{P_2}]$
  - f can be computed in polynomial time

NP-Completeness and Boolean Satisfiability

The NP-Complete Class (2)

- More formally, $L_1 \subseteq \Sigma_1^*$ polynomially transforms to $L_2 \subseteq \Sigma_2^*$, written $L_1 \propto L_2$ or $L_1 \leq_p L_2$, iff $\exists f : \Sigma_1^* \rightarrow \Sigma_2^*$ such that
  - $\forall x \in \Sigma_1^*[x \in L_1 \Leftrightarrow f(x) \in L_2]$
  - There is a polynomial time TM that computes f
- *Transitivity* If $L_1 \propto L_2$ and $L_2 \propto L_3$ then $L_1 \propto L_3$
- If L is NP-Hard and $L \in$ P then P $=$ NP
- If L is NP-Complete, then $L \in$ P if and only if P $=$ NP
- If L_0 is NP-Complete and $L \in$ NP and $L_0 \propto L$ then L is NP-Complete
- Hence if we find one NP-Complete problem, it may become easier to find more
- In 1971/1973 Cook-Levin showed that the Boolean satisfiability problem (SAT) is NP-Complete

NP-Completeness and Boolean Satisfiability

The Boolean Satisfiability Problem

- A propositional logic formula or Boolean expression is built from variables, operators: AND (conjunction, $\wedge$), OR (disjunction, $\vee$), NOT (negation, $\neg$)
- A formula is said to be *satisfiable* if it can be made True by some assignment to its variables.
- *The Boolean Satisfiability Problem* is, given a formula, check if it is satisfiable.
  - *Instance:* a finite set U of Boolean variables and a finite set C of clauses over U
  - *Question:* Is there a satisfying truth assignment for C ?
- A *clause* is is a disjunction of variables or negations of variables
- *Conjunctive normal form (CNF)* is a conjunction of clauses
- Any Boolean expression can be transformed to CNF

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

NP-Completeness and Boolean Satisfiability

The Boolean Satisfiability Problem (2)

- Given a set of Boolean variable $U = \{u_1, u_2, \ldots, u_n\}$
- A literal from U is either any u_i or the negation of some u_i (written $\overline{u_i}$)
- A clause is denoted as a subset of literals from U — $\{u_2, \overline{u_4}, u_5\}$
- A clause is satisfied by an assignment to the variables if at least one of the literals evaluates to True (just like disjunction of the literals)
- Let C be a set of clauses over U — C is satisfiable iff there is some assignment of truth values to the variables so that every clause is satisfied (just like CNF)
- $C = \{\{u_1, u_2, u_3\}, \{\overline{u_2}, \overline{u_3}\}, \{u_2, \overline{u_3}\}\}$ is satisfiable
- $C = \{\{u_1, u_2\}, \{u_1, \overline{u_2}\}, \{\overline{u_1}\}\}$ is not satisfiable

NP-Completeness and Boolean Satisfiability

The Boolean Satisfiability Problem (3)

- Proof that SAT is NP-Complete looks at the structure of NDTMs and shows you can transform any NDTM to SAT in polynomial time (in fact logarithmic space suffices)
- SAT is in NP since you can check a solution in polynomial time
- To show that $\forall L \in \text{NP} : L \propto \text{SAT}$ invent a polynomial time algorithm for each polynomial time NDTM, M, which takes as input a string x and produces a Boolean formula E_x which is satisfiable iff M accepts x
- See Cook-Levin theorem

NP-Completeness and Boolean Satisfiability

Coping with NP-Completeness

- What does it mean if a problem is NP-Complete ?
  - There is a P time verification algorithm.
  - There is a P time algorithm to solve it iff P = NP (?)
  - No one has yet found a P time algorithm to solve any NP-Complete problem
  - So what do we do ?
- Improved exhaustive search — Dynamic Programming; Branch and Bound
- Heuristic methods — *acceptable* solutions in *acceptable* time — compromise on optimality
- Average time analysis — look for an algorithm with good average time — compromise on generality (see Big-O Algorithm Complexity Cheatsheet)
- Probabilistic or Randomized algorithms — compromise on correctness

M269 Specimen Exam

Q14 topics

- Unit 7
- Proofs
- Natural deduction

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

Logic

Logicians, Logics, Notations

- A plethora of logics, proof systems, and different notations can be puzzling.
- Martin Davis, Logician When I was a student, even the topologists regarded mathematical logicians as living in outer space. Today the connections between logic and computers are a matter of engineering practice at every level of computer organization
- Various logics, proof systems , were developed well before programming languages and with different motivations,

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

Logic

Logic and Programming Languages

- Turing machines, Von Neumann architecture and procedural languages Fortran, C, Java, Perl, Python, JavaScript
- Resolution theorem proving and logic programming — Prolog
- Logic and database query languages — SQL (Structured Query Language) and QBE (Query-By-Example) are syntactic sugar for first order logic
- Lambda calculus and functional programming with Miranda, Haskell, ML, Scala

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Logical Arguments

Validity and Justification

- There are two ways to model what counts as a logically good argument:
  - the **semantic** view
  - the **syntactic** view
- The notion of a valid argument in propositional logic is rooted in the semantic view.
- It is based on the semantic idea of interpretations: assignments of truth values to the propositional variables in the sentences under discussion.
- A *valid argument* is defined as one that preserves truth from the premises to the conclusions
- The syntactic view focuses on the syntactic form of arguments.
- Arguments which are correct according to this view are called *justified arguments*.

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

Logical Arguments

Proof Systems, Soundness, Completeness

- Semantic validity and syntactic justification are different ways of modelling the same intuitive property: whether an argument is logically good.
- A proof system is *sound* if any statement we can prove (justify) is also valid (true)
- A proof system is *adequate* if any valid (true) statement has a proof (justification)
- A proof system that is sound and adequate is said to be *complete*
- Propositional and predicate logic are *complete* — arguments that are valid are also justifiable and vice versa
- Unit 7 section 2.4 describes another logic where there are valid arguments that are not justifiable (provable)

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Logical Arguments
Valid arguments

- Unit 6 defines valid arguments with the notation $\dfrac{\begin{array}{c}P_1\\ \vdots\\ P_n\end{array}}{C}$
- The argument is *valid* if and only if the value of C is *True* in each interpretation for which the value of each premise P_i is *True* for $1 \leq i \leq n$
- In some texts you see the notation $\{P_1, \ldots, P_n\} \models C$
- The expression denotes a *semantic sequent* or *semantic entailment*
- The $\models$ symbol is called the *double turnstile* and is often read as *entails* or *models*
- In LaTeX $\vDash$ and $\models$ are produced from `\vDash` and `\models` — see also the *turnstile* package
- In Unicode $\models$ is called *TRUE* and is U+22A8, HTML ⊨

