UNIVERSITAS TRISAKTI
PRODI TEKNIK PERTAMBANGAN

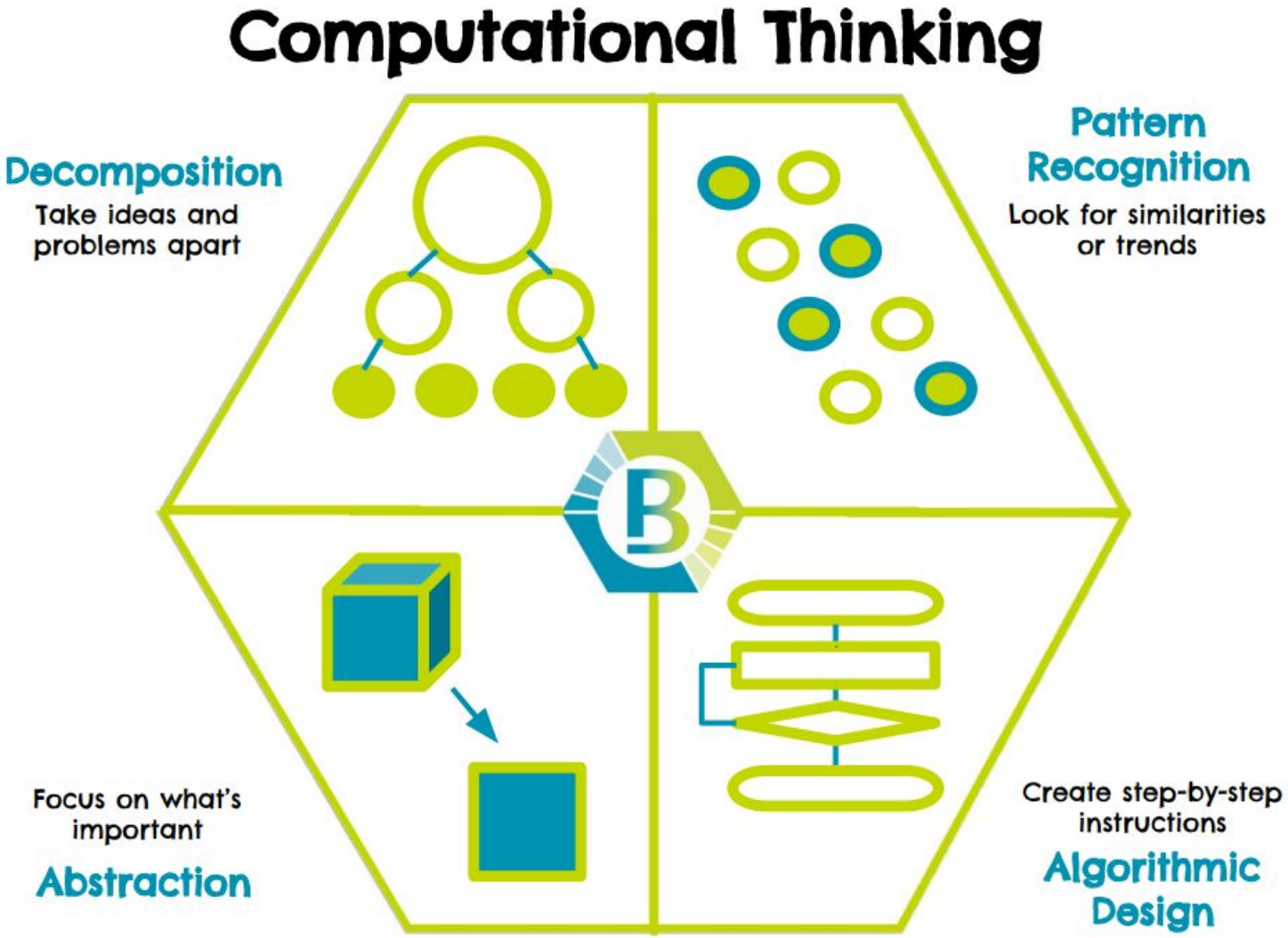
INTRODUCTION

MFU6207 - KOMPUTASI DASAR
(*COMPUTATIONAL THINKING*)

1st session
1.5 hours

Lecturer :
Danu Putra, S.T., M.T.

