

GEM 1501 Problem Solving With Computers

Lecture 1:

What Problems?

Frank Stephan

Overview of this Lecture

- **Administrative matters**
- Course overview
- What problems?
- What computers?
- Problem solving (with and without computers)
- Goals of the course

Timetable

- The lecture is 2 hours per week on Wednesdays, 12.00-14.00h in LT34 in SoC1.
- Laboratory sessions will be held in small groups. The schedule is Fridays, the groups are Fr 12.00-14.00h and Fr 14.00-16.00h. The room is Programming Lab 4 (COM1#B-11). The laboratory sessions start in Week 3 (30.01.2009).
- This course carries 100 marks which are distributed on midterm tests (12 marks each), the assignments (26 marks) and the final exam (50 marks).
- The two midterm examinations will be held at 18.02.2009 and 08.04.2009 on each day in the first half hour of the lecture.
- One assignment per laboratory session can have help.
1 assignment = 1 mark, 2 assignments = 2 marks, ..., 24 assignments = 24 marks, all 28 assignments = 26 marks.

Administrative Matters

- Course web page <http://www.comp.nus.edu.sg/~gem1501>
- No tutorials but labs; register, once the system is up; bring Student Card for entering laboratories
- If possible, create your SoC UNIX account some days prior going to the laboratory.
- IVLE (keep checking for announcements)
- Assignments (discussed in labs)
- Discussion forums (IVLE)
- Assignments can be solved during laboratory sessions
- Be on time for lectures and lab sessions!

Overview of this Lecture

- Administrative matters
- **Course overview**
- What problems?
- What computers?
- Problem solving (with and without computers)
- Goals of the course

GEM1501

- Goal: Introduction to **problem solving** with **computers**
- Method: Principled approach, augmented with practical programming exercises
- Emphasis: Possibilities and limitations of computers
- General outline: Harel and Feldman, Algorithmics
- Programming Language: Java Script

Content of GEM1501

- What problems?
- Algorithms and data
- Programming languages
- Algorithmic methods
- Correctness and efficiency
- Limitations of computers

Algorithmic Methods

- Divide-and-conquer
Split problems in subproblems
- Greedy algorithms
Maximize profit in each step for a good performance
- Dynamic programming
A more involved method to find the solution
- Traversal/search
Searching in large data bases and finding solutions to mathematical questions

Limitations of Computers

- Intractable problems
Can be solved, but only in millions of years
- NP-complete problems
Unknown whether easy or difficult
- Excursion: Public-key encryption
Usage of difficult problems for coding
- Undecidable problems
Problems which cannot be solved at all

Overview of this Lecture

- Administrative matters
- Course overview
- **What problems?**
- What computers?
- Problem solving (with and without computers)
- Goals of the course

What Problems?

Definitions of Problems

- Etymology
- Three definitions
- Getting closer

Etymology

- Middle English *probleme*, from Middle French, from Latin *problema*, from Greek *problēma*,
- Literally: obstacle,
from Greek *proballein* to throw forward,
from *pro-* forward + *ballein* to throw

Three Definitions of “Problem”

<http://dictionary.com>:

- A misgiving, objection or complaint: *I have a problem with his cynicism.*
- A situation, matter, or person that presents perplexity or difficulty: was having problems breathing; *considered the main problem to be his boss.*
- A question to be considered, solved, or answered: math problems; the problem of how to arrange transportation.

Definition 1: A misgiving, objection ...

“A misgiving, objection or complaint: I have a problem with his cynicism.”

- Refers to the subjective perception of a situation
- Often *perception* is the main issue
- Problem solving typically involve “emotional” skill
- Aspects of such problems may or may not be able to be addressed with the help of computers.

Definition 2: A situation,... that presents perplexity...

“A situation, matter or person that presents perplexity or difficulty: was having problems breathing; *considered the main problem to be his boss.*”

- refers to practical aspects of daily life
- computers don't solve these problems, but may contribute
- Examples: management tools, spell-checking, pass-times

Not the focus here, but aspects of this type of problems may come up during the course

Definition 3: A question to be considered, solved...

”A question to be considered, solved, or answered: math problems; the problem of how to arrange transportation.”