Logical Arguments

Valid arguments — Tautology

- The argument $\{\} \models C$ is valid if and only if C is *True* in all interpretations
- That is, if and only if C is a tautology
- Beware different notations that mean the same thing
  - Alternate symbol for empty set: $\emptyset \models C$
  - Null symbol for empty set: $\models C$
  - Original M269 notation with null axiom above the line: $\overline{C}$

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

Logic

Justified Arguments

- Definition 7.1 An argument $\{P_1, P_2, \ldots, P_n\} \vdash C$ is a justified argument if and only if either the argument is an instance of an axiom or it can be derived by means of an inference rule from one or more other justified arguments.
- Axioms

$$\boldsymbol{\Gamma} \cup \{\boldsymbol{A}\} \vdash \boldsymbol{A} \text{ (axiom schema)}$$

- This can be read as: any formula $\boldsymbol{A}$ can be derived from the assumption (premise) of $\{\boldsymbol{A}\}$ itself
- The $\vdash$ symbol is called the *turnstile* and is often read as *proves*, denoting *syntactic entailment*
- In LaTeX $\vdash$ is produced from `\vdash`
- In Unicode $\vdash$ is called *RIGHT TACK* and is U+22A2, HTML ⊢

Logic
Justified Arguments

- Section 2.3 of Unit 7 (not the Unit 6, 7 Reader) gives the inference rules for $\rightarrow$, $\wedge$, and $\vee$ — only dealing with *positive propositional logic* so not making use of negation — see List of logic systems
- Usually (Classical logic) have a functionally complete set of logical connectives — that is, every binary Boolean function can be expressed in terms the functions in the set

Justified Arguments

Inference Rules — Notation

- Inference rule notation:

$$\frac{Argument_1 \quad \ldots \quad Argument_n}{Argument}\ (label)$$

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Justified Arguments

Inference Rules — Conjunction

- $\dfrac{\boldsymbol{\Gamma} \vdash \boldsymbol{A} \quad \boldsymbol{\Gamma} \vdash \boldsymbol{B}}{\boldsymbol{\Gamma} \vdash \boldsymbol{A} \wedge \boldsymbol{B}}$ ($\wedge$-introduction)
- $\dfrac{\boldsymbol{\Gamma} \vdash \boldsymbol{A} \wedge \boldsymbol{B}}{\boldsymbol{\Gamma} \vdash \boldsymbol{A}}$ ($\wedge$-elimination left)
- $\dfrac{\boldsymbol{\Gamma} \vdash \boldsymbol{A} \wedge \boldsymbol{B}}{\boldsymbol{\Gamma} \vdash \boldsymbol{B}}$ ($\wedge$-elimination right)

Justified Arguments

Inference Rules — Implication

- $$\frac{\mathbf{\Gamma} \cup \{\boldsymbol{A}\} \vdash \boldsymbol{B}}{\mathbf{\Gamma} \vdash \boldsymbol{A} \rightarrow \boldsymbol{B}} \ (\rightarrow\text{-introduction})$$
- The above should be read as: If there is a proof (justification, inference) for $\boldsymbol{B}$ under the set of premises, $\mathbf{\Gamma}$, augmented with $\boldsymbol{A}$, then we have a proof (justification. inference) of $\boldsymbol{A} \rightarrow \boldsymbol{B}$, *under the unaugmented set of premises,* $\mathbf{\Gamma}$.
  The unaugmented set of premises, $\mathbf{\Gamma}$ may have contained $\boldsymbol{A}$ already so we cannot assume

  $$(\mathbf{\Gamma} \cup \{\boldsymbol{A}\}) - \{\boldsymbol{A}\} \text{ is equal to } \mathbf{\Gamma}$$
- $$\frac{\mathbf{\Gamma} \vdash \boldsymbol{A} \quad \mathbf{\Gamma} \vdash \boldsymbol{A} \rightarrow \boldsymbol{B}}{\mathbf{\Gamma} \vdash \boldsymbol{B}} \ (\rightarrow\text{-elimination})$$

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

Justified Arguments

Inference Rules — Disjunction

- $\dfrac{\mathbf{\Gamma} \vdash \boldsymbol{A}}{\mathbf{\Gamma} \vdash \boldsymbol{A} \vee \boldsymbol{B}}$ ($\vee$-introduction left)
- $\dfrac{\mathbf{\Gamma} \vdash \boldsymbol{B}}{\mathbf{\Gamma} \vdash \boldsymbol{A} \vee \boldsymbol{B}}$ ($\vee$-introduction right)
- Disjunction elimination

$$\frac{\mathbf{\Gamma} \vdash \boldsymbol{A} \vee \boldsymbol{B} \quad \mathbf{\Gamma} \cup \{\boldsymbol{A}\} \vdash \boldsymbol{C} \quad \mathbf{\Gamma} \cup \{\boldsymbol{B}\} \vdash \boldsymbol{C}}{\mathbf{\Gamma} \vdash \boldsymbol{C}} \ (\vee\text{-elimination})$$

- The above should be read: if a set of premises $\mathbf{\Gamma}$ justifies the conclusion $\boldsymbol{A} \vee \boldsymbol{B}$ and $\mathbf{\Gamma}$ augmented with each of $\boldsymbol{A}$ or $\boldsymbol{B}$ separately justifies $\boldsymbol{C}$, then $\mathbf{\Gamma}$ justifies $\boldsymbol{C}$

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

Proofs in Tree Form

- The syntax of proofs is recursive:
- A proof is either an axiom, or the result of applying a rule of inference to one, two or three proofs.
- We can therefore represent a proof by a tree diagram in which each node have one, two or three children
- For example, the proof of $\{P \land (P \to Q)\} \vdash Q$ in *Question 4* (in the Logic tutorial notes) can be represented by the following diagram:

$$\dfrac{\dfrac{\{P \land (P \to Q)\} \vdash P \land (P \to Q)}{\{P \land (P \to Q)\} \vdash P}\,(\land\text{-E left}) \qquad \dfrac{\{P \land (P \to Q)\} \vdash P \land (P \to Q)}{\{P \land (P \to Q)\} \vdash P \to Q}\,(\land\text{-E right})}{\{P \land (P \to Q)\} \vdash Q}\,(\to\text{-E})$$

M269
Phil Molyneux
M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Justified Arguments