WHAT IS COMPUTATIONAL THINKING ?



SEMESTER SYLLABUS

1. Introduction
2. Apa itu computational thinking ?
3. Berpikir Logika (*Logical Thinking*) dan Algoritma (*Algorithmic Thinking*)
4. Berpikir Algoritma (*Algorithmic Thinking*) dan Penyelesaian Masalah *(Problem Solving)*
5. Dekomposisi (*Decomposition*) dan Abstraksi (*Abstraction*)
6. Model dan Error

Mid Term

1. Siklus Operasi Penambangan (Batubara)
2. Siklus Operasi Penambangan (Mineral)
3. Kebidangan dalam Pertambangan
4. Berpikir Komputasi dalam Software Penambangan
5. Case Study
6. Case Study
7. Case Study

Final Term

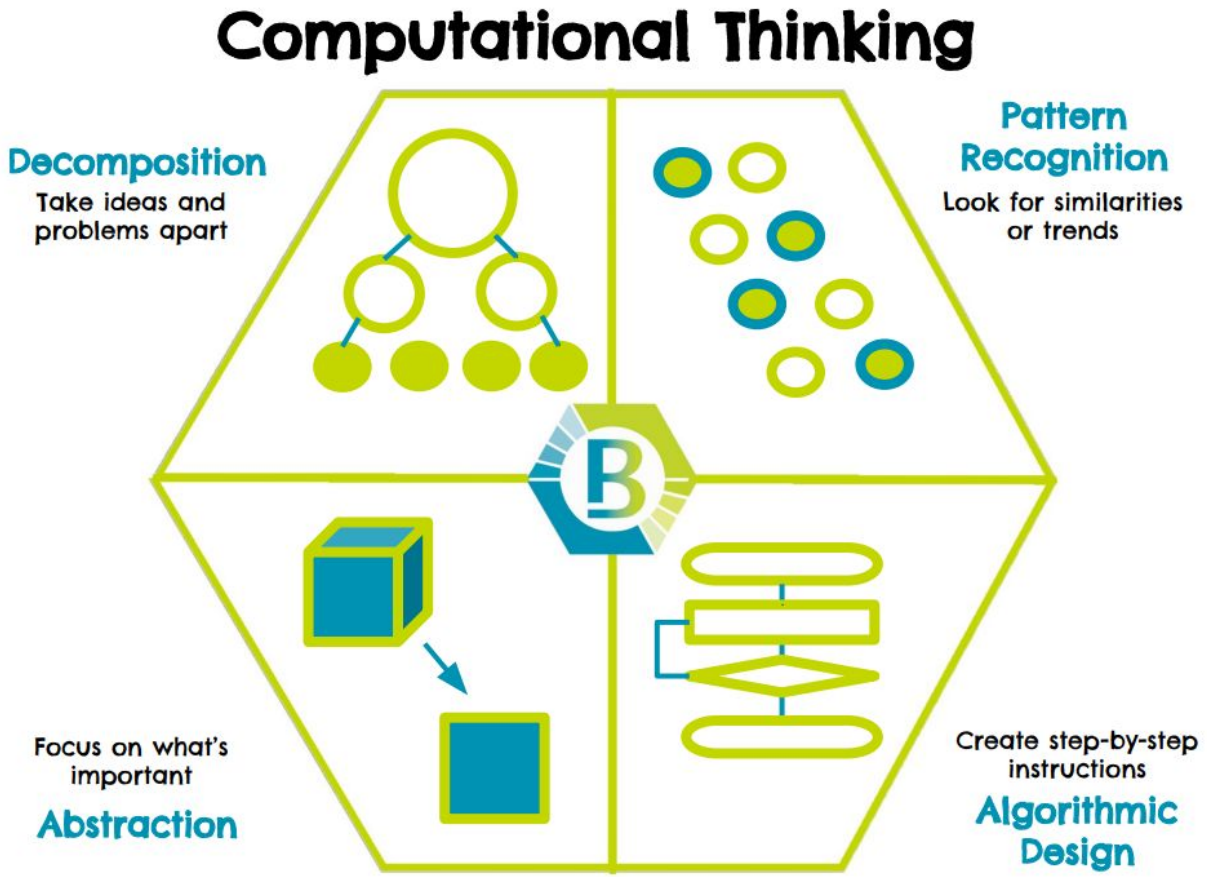
UNIVERSITAS TRISAKTI
PRODI TEKNIK PERTAMBANGAN

WHAT IS COMPUTATIONAL THINKING ?