- In math problem example, the teacher “throws forward” problems to the student in order to achieve a didactic objective (computers may be involved as tools)
- Transportation problem example: how to visit 25 cities on shortest route? (Travelling Salesman Problem)

Gauss' Story

- Teacher to 8 year old pupils:

“Compute the sum $1 + 2 + 3 + \dots + 100$:

$$1 + 2 = 3,$$

$$1 + 2 + 3 = 3 + 3 = 6,$$

$$1 + 2 + 3 + 4 = 6 + 4 = 10,$$

$$1 + 2 + 3 + 4 + 5 = 10 + 5 = 15, \dots”$$

- So pupils are busy and teacher has easy job. But Gauss solved that problem fast and impressed the teacher.
- <http://www.nzmaths.co.nz/HelpCentre/Seminars/Gauss.htm>

Gauss' Solution

5		1	$5 + 1 = 6;$
4		2	$4 + 2 = 6;$
3		3	$3 + 3 = 6;$
2		4	$2 + 4 = 6;$
1		5	$1 + 5 = 6;$

$$1 + 2 + 3 + 4 + 5 = \frac{1}{2} \cdot 5 \cdot 6 = 15;$$

$$1 + 2 + 3 + \dots + 100 = \frac{1}{2} \cdot 100 \cdot 101 = 5050.$$

Carl Friedrich Gauss (1777-1855) lived in Germany and worked at the University of Göttingen.

Our Definition of “Problems”

A **relevant problem** is a technical question in a practical (business, scientific, engineering) domain, whose solution benefits a person or an organization.

An **academic problem** is a formalization of a relevant problem studied up to the last detail.

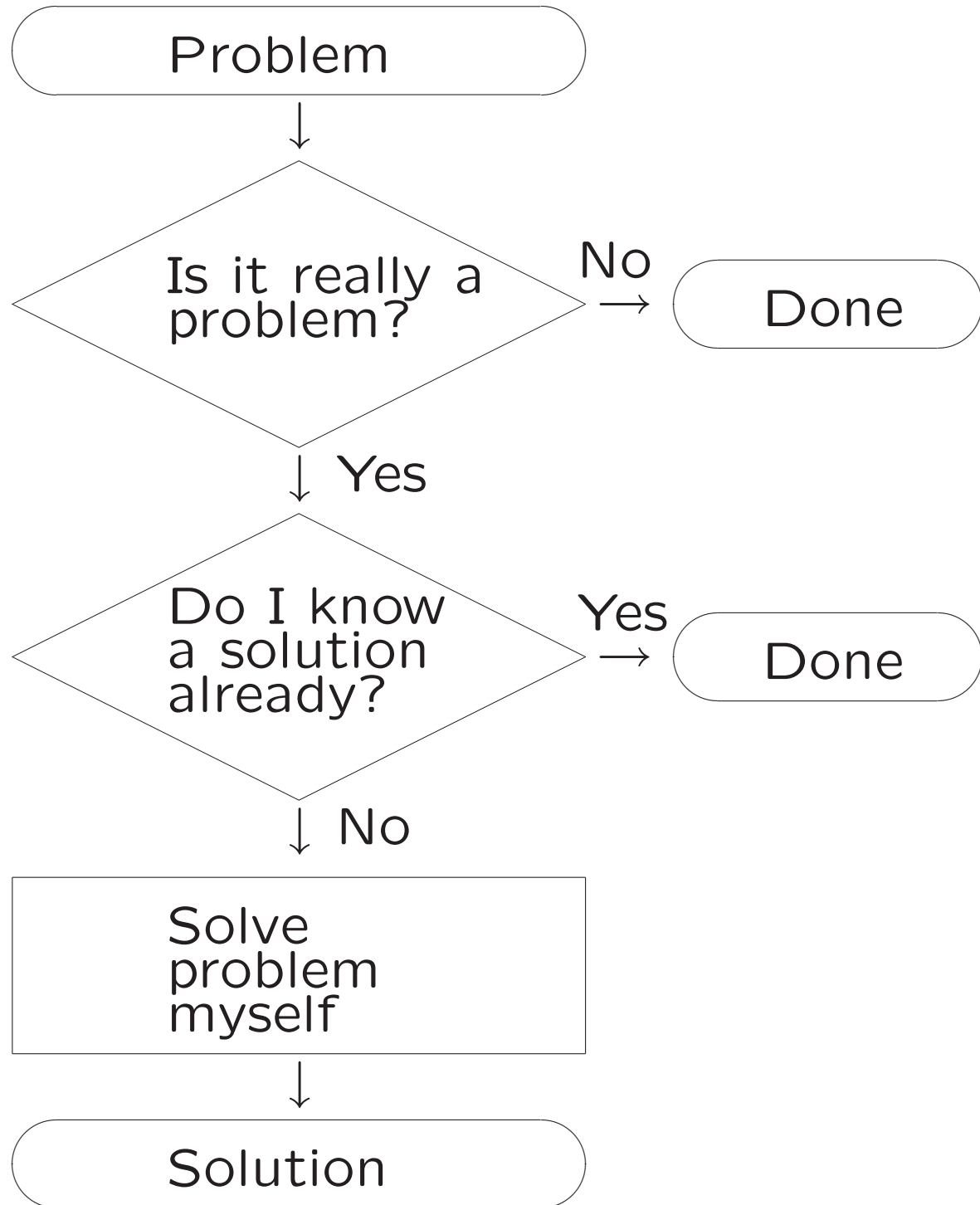
See <http://www.tsp.gatech.edu/sweden/index.html> for the shortest possible round tour through all 24978 cities in Sweden.