Self-Assessment activity 7.4

- Let $\Gamma = \{P \to R, Q \to R, P \vee Q\}$
- $\dfrac{\Gamma \vdash P \vee Q \quad \Gamma \cup \{P\} \vdash R \quad \Gamma \cup \{Q\} \vdash R}{\Gamma \vdash R}$ ($\vee$-elimination)
- $\dfrac{\Gamma \cup \{P\} \vdash P \quad \Gamma \cup \{P\} \vdash P \to R}{\Gamma \cup \{P\} \vdash R}$ ($\to$-elimination)
- $\dfrac{\Gamma \cup \{Q\} \vdash Q \quad \Gamma \cup \{Q\} \vdash Q \to R}{\Gamma \cup \{Q\} \vdash R}$ ($\to$-elimination)
- Complete tree layout
- $$\dfrac{\Gamma \vdash P \vee Q \quad \dfrac{\begin{array}{l}\Gamma \cup \{P\} \\ \vdash P\end{array} \quad \begin{array}{l}\Gamma \cup \{P\} \\ \vdash P \to R\end{array}}{\Gamma \cup \{P\} \vdash R}(\to\text{-E}) \quad \dfrac{\begin{array}{l}\Gamma \cup \{Q\} \\ \vdash Q\end{array} \quad \begin{array}{l}\Gamma \cup \{Q\} \\ \vdash Q \to R\end{array}}{\Gamma \cup \{Q\} \vdash R}(\to\text{-E})}{\Gamma \vdash R}(\vee\text{-E})$$

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

Justified Arguments

Self-assessment activity 7.4 — Linear Layout

1.	$\{P \to R, Q \to R, P \lor Q\} \vdash P \lor Q$	[Axiom]
2.	$\{P \to R, Q \to R, P \lor Q\} \cup \{P\} \vdash P$	[Axiom]
3.	$\{P \to R, Q \to R, P \lor Q\} \cup \{P\} \vdash P \to R$	[Axiom]
4.	$\{P \to R, Q \to R, P \lor Q\} \cup \{Q\} \vdash Q$	[Axiom]
5.	$\{P \to R, Q \to R, P \lor Q\} \cup \{Q\} \vdash Q \to R$	[Axiom]
6.	$\{P \to R, Q \to R, P \lor Q\} \cup \{P\} \vdash R$	[2, 3, $\to$-E]
7.	$\{P \to R, Q \to R, P \lor Q\} \cup \{Q\} \vdash R$	[4, 5, $\to$-E]
8.	$\{P \to R, Q \to R, P \lor Q\} \vdash R$	[1, 6, 7, $\lor$-E]

M269 Exam 2013J

Q 14

- Consider the following axiom schema and rules:
- *Axiom schema:* $\{\boldsymbol{A}\} \vdash \boldsymbol{A}$
- *Rules: (as Unit 7 for Natural Deduction)*
  - $\wedge$-elimination left, $\wedge$-elimination right
  - $\wedge$-introduction
  - $\rightarrow$-introduction, $\rightarrow$-elimination
- Complete the following proof:

1.	$\{P \wedge (Q \wedge R)\} \vdash P \wedge (Q \wedge R)$	[Axiom]
2.	....	[1,$\wedge$-elimination left]
3.	$\emptyset \vdash (P \wedge (Q \wedge R)) \rightarrow P$	....

M269

Phil Molyneux

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J

Q 14 Solution

1.	$\{P \wedge (Q \wedge R)\} \vdash P \wedge (Q \wedge R)$	[Axiom]
2.	$\{P \wedge (Q \wedge R)\} \vdash P$	[1,$\wedge$-elimination left]
3.	$\emptyset \vdash (P \wedge (Q \wedge R)) \rightarrow P$	[2,$\rightarrow$-introduction]

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Specimen Exam

Q13 Topics

- Unit 6
- SQL queries

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J
Q 13

- A database contains the following tables, *oilfield* and *operator*

oilfield	
name	*production*
Warga	3
Lolli	5
Tolstoi	0.5
Dakhun	2
Sugar	3

operator	
company	*field*
Amarco	Warga
Bratape	Lolli
Rosbif	Tolstoi
Taqar	Dakhun
Bratape	Sugar

- Q 13 continued on next slide

M269
Phil Molyneux
M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J

Q 13 (continued)

- For each of the following SQL queries, give the table returned by the query

(a)

```
SELECT *
FROM operator;
```

(b)

```
SELECT name, production
FROM oilfield
WHERE production > 2;
```

(c)

```
SELECT name, production, company
FROM oilfield CROSS JOIN operator
WHERE name = field;
```


M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J

Q 13(a) Solution

(a) This is simply the whole *operator* table.

company	*field*
Amarco	Warga
Bratape	Lolli
Rosbif	Tolstoi
Taqar	Dakhun
Bratape	Sugar

M269

Phil Molyneux

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

(b) Retaining only the rows with *production* > 2

name	*production*
Warga	3
Lolli	5
Sugar	3

M269
Phil Molyneux
M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J

Q 13(c) Solution

(c) Joining the tables

name	*production*	*company*
Warga	3	Amarco
Lolli	5	Bratape
Tolstoi	0.5	Rosbif
Dakhun	2	Taqar
Sugar	3	Bratape

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Specimen Exam

Q12 Topics

- Unit 6
- Predicate Logic
- Translation to/from English
- Interpretations

M269 Exam 2013J
Q 12

- A particular interpretation of predicate logic allows facts to be expressed about films that people have seen, and of which they own copies.
- Some of the assignments in the interpretation are given below (where the symbol $\mathcal{I}$ is used to show assignment).
- The interpretation assigns *Jane*, *John* and *Saira* to the constants *jane*, *john* and *saira*.

$$
\begin{aligned}
\mathcal{I}(\text{jane}) &= \text{Jane} \\
\mathcal{I}(\text{john}) &= \text{John} \\
\mathcal{I}(\text{saira}) &= \text{Saira}
\end{aligned}
$$

- Q 12 continued on next slide

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J

Q 12

- The predicates *owns* and *has_seen* are assigned to binary relations. The comprehensions of the relations are:
  - $\mathcal{I}(\text{owns}) = \{(\text{A,B})\text{: the person A owns a copy of film B}\}$
  - $\mathcal{I}(\text{has\_seen}) = \{(\text{A,B})\text{: the person A has seen film B}\}$
- The enumerations of the relations are:

$$\begin{aligned}
\mathcal{I}(\text{owns}) &= \{(\text{Jane, Django}), (\text{Jane, Casablanca}), (\text{John, Jaws}), (\text{John, The Omen}), (\text{John, El Topo}), (\text{Saira, El Topo}), (\text{Saira, Casablanca})\} \\
\mathcal{I}(\text{has_seen}) &= \{(\text{Jane, Django}), (\text{Jane, Candide}), (\text{Jane, Casablanca}), (\text{John, The Omen}), (\text{John, El Topo}), (\text{Saira, Django}), (\text{Saira, The Omen})\}
\end{aligned}$$

- Q 12 continued on next slide

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J
Q 12

- Parts (a) and (b) of this question are on the next page.
- In both parts, you are given a sentence of predicate logic and asked to provide an English translation of the sentence in the box immediately following it.
- You also need to state whether the sentence is TRUE or FALSE in the interpretation that is provided on this page, and give an explanation of your answer.
- In your explanation you need to consider any relevant values for the variable X, *and show, using the interpretation above, whether it makes the quantified expression TRUE*.