MFU6207 - KOMPUTASI DASAR
(*COMPUTATIONAL THINKING*)

2nd session
1.5 hours

Lecturer :
Danu Putra, S.T., M.T.



www.wcpss.net

SEMESTER SYLLABUS

1. Introduction
2. Apa itu computational thinking ?
3. Berpikir Logika (*Logical Thinking*) dan Algoritma (*Algorithmic Thinking*)
4. Berpikir Algoritma (*Algorithmic Thinking*) dan Penyelesaian Masalah *(Problem Solving)*
5. Dekomposisi (*Decomposition*) dan Abstraksi (*Abstraction*)
6. Model dan Error

Mid Term

1. Siklus Operasi Penambangan (Batubara)
2. Siklus Operasi Penambangan (Mineral)
3. Kebidangan dalam Pertambangan
4. Berpikir Komputasi dalam Software Penambangan
5. Case Study
6. Case Study
7. Case Study

Final Term

DEFINITION

Computational thinking is the **thought processes** involved in **formulating a problem** and **expressing its solution(s)** in such a way that a computer—human or machine—**can effectively carry out** **(Wing, 2014)**

The mental activity for **abstracting problems** and **formulating solutions** that **can be automated** **(Yadav et al., 2014)**

A mental orientation to **formulating problems** as conversions of some input to an output and **looking for algorithms to perform the conversions**. Today the term has been expanded to include thinking with many levels of abstractions, use of mathematics to develop algorithms, and **examining how well a solution scales** across different sizes of problems. **(Denning, 2009)**

CORE CONCEPTS

- logical thinking;
- algorithmic thinking;
- decomposition;
- generalisation and pattern recognition;
- modelling;
- abstraction;
- evaluation.

Beecher, Karl. 2017

HOW IS IT BEING USED? – LOGICAL THINKING

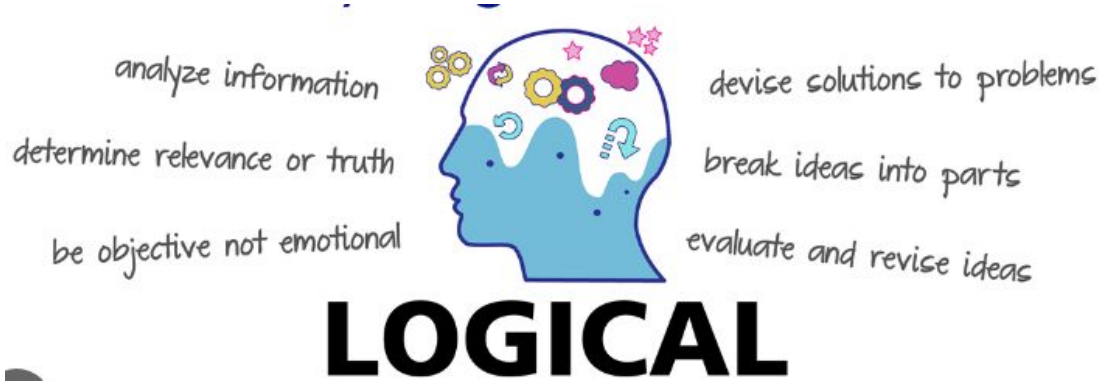
- logical thinking;
- algorithmic thinking;
- decomposition;
- generalisation and pattern recognition;
- modelling;
- abstraction;
- evaluation.

Beecher, Karl. 2017

To a computer scientist, it means the **study of algorithms and their application** to different problems.