Is It Really a Problem?

- “A problem is a gap between what is actual and what is ideal.” For example, actual versus shortest tour.
- What is “ideal” in a given situation is in the eye of the beholder? Is shortest tour most convenient one?
- The perception of the “ideal” lies at the heart of all problems, even technical problems.

Do I know
a solution
already?

Solve a problem only
if it is really necessary
and the solution
is not yet known



Overview of this Lecture

- Administrative matters
- Course overview
- What problems?
- **What computers?**
- Problem solving (with and without computers)
- Goals of the course

What Computers?

- Computers are everywhere!
- What are computers?
- How do they solve problems?

Immersed in Computers

- Computers are ubiquitous
- You are using computers all the time
- Most of the time you don't notice it
- You are not aware of the problem solving
- Many things which did not have computers 20 years ago do have them built in now (clocks, televisions, washing machines, cars, ...)

Examples of Ubiquitous Computing

- Spell-checking
- Web-surfing
- Game playing
- Mobile telephony
- ATM banking
- Theft Protection

Advanced Computer Applications

- Airtraffic control
- Animating movies like “The Lord of the Rings”
- Controlling nuclear power plants
- Interplanetary voyages
- Designing computers!

What Are Computers?

- States, realized with electric/magnetic fields
- A “bit” can take two states, let us call them “0” and “1”
- Examples of operations on bits
 - Flipping (from “0” to “1” or vice versa)
 - Zeroing bits (setting a bit to “0”)
 - Testing bits (flip a bit if another bit is “1”)

Combining Bits

Eight bits are one byte: 01100100 is Hundred

Binary numbers can be added similar to decimals

01100100	100	10000000	128
+ 00011001	+ 25	- 00000011	- 3
= 01111101	= 125	= 01111101	= 125

Nonnumerical information can also be coded (ASCII)