- Q 12 continued on next slide

M269 Exam 2013J

Q 12

(a) $\forall X.\ (\text{owns(saira,X)} \rightarrow \text{has\_seen(saira,X)})$ can be translated in English as:

- This sentence is TRUE/FALSE because:

(b) $\exists X.(\text{has\_seen(jane,X)} \wedge \text{owns(jane,X)})$ can be translated in English as:

- This sentence is TRUE/FALSE because:

M269 Exam Revision Agenda & Aims

Units 6 & 7

Computability

M269 Exam 2013J Q 15

Complexity

NP-Completeness and Boolean Satisfiability

Logic

M269 Exam 2013J Q 14

SQL Queries

M269 Exam 2013J Q 13

Predicate Logic

M269 Exam 2013J Q 12

Propositional Logic

M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J

Q 12(a) Solution

(a) *For all films, if Saira owns a copy of the film, then Saira has seen the film.*
Or more idiomatically, *Saira has seen all of the films that she owns.*

- The sentence is *FALSE*, because the enumerations of the relations show that she owns a copy of *Casablanca*, but this is not one of the films that she has seen. She also owns a copy of *El Topo*, which she has not seen either, but we only need one counter-example to show that the sentence is false.

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J

Q 12(b) Solution

(b) *There exists a film, such that Jane has seen it and Jane owns it.*
Or more idiomatically, *Jane has seen at least one of the films that she owns*

- This sentence is *TRUE*. The enumerations show that she owns *Casablanca* and that she has seen it. *Django* also provides a sufficient example to show that the sentence is true.

M269 Specimen Exam

Q11 Topics

- Unit 6
- Sets
- Propositional Logic
- Truth tables
- Valid arguments
- Infinite sets

M269 Exam Revision Agenda & Aims

Units 6 & 7
- Computability
  - M269 Exam 2013J Q 15
- Complexity
  - NP-Completeness and Boolean Satisfiability
- Logic
  - M269 Exam 2013J Q 14
- SQL Queries
  - M269 Exam 2013J Q 13
- Predicate Logic
  - M269 Exam 2013J Q 12
- Propositional Logic
  - M269 Exam 2013J Q 11

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J
Q 11

(a) What does it mean to say that a well-formed formula (WFF) is *satisfiable* ?

(b) Is the following WFF satisfiable ?

$$(P \rightarrow (Q \rightarrow P)) \vee \neg R$$

Explain how you arrived at your answer

M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J

Q 11 Solution

(a) A WFF is *satisfiable* if it is possible to find an interpretation that makes the formula true.

(b) Truth table for the WFF

P	Q	R	$Q \to P$	$P \to (Q \to P)$	$\neg R$	$(P \to (Q \to P)) \vee \neg R$
F	F	F	T	T	T	T
F	F	T	T	T	F	T
F	T	F	F	T	T	T
F	T	T	F	T	F	T
T	F	F	T	T	T	T
T	F	T	T	T	F	T
T	T	F	T	T	T	T
T	T	T	T	T	F	T

- The truth table shows that the WFF $(P \to (Q \to P)) \vee \neg R$ is always true, so it satisfiable under any interpretation. But we don't need the whole truth table to prove this; the WFF is true for any interpretation in which R is false (for example).

M269
Phil Molyneux
M269 Exam Revision Agenda & Aims
Units 6 & 7
Computability
M269 Exam 2013J Q 15
Complexity
NP-Completeness and Boolean Satisfiability
Logic
M269 Exam 2013J Q 14
SQL Queries
M269 Exam 2013J Q 13
Predicate Logic
M269 Exam 2013J Q 12
Propositional Logic
M269 Exam 2013J Q 11
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Specimen Exam

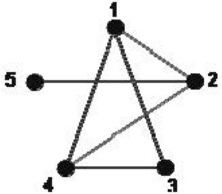
Unit 5 Topics, Q9, Q10

- Unit 5 Optimisation
- Graphs searching: DFS, BFS
- Distance: Dijkstra's algorithm
- Greedy algorithms: Minimum spanning trees, Prim's algorithm
- Dynamic programming: Knapsack problem, Edit distance

M269 Exam 2013J

Q 10

- Consider the following graph:



- Complete the table below to show the order in which the vertices of the above graph could be visited in a Depth First Search (DFS) starting at vertex 3 and always choosing first the leftmost not yet visited vertex (as seen from the current vertex):

Vertex	3				

M269
Phil Molyneux
M269 Exam Revision Agenda & Aims
Units 6 & 7
Units 3, 4 & 5
Unit 5 Optimisation
M269 Exam 2013J Q 10
M269 Exam 2013J Q 9
Unit 4 Searching
M269 Exam 2013J Q 8
M269 Exam 2013J Q 7
Unit 3 Sorting
M269 Exam 2013J Q 6
M269 Exam 2013J Q 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J

Q 10 Solution

- Depth First Search (DFS) starting at vertex 3 and always choosing first the leftmost not yet visited vertex (as seen from the current vertex):

Vertex	3	4	1	2	5

- Notice the ambiguity about the term *leftmost* — an alternative view could have been:

Vertex	3	1	4	2	5

M269 Exam 2013J
Q 9

- Recall that the structured English for Dijkstra's algorithm is:

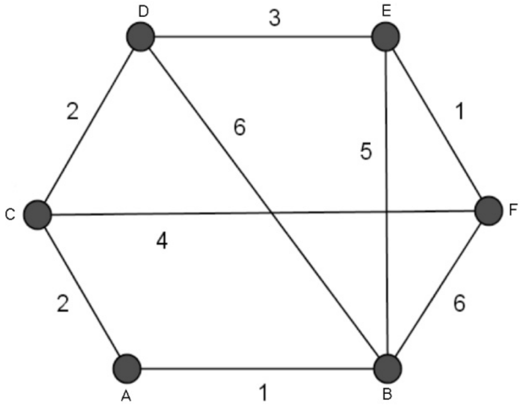
```
create priority~queue
set dist to 0 for v and dist to infinity
  for all other vertices
add all vertices to priority~queue
ITERATE while priority~queue is not empty
  remove u from the front of the queue
  ITERATE over w in the neighbours of u
    set new~distance to
      dist u + length of edge from u to w
    IF new~distance is less than dist w
      set dist w to new~distance
      change priority(w, new~distance)
```