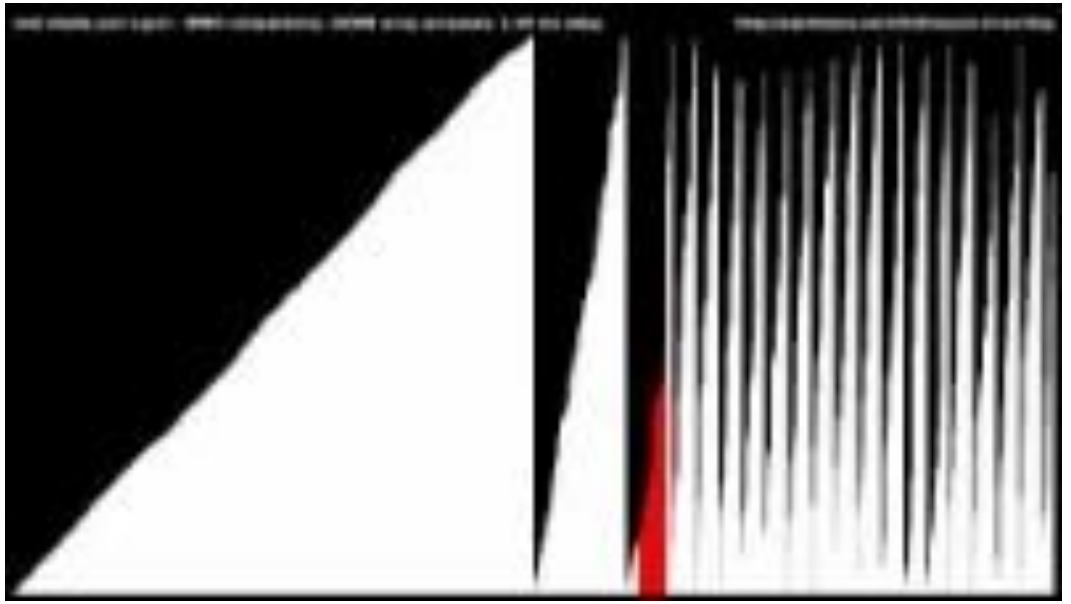
To a mathematician, it might mean **carrying out long division factoring** or doing carries in addition or subtraction.

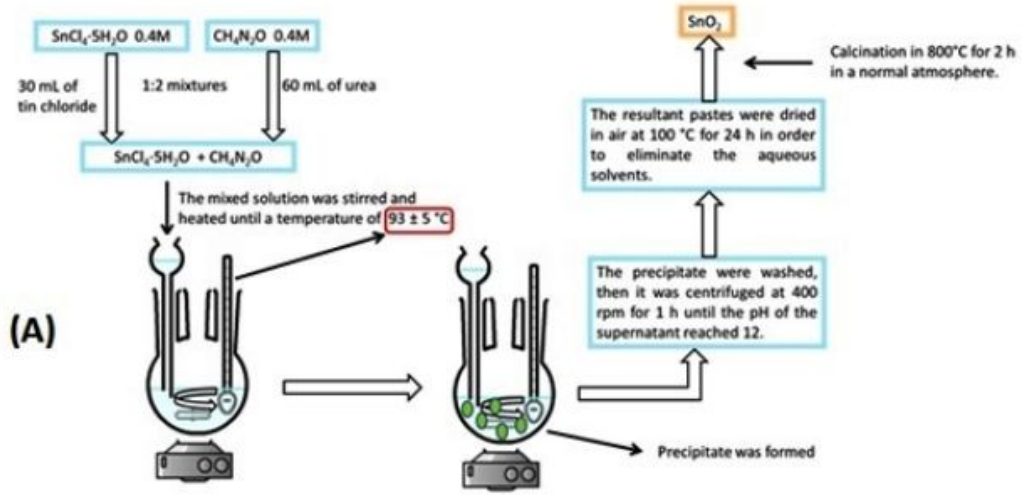
A scientist might think of it as the process of doing an **experimental procedure.**



HOW IS IT BEING USED? – ALGORITHMIC THINKING

- logical thinking;
- algorithmic thinking;
- decomposition;
- generalisation and pattern recognition;
- modelling;
- abstraction;
- evaluation.