"A" is 01000001 "Z" is 01011010

"5" is 00110101 ";" is 00111011

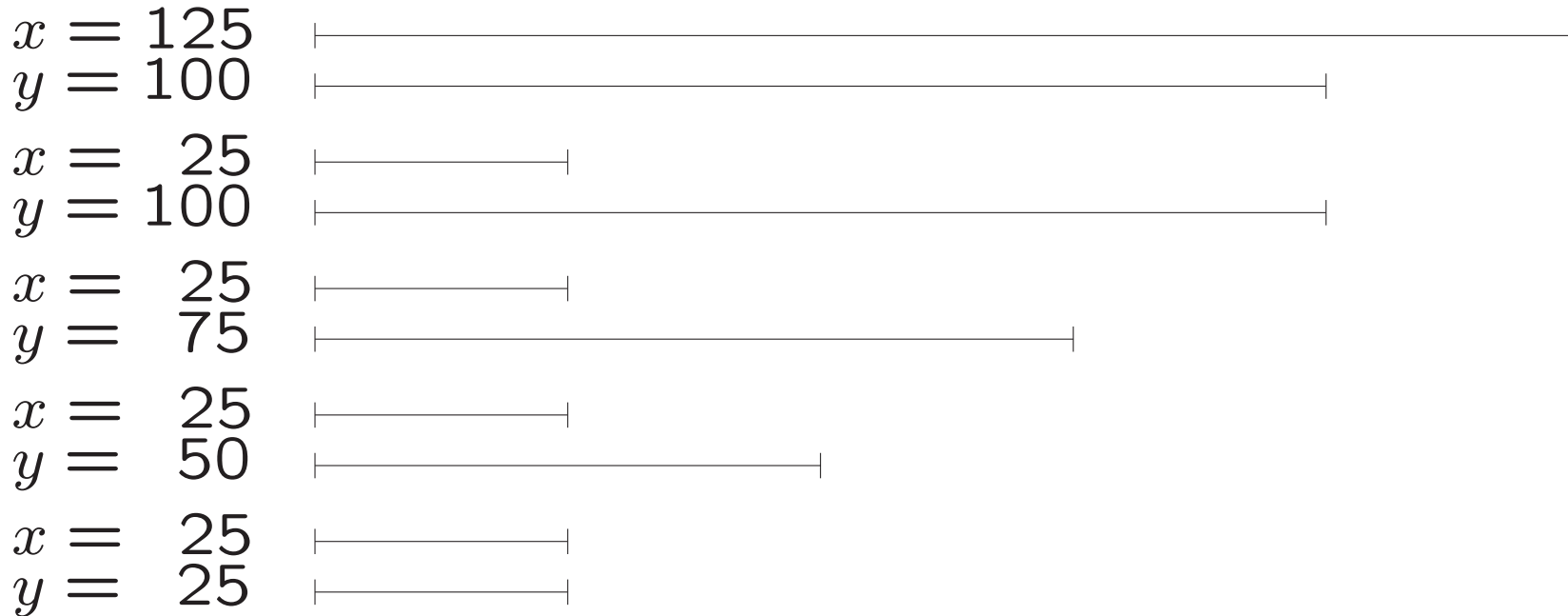
How Does It Work?

- Describe problem as “recipe” (algorithm)
- Write the algorithm as a “program” in a programming language
- Translate the program to a form that the computer can execute
- Execute the program

The Oldest Algorithm (Euclid)

- Find Greatest Common Factor of x and y .
- Greatest Common Factor of 125 and 100 is 25.
- Input x, y ;
While $x \neq y$ Do Begin
 If $x > y$ Then Update $x = x - y$;
 Else Update $y = y - x$; End
Output x .
- Faster with modulo-operation instead of subtraction.

Euclid's Algorithm comes from Geometry



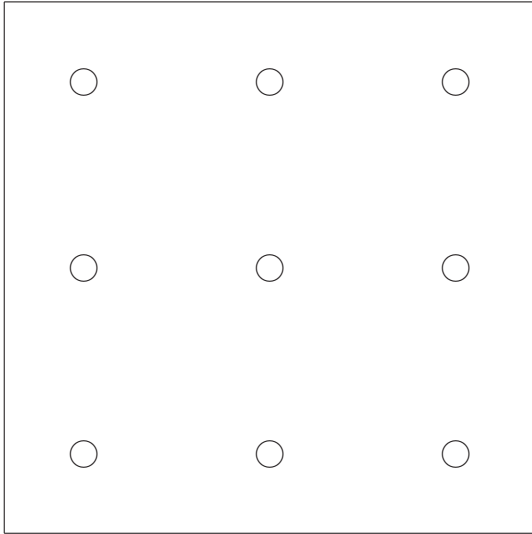
Find longest line such that two given lines are multiples of it.

While lines are not of equal length, subtract shorter line from longer one.

Overview of this Lecture

- Administrative matters
- Course overview
- What problems?
- What computers?
- **Problem solving (with and without computers)**
- Goals of the course