- Q 9 continued on next slide

M269 Exam 2013J

Q 9

- Now consider the following weighted graph:



- Q 9 continued on next slide

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Unit 5 Optimisation

M269 Exam 2013J Q 10

M269 Exam 2013J Q 9

Unit 4 Searching

M269 Exam 2013J Q 8

M269 Exam 2013J Q 7

Unit 3 Sorting

M269 Exam 2013J Q 6

M269 Exam 2013J Q 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J
Q 9

- **Starting from vertex B**, the following table represents the distances after the second line of structured English is executed for the graph given above (using the convention that a blank cell represents infinity):

Vertex	A	B	C	D	E	F
Distance		0				

- Now, complete the appropriate boxes in the next table to show the distances after the first and second iterations of the while loop of the algorithm.

Vertex	A	B	C	D	E	F	
Distance		0					First iteration
Distance		0					Second iteration

M269 Exam 2013J

Q 9 Solution

- The completed table:

Vertex	A	B	C	D	E	F	
Distance	1	0		6	5	6	First iteration
Distance	1	0	3	6	5	6	Second iteration

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Unit 5 Optimisation

M269 Exam 2013J Q 10

M269 Exam 2013J Q 9

Unit 4 Searching

M269 Exam 2013J Q 8

M269 Exam 2013J Q 7

Unit 3 Sorting

M269 Exam 2013J Q 6

M269 Exam 2013J Q 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

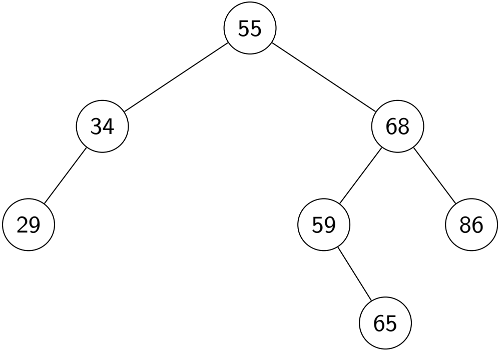
M269 Specimen Exam

Unit 4 Topics, Q7, Q8

- Unit 4 Searching
- String searching: Quick search Sunday algorithm, Knuth-Morris-Pratt algorithm
- Hashing and hash tables
- Search trees: Binary Search Trees
- Search trees: Height balanced trees: AVL trees

M269 Exam 2013J
Q 8

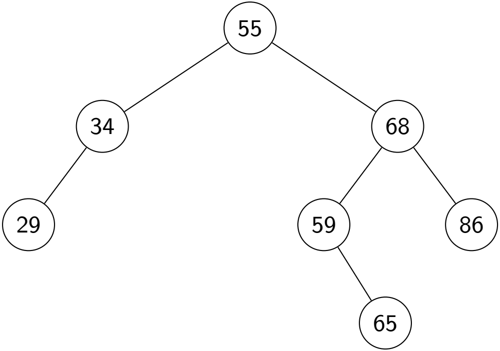
(a) Consider the following Binary Search Tree.



- Modify (draw on) the above Binary Search Tree to insert a node with a key of 57.
- Q 8 continued on next slide

M269 Exam 2013J
Q 8

(b) Once again, consider the same Binary Search Tree.



- Calculate the balance factors of each node in the tree above and modify the diagram to show these balance factors.

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Unit 5 Optimisation

M269 Exam 2013J Q 10

M269 Exam 2013J Q 9

Unit 4 Searching

M269 Exam 2013J Q 8

M269 Exam 2013J Q 7

Unit 3 Sorting

M269 Exam 2013J Q 6

M269 Exam 2013J Q 5

Units 1 & 2

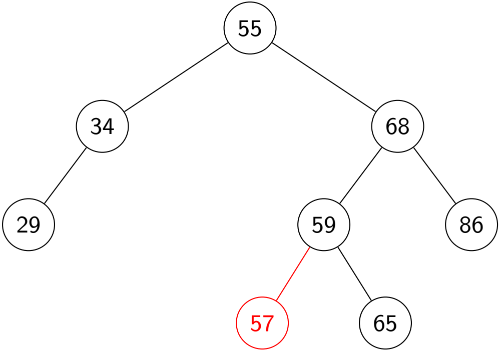
M269 Exam Section 2

Exam Techniques

M269 Exam 2013J

Q 8(a) Solution

(a) Answer, with inserted node shown in red



M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Unit 5 Optimisation

M269 Exam 2013J Q 10

M269 Exam 2013J Q 9

Unit 4 Searching

M269 Exam 2013J Q 8

M269 Exam 2013J Q 7

Unit 3 Sorting

M269 Exam 2013J Q 6

M269 Exam 2013J Q 5

Units 1 & 2

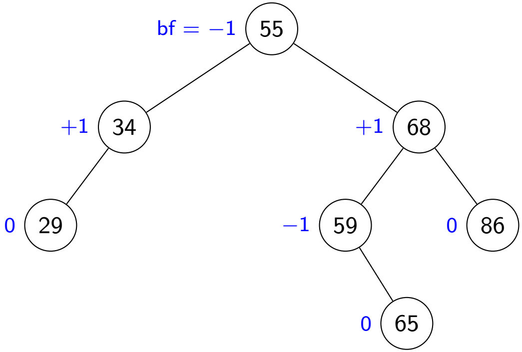
M269 Exam Section 2

Exam Techniques

M269 Exam 2013J

Q 8(b) Solution

- Answer, with balance factors shown in blue



M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Unit 5 Optimisation

M269 Exam 2013J Q 10

M269 Exam 2013J Q 9

Unit 4 Searching

M269 Exam 2013J Q 8

M269 Exam 2013J Q 7

Unit 3 Sorting

M269 Exam 2013J Q 6

M269 Exam 2013J Q 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J

Q 7

- In the KMP algorithm, for each character in the target string T we identify the longest substring of T ending with that character which matches a prefix of the target string.
- These lengths are stored in what is known as a **prefix table** (which in Unit 4 we represented as a list).
- Consider the target string T

A	B	A	C	A	B	A	C	A	C

- Below is an incomplete prefix table for the target string given above. Complete the prefix table by writing the missing numbers in the appropriate boxes.