Beecher, Karl. 2017

HOW IS IT BEING USED? – DECOMPOSITION

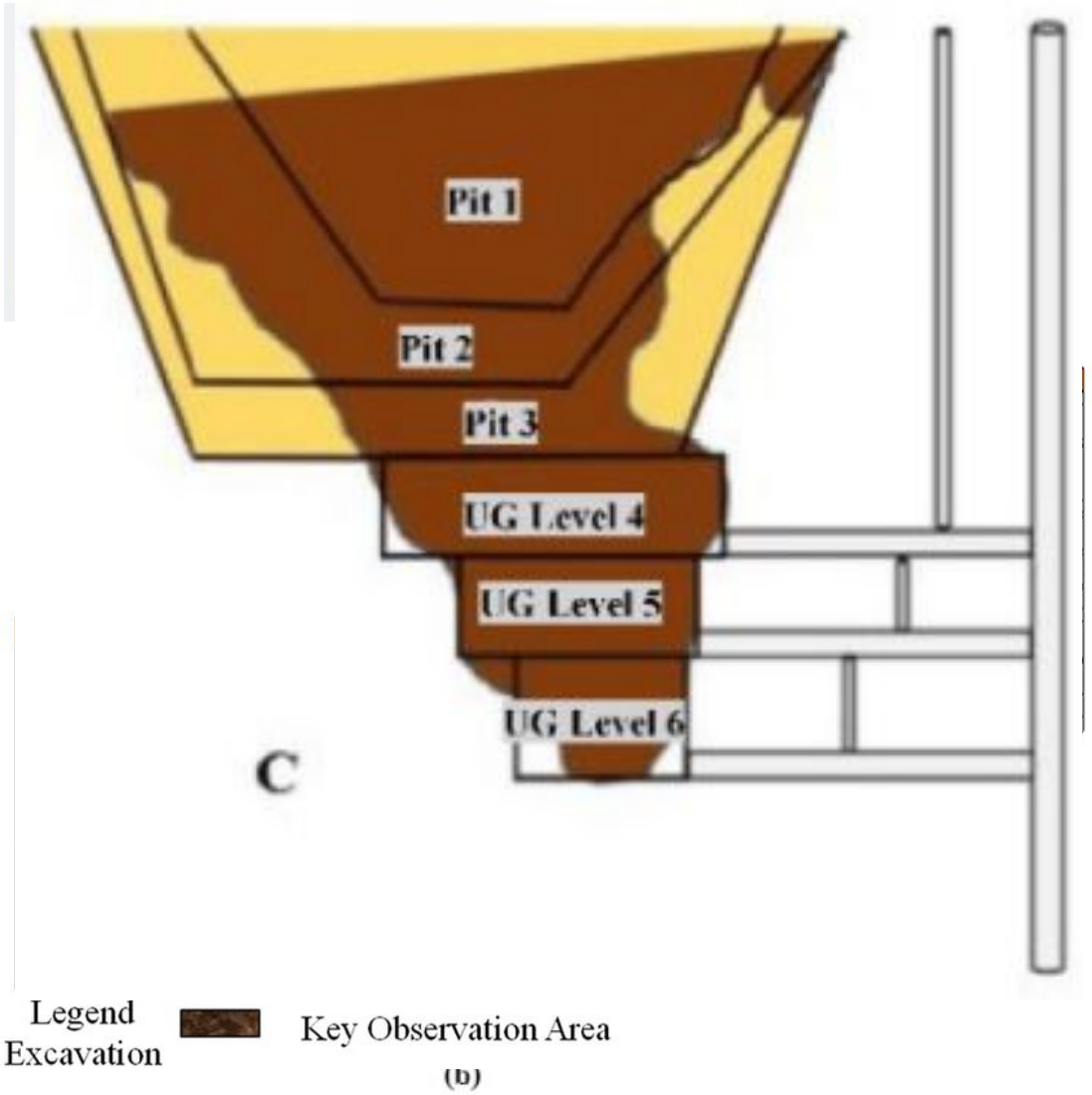
- logical thinking;
- algorithmic thinking;
- decomposition;
- generalisation and pattern recognition;
- modelling;
- abstraction;
- evaluation.