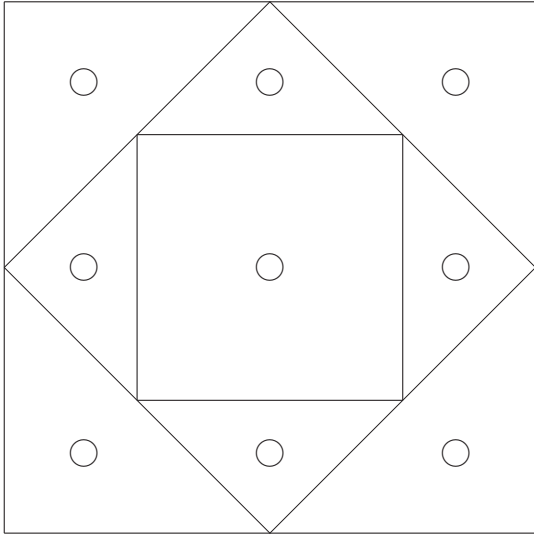
Example 1



Place two more squares
to give each peg its own
compartment

after Susan Blackmore,
Consciousness

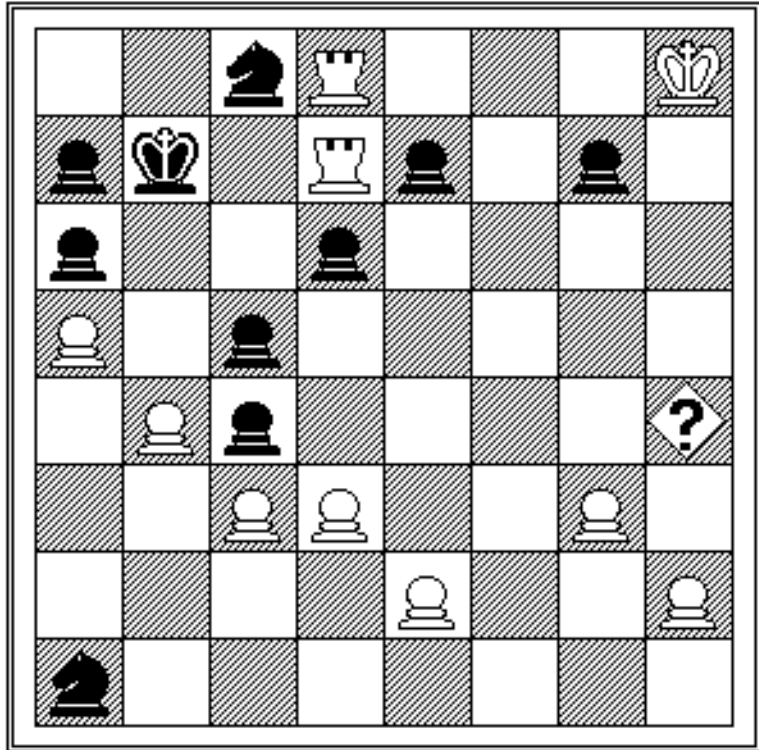
Solution



First new square
is placed as a diamond
inside original square.
Second new square is
placed inside first one.

after Susan Blackmore,
Consciousness

Example 2



Find the missing piece

from Raymond Smullyan,
*The Chess Mysteries
of Sherlock Homes.*

“Retrograde Analysis.”

Black has moved.

White will take Black’s king.

Question: What can be said
about the missing piece?

Solution

- Missing piece is the white bishop (that one which walks on black squares).
- Black king is in check by walking there. This can only happen if the black king is checkmate anyway. So black king has taken a white piece by moving from a8 or a7 to the white square b7.
- Two black pawns have taken 4 white pieces which can walk on white squares.
- Five white pieces which can walk on white fields have been taken: the queen, two Knights, one bishop (which walks on white squares) and one pawn. The white bishop walking on black squares is the remaining missing piece.

Example 3

- There are three animals: a sheep, a cat, a mouse.
- The sheep is black or white.
- The cat is black or white or brown.
- The mouse is white or grey.
- All animals have different colours.
- The sheep is darker than the cat.
- If the cat is brown then the sheep is white.
- Which animal has which colour?

Formalization

- Predicates: $X(Y)$ says “The animal Y has the colour X ”.
- Quantification: \forall Animal $Y \exists$ Colour $X [X(Y)]$.
- Implication: $\text{Brown}(\text{Cat}) \Rightarrow \text{White}(\text{Sheep})$.
- Disjunction (Or): $\text{Black}(\text{Sheep}) \vee \text{White}(\text{Sheep})$.
- Negation (Not): $\neg \text{Brown}(\text{Sheep})$.
- Conjunction (And): $\neg \text{Brown}(\text{Sheep}) \wedge \neg \text{Brown}(\text{Mouse})$.
- Formal facts of the previous slide:
 - $\text{Black}(\text{Sheep}) \vee \text{White}(\text{Sheep})$.
 - $\text{Black}(\text{Cat}) \vee \text{White}(\text{Cat}) \vee \text{Brown}(\text{Cat})$.
 - $\text{White}(\text{Mouse}) \vee \text{Grey}(\text{Mouse})$.
 - \forall Colour $X \forall$ Different Animals Y and $Z [\neg X(Y) \vee \neg X(Z)]$.
 - $\text{Black}(\text{Sheep}) \wedge [\text{Brown}(\text{Cat}) \vee \text{White}(\text{Cat})]$.
 - $\text{Brown}(\text{Cat}) \Rightarrow \text{White}(\text{Sheep})$.
- Solution: $\text{Black}(\text{Sheep}) \wedge \text{White}(\text{Cat}) \wedge \text{Grey}(\text{Mouse})$.