0		1	0		2		4		0

M269 Exam 2013J

Q 7 Solution

- The complete prefix table, with new entries in red:

0	0	1	0	1	2	3	4	5	0

- Here is the target, prefix and shift:

	A	B	A	C	A	B	A	C	A	C	Target string, t
	0	1	2	3	4	5	6	7	8	9	Position (Index), p
0	1	2	3	4	5	6	7	8	9	10	Match, q
	0	0	1	0	1	2	3	4	5	0	$\text{prefixTable}(t, p)$
1	1	2	2	4	4	4	4	4	4	10	$\text{shift}(t, q)$

- The shift function takes the *target string*, t, and the number of characters matched, q.
- $\text{shift}(t, 0) = 1$
- $\text{shift}(t, q) = q - \text{prefixTable}(t, q - 1)$

M269
Phil Molyneux
M269 Exam Revision Agenda & Aims
Units 6 & 7
Units 3, 4 & 5
Unit 5 Optimisation
M269 Exam 2013J Q 10
M269 Exam 2013J Q 9
Unit 4 Searching
M269 Exam 2013J Q 8
M269 Exam 2013J Q 7
Unit 3 Sorting
M269 Exam 2013J Q 6
M269 Exam 2013J Q 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Specimen Exam

Unit 3 Topics, Q5, Q6

- Unit 3 Sorting
- Elementary methods: Bubble sort, Selection sort, Insertion sort
- Recursion — base case(s) and recursive case(s) on smaller data
- Quicksort, Merge sort
- Sorting with data structures: Tree sort, Heap sort
- See sorting notes for abstract sorting algorithm

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Unit 5 Optimisation

M269 Exam 2013J Q 10

M269 Exam 2013J Q 9

Unit 4 Searching

M269 Exam 2013J Q 8

M269 Exam 2013J Q 7

Unit 3 Sorting

M269 Exam 2013J Q 6

M269 Exam 2013J Q 5

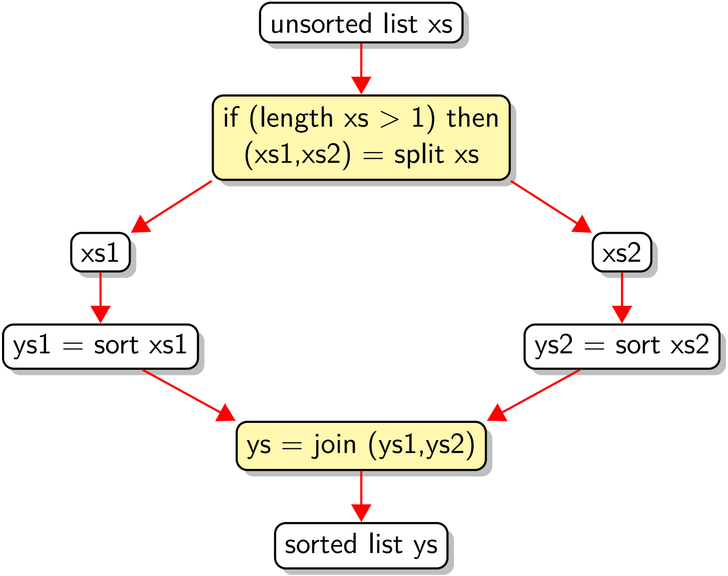
Units 1 & 2

M269 Exam Section 2

Exam Techniques

Unit 3 Sorting

Abstract Sorting Algorithm



M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Unit 5 Optimisation

M269 Exam 2013J Q 10

M269 Exam 2013J Q 9

Unit 4 Searching

M269 Exam 2013J Q 8

M269 Exam 2013J Q 7

Unit 3 Sorting

M269 Exam 2013J Q 6

M269 Exam 2013J Q 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

Unit 3 Sorting

Sorting Algorithms

Using the *Abstract sorting algorithm*, describe the *split* and *join* for:

- Insertion sort
- Selection sort
- Merge sort
- Quicksort
- Bubble sort (the odd one out)

M269 Exam 2013J
Q 6

- Consider the following function, which takes a list as an argument. You may assume that the list contains a number of integer values and is not empty.

```
def average(aList):
    n = len(aList)
    total = 0
    for item in aList:
        total = total + item
    mean = total / n
return mean
```

- Q 6 continued on next slide

M269 Exam 2013J

Q 6

- From the five options below, select the **one** that represents the correct combination of $T(n)$ and Big-O complexity for this function. You may assume that a step (i.e. the basic unit of computation) is the assignment statement.

A. $T(n) = 3 + n^2$ and $O(n^2)$

B. $T(n) = n + 2$ and $O(n^2)$

C. $T(n) = 2n + 2$ and $O(n)$

D. $T(n) = 3n + n^2$ and $O(n^2)$

E. $T(n) = n + 3$ and $O(n)$

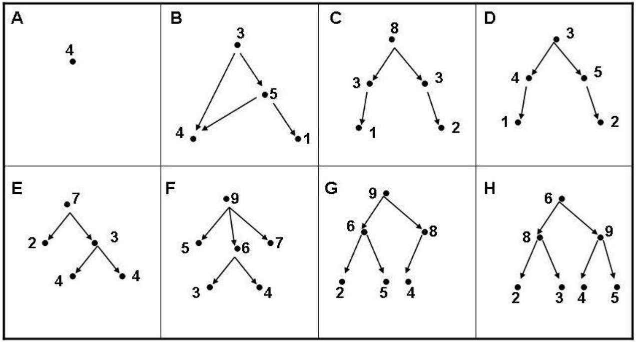
M269 Exam 2013J
Q 6 Solution

- Option E is correct.
- The function does three assignments once per call, and one assignment for each of the n items in the argument, hence $T(n) = n + 3$.

M269 Exam Revision Agenda & Aims
Units 6 & 7
Units 3, 4 & 5
Unit 5 Optimisation
M269 Exam 2013J Q 10
M269 Exam 2013J Q 9
Unit 4 Searching
M269 Exam 2013J Q 8
M269 Exam 2013J Q 7
Unit 3 Sorting
M269 Exam 2013J Q 6
M269 Exam 2013J Q 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J
Q 5

- Consider the following diagrams A–H. Nodes are represented by black dots and edges by arrows. The numbers represent a node's key.



- Q 5 continued on next slide

M269
Phil Molyneux
M269 Exam Revision Agenda & Aims
Units 6 & 7
Units 3, 4 & 5
Unit 5 Optimisation
M269 Exam 2013J Q 10
M269 Exam 2013J Q 9
Unit 4 Searching
M269 Exam 2013J Q 8
M269 Exam 2013J Q 7
Unit 3 Sorting
M269 Exam 2013J Q 6
M269 Exam 2013J Q 5
Units 1 & 2
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J
Q 5

- Answer the following questions. Write your answer on the line that follows each question. In each case there is at least one diagram in the answer but there may be more than one. Explanations are not required.

(a) Which of A, B, C and D **do not** show trees ?

(b) Which of E, F, G and H are binary trees ?

(c) Which of C, D, G and H are complete binary trees ?

(d) Which of C, D, G and H are heaps ?

M269 Exam 2013J

Q 5 Solution

(a) B is not a tree; it has more than one route from node 3 to node 4.

(b) E, G, and H are binary trees; (no more than 2 children per node).

(c) G, and H are complete binary trees.

(d) Only G is a heap; (complete binary tree, and parent nodes $>$ children).

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Unit 5 Optimisation

M269 Exam 2013J Q 10

M269 Exam 2013J Q 9

Unit 4 Searching

M269 Exam 2013J Q 8

M269 Exam 2013J Q 7

Unit 3 Sorting

M269 Exam 2013J Q 6

M269 Exam 2013J Q 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques

M269 Specimen Exam

Unit 2 Topics, Q3, Q4

- Unit 2 From Problems to Programs
- Abstract Data Types
- Pre and Post Conditions
- Logic for loops

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J

Q 4

- Consider the guard in the following Python while loop header:

```
while (a < 6 and b > 8) or not(a >= 6 or b <= 8):
```

(a) Make the following substitutions:

P represents `a < 6`
Q represents `b > 8`

Then complete the following truth table:

P	Q	$\neg P$	$\neg Q$	$P \wedge Q$	$\neg P \vee \neg Q$	$\neg(\neg P \vee \neg Q)$	$(P \wedge Q) \vee \neg(\neg P \vee \neg Q)$
F	F						
F	T						
T	F						
T	T						

- Q 4 continued on next slide

M269 Exam Revision Agenda & Aims
Units 6 & 7
Units 3, 4 & 5
Units 1 & 2
Unit 2 From Problems to Programs
M269 Exam 2013J Q 4
M269 Exam 2013J Q 3
Example Algorithm Design — Searching
Unit 1 Introduction
M269 Exam 2013J Q 2
M269 Exam 2013J Q 1
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J

Q 4

(b) Use the results from your truth table to choose which one of the following expressions could be used as the simplest equivalent to the above guard.

A. `(a < 6 and b > 8)`
B. `not(a < 6 and b > 8)`
C. `(a >= 6 or b <= 8)`
D. `(a >= 6 and b <= 8)`
E. `(a < 6 and b <= 8)`

M269 Exam Revision Agenda & Aims
Units 6 & 7
Units 3, 4 & 5
Units 1 & 2
Unit 2 From Problems to Programs
M269 Exam 2013J Q 4
M269 Exam 2013J Q 3
Example Algorithm Design — Searching
Unit 1 Introduction
M269 Exam 2013J Q 2
M269 Exam 2013J Q 1
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J

Q 4 Solution

(a) The completed truth table:

P	Q	$\neg P$	$\neg Q$	$P \wedge Q$	$\neg P \vee \neg Q$	$\neg(\neg P \vee \neg Q)$	$(P \wedge Q) \vee \neg(\neg P \vee \neg Q)$
F	F	T	T	F	T	F	F
F	T	T	F	F	T	F	F
T	F	F	T	F	T	F	F
T	T	F	F	T	F	T	T

(b) A is the simplest equivalent of the guard given.

M269 Exam 2013J
Q 3

- A binary search is being carried out on the list shown below for item 67:

  `[12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]`

- For each pass of the algorithm, draw a box around the items in the partition to be searched during that pass, continuing for as many passes as you think are needed.
- We have done the first pass for you showing that the search starts with the whole list. Draw your boxes below for each pass needed; you may not need to use all the lines below. *(The question had 8 rows)*

(Pass 1) [$\boxed{\texttt{12,16,17,24,41,49,51,62,67,69,75,80,89,97,101}}$]
(Pass 2) `[12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]`
(Pass 3) `[12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]`

M269 Exam 2013J
Q 3 Solution

- The complete binary search:

  (Pass 1) [$\boxed{\texttt{12,16,17,24,41,49,51,62,67,69,75,80,89,97,101}}$]

  (Pass 2) [12,16,17,24,41,49,51,62, $\boxed{\texttt{67,69,75,80,89,97,101}}$]

  (Pass 3) [12,16,17,24,41,49,51,62, $\boxed{\texttt{67,69,75}}$,80,89,97,101]

  (Pass 4) [12,16,17,24,41,49,51,62, $\boxed{\texttt{67}}$,69,75,80,89,97,101]

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

Example Algorithm Design

Searching

- Given an ordered list (`xs`) and a value (`val`), return
  - Position of `val` in `xs` or
  - Some indication if `val` is not present
- *Simple strategy:* check each value in the list in turn
- *Better strategy:* use the ordered property of the list to reduce the range of the list to be searched each turn
  - Set a range of the list
  - If `val` equals the mid point of the list, return the mid point
  - Otherwise half the range to search
  - If the range becomes negative, report not present (return some distinguished value)

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

Example Algorithm Design

Binary Search Iterative

```python
def binarySearchIter(xs, val):
  lo = 0
  hi = len(xs) - 1

  while lo <= hi:
    mid = (lo + hi) // 2
    guess = xs[mid]

    if val == guess:
      return mid
    elif val < guess:
      hi = mid - 1
    else:
      lo = mid + 1

  return None
```


M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

Divide and Conquer

Binary Search Recursive

```
def binarySearchRec(xs, val, lo=0, hi=-1):
  if (hi == -1):
    hi = len(xs) - 1

  mid = (lo + hi) // 2

  if hi < lo:
    return None
  else:
    guess = xs[mid]
    if val == guess:
      return mid
    elif val < guess:
      return binarySearchRec(xs, val, lo, mid-1)
    else:
      return binarySearchRec(xs, val, mid+1, hi)
```

Divide and Conquer

Binary Search Recursive — Solution

```
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs, 67)
xs = Highlight the mid value and search range
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
Return value: ??
```

Divide and Conquer

Binary Search Recursive — Solution

```
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs, 67)
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
Return value: ??
```


M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

Divide and Conquer

Binary Search Recursive — Solution

```
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs, 67)
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,14) by line 15
xs = Highlight the mid value and search range
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
Return value: ??
```


M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

Divide and Conquer

Binary Search Recursive — Solution

```
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs, 67)
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,14) by line 15
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
Return value: ??
```


M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

Divide and Conquer

Binary Search Recursive — Solution

```
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs, 67)
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,14) by line 15
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,10) by line 13
xs = Highlight the mid value and search range
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
Return value: ??
```


M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

Divide and Conquer

Binary Search Recursive — Solution

```
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs, 67)
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,14) by line 15
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,10) by line 13
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,25,??,??)
xs = Highlight the mid value and search range
Return value: ??
```


M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

Divide and Conquer

Binary Search Recursive — Solution

```
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs, 67)
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,14) by line 15
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,10) by line 13
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,8) by line 13
xs = Highlight the mid value and search range
Return value: ??
```


M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

Divide and Conquer

Binary Search Recursive — Solution

```
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs, 67)
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,14) by line 15
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,10) by line 13
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,8) by line 13
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
```

Return value: ??