Digits of $\pi = 3.141592653589793\dots$

- Tsu Chung-Chih, Tsu Keng-Chih, 499: 7 digits
- Adriaen van Rooman, 1593: 15 digits
- Ludolph van Ceulen and wife, 1615: 32 digits
- John Machin, 1706: 100 digits
- Martin Zacharias Dase, 1861: 200 digits
- William Shanks, 1873: 707 digits, first 526 correct
- Ferguson, 1947: 808 digits, with desk calculator

Digits of π with Computers

- ENIAC, 1949: 2037 digits
- NORC, 1955: 3089 digits
- IBM 7090, 1961: 200265 digits
- CDC 6600, 1967: 500000 digits
- Hitachi SR8000, 2003: 12411000000000 digits
programmed by Ushio and Kuroda

Petr Beckmann, A history of π , The Golem Press, 1971.

<http://mathworld.wolfram.com/PiDigits.html>

More on $\pi = 3.$

1415926535897932384626433832795028841971693993751058209749445923
0781640628620899862803482534211706798214808651328230664709384460
9550582231725359408128481117450284102701938521105559644622948954
9303819644288109756659334461284756482337867831652712019091456485
6692346034861045432664821339360726024914127372458700660631558817
4881520920962829254091715364367892590360011330530548820466521384
1469519415116094330572703657595919530921861173819326117931051185
4807446237996274956735188575272489122793818301194912983367336244
0656643086021394946395224737190702179860943702770539217176293176
7523846748184676694051320005681271452635608277857713427577896091
7363717872146844090122495343014654958537105079227968925892354201
9956112129021960864034418159813629774771309960518707211349999998
3729780499510597317328160963185950244594553469083026425223082533
4468503526193118817101000313783875288658753320838142061717766914
7303598253490428755468731159562863882353787593751957781857780532
171226806613001927876611195909216420199

<http://www.factmonster.com/ipka/A0876705.html>

Half of these digits were found before the age of computers.

Summary “Problem Solving”

- Some problems cannot be solved with computers;
- Some problems *should* not be solved with computers;
- Some problems can be solved by humans quite well although computers are better.

Overview of this Lecture

- Administrative matters
- Course overview
- What problems?
- What computers?
- Problem solving (with and without computers)
- **Goals of the course**

The Goals of This Course

- **Learn about the possibilities of computers:**

Get a feeling for what kinds of problems can be solved with computers.

Learn basic programming skills.

Practice algorithms and topics from the lecture by completing the given assignments.

- **Learn about the limitations of computers:**

Get a feeling for problems that are hard for computers.

Find out why some problems cannot be solved with computers.

Find out why some problems cannot be solved at all.

- **Programming helps understanding Computer Science:**

Appreciation for computer-based problem solving requires doing it!

First assignments teach mainly the programming language.

Later assignments help to understand the topics of the lecture better by completing programs for them.

Assignments

- Create your account before laboratory session starts in Week 3.
- Download and edit page in the laboratory session.
- Edit program in own webpage with “pico” or “vi”; check out result with browser.
- 1 assignment = 1 mark, 2 assignments = 2 marks, ..., 24 assignments = 24 marks, all 28 assignments = 26 marks;
- Solve assignments either at home or in the laboratory and show them to the lecturer in the laboratory. In each session one assignment can get help.
- The programming language is Java Script. The assignments will consist of existing programs by filling in missing details or editing them in order to make it easier to deal with the new language.
- <http://www.comp.nus.edu.sg/~gem1501#assignments>
<http://www.comp.nus.edu.sg/~gem1501/javascriptintroduction.html>

Next Week

- Algorithms and Data
 - Control structures
 - Diagrams for algorithms
 - Subroutines, procedures
 - Recursion
 - Data types and data structures