Divide and Conquer

Binary Search Recursive — Solution

```
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs, 67)
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,14) by line 15
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,10) by line 13
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
binarySearchRec(xs,67,8,8) by line 13
xs = [12,16,17,24,41,49,51,62,67,69,75,80,89,97,101]
Return value: 8 by line 11
```


M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

Example Algorithm Design

Binary Search Iterative — Miller & Ranum

```
def binarySearchIterMR(alist, item):
  first = 0
  last = len(alist)-1
  found = False

  while first<=last and not found:
    midpoint = (first + last)//2
    if alist[midpoint] == item:
      found = True
    else:
      if item < alist[midpoint]:
        last = midpoint-1
      else:
        first = midpoint+1

  return found
```

Divide and Conquer

Binary Search Recursive — Miller & Ranum

```
def binarySearchRecMR(alist, item):
  if len(alist) == 0:
    return False
  else:
    midpoint = len(alist)//2
    if alist[midpoint]==item:
      return True
    else:
      if item<alist[midpoint]:
        return binarySearchRecMR(alist[:midpoint],item)
      else:
        return binarySearchRecMR(alist[midpoint+1:],item)
```


M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

M269 Specimen Exam

Unit 1 Topics, Q1, Q2

- Unit 1 Introduction
- Computation, computable, tractable
- Introducing Python
- What are the three most important concepts in programming ?
  1.
  2.
  3.
- Quote from Paul Hudak (1952–2015)

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

M269 Specimen Exam

Unit 1 Topics, Q1, Q2

- Unit 1 Introduction
- Computation, computable, tractable
- Introducing Python
- What are the three most important concepts in programming ?
    1. Abstraction
    2.
    3.
- Quote from Paul Hudak (1952–2015)

M269 Specimen Exam

Unit 1 Topics, Q1, Q2

- Unit 1 Introduction
- Computation, computable, tractable
- Introducing Python
- What are the three most important concepts in programming ?
  1. Abstraction
  2. Abstraction
  3.
- Quote from Paul Hudak (1952–2015)

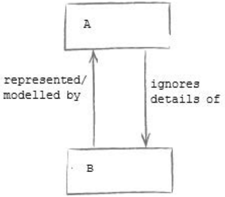
M269 Specimen Exam

Unit 1 Topics, Q1, Q2

- Unit 1 Introduction
- Computation, computable, tractable
- Introducing Python
- What are the three most important concepts in programming ?
  1. Abstraction
  2. Abstraction
  3. Abstraction
- Quote from Paul Hudak (1952–2015)

M269 Exam 2013J
Q 2

- The general idea of abstraction as modelling can be shown with the following diagram.



- Complete the diagram above by adding an appropriate label (one of the numbers 1 to 4) in the space indicated by **A** and one in the space indicated by **B**. The possible answers are shown as 1 to 4 below. *The exam question had some pictures next to the texts*
  1. A car crash test dummy in the real world
  2. An action man doll in the real world
  3. A real car in the real world (after crashing)
  4. A real driver in the real world

M269 Exam Revision Agenda & Aims
Units 6 & 7
Units 3, 4 & 5
Units 1 & 2
Unit 2 From Problems to Programs
M269 Exam 2013J Q 4
M269 Exam 2013J Q 3
Example Algorithm Design — Searching
Unit 1 Introduction
M269 Exam 2013J Q 2
M269 Exam 2013J Q 1
M269 Exam Section 2
Exam Techniques

M269 Exam 2013J

Q 2 Solution

- A real driver is modelled by a car crash test dummy, so A = 1 and B = 4

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J

Q 1

- Which **two** of the following statements are **true**?

A. A decision problem is any problem stated in a formal language.

B. A computational problem is a problem that is expressed sufficiently precisely that it is possible to build an algorithm that will solve all instances of that problem.

C. An algorithm consists of a precisely stated, step-by-step list of instructions.

D. Computational thinking is the skill to formulate a problem as a computational problem, and then construct a good computational solution, in the form of an algorithm, to solve this problem, or explain why there is no such solution.

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J

Q 1 Solution

- Options C and D are true.
- Option A is wrong because decision problems have to have a yes-no answer.
- Option B is wrong because there are computational problems that we can state and build algorithms for, but cannot always be solved.

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

Unit 2 From Problems to Programs

M269 Exam 2013J Q 4

M269 Exam 2013J Q 3

Example Algorithm Design — Searching

Unit 1 Introduction

M269 Exam 2013J Q 2

M269 Exam 2013J Q 1

M269 Exam Section 2

Exam Techniques

M269 Exam 2013J
Q 16

- Multipart question
- Specification of program, data structures, pre and post conditions
- Write a small program
- Give the complexity of the small program
- Give insight into a sorting algorithm
- Give insight into insertion into a binary search tree
- See notes version for text

M269 Exam 2013J

Q 17

- Write short report on a computational topic
- Suitable title for the topic and audience
- Paragraph setting the scene — the context of the topic
- Paragraph describing the topic
- Paragraph on the role the topic plays in some area
- Conclusions justifying the importance of the topic
- See notes version for text.

M269 Exam Revision Agenda & Aims
Units 6 & 7
Units 3, 4 & 5
Units 1 & 2
M269 Exam Section 2
M269 Exam 2013J Q 16
M269 Exam 2013J Q 16 Text
M269 Exam 2013J Q 17
M269 Exam 2013J Q 17 Text
Exam Techniques

M269 Exam
Exam Techniques

- Surviving in a time of great stress
- Each give one exam tip to the group
- TODO: add some more points

M269 Exam Revision Agenda & Aims

Units 6 & 7

Units 3, 4 & 5

Units 1 & 2

M269 Exam Section 2

Exam Techniques