Beecher, Karl. 2017

+ MINING
+ COAL PREPARATION
+ OVERLAND CONVEYOR
+ TANJUNG BARA COAL TERMINAL
+ TRANSHIPPING OPERATION
+ QUALITY ASSURANCE
+ OPERATIONS INFRASTRUCTURE
+ ENVIRONMENTAL MANAGEMENT
+ THE HUMAN ENVIRONMENT
+ SAFETY
+ COMMUNITY DEVELOPMENT PROGRAM

HOW IS IT BEING USED? – GENERALIZATION AND PATTERN RECOGNITION

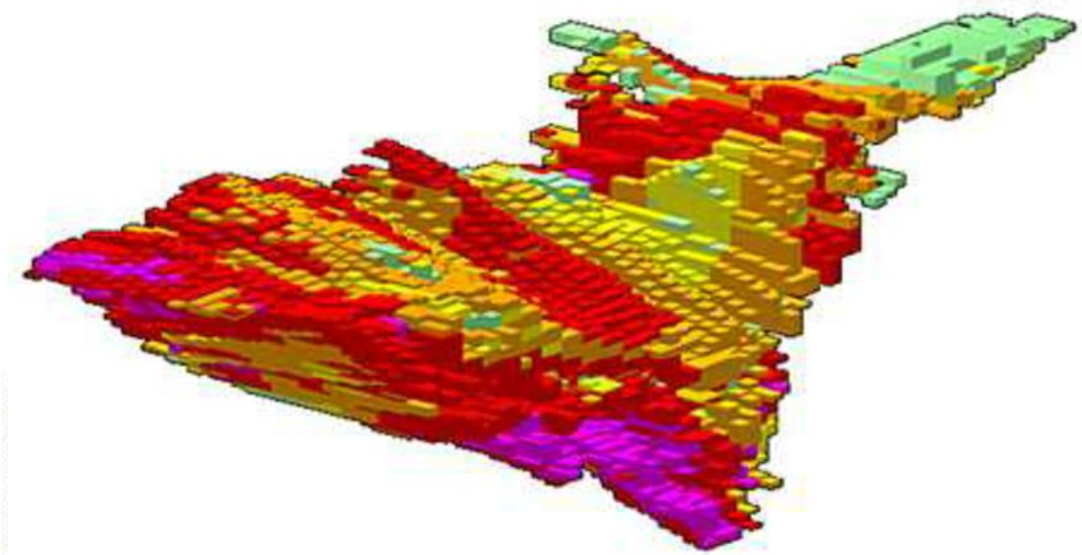
- logical thinking;
- algorithmic thinking;
- decomposition;
- generalisation and pattern recognition;
- modelling;
- abstraction;
- evaluation.

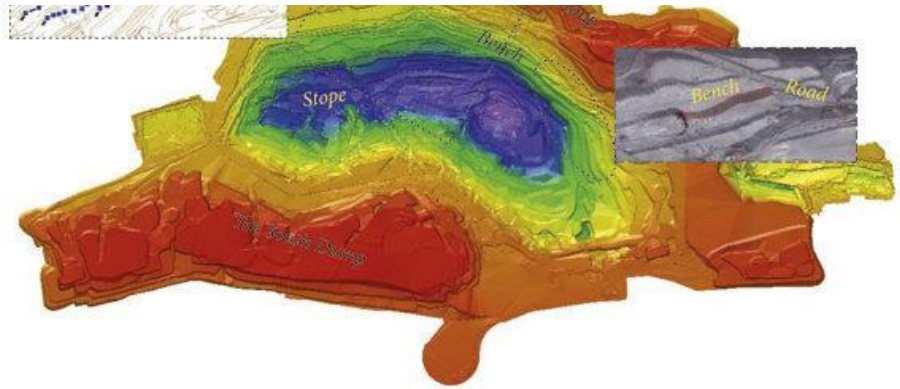


Beecher, Karl. 2017

HOW IS IT BEING USED? – MODELING

- logical thinking;
- algorithmic thinking;
- decomposition;
- generalisation and pattern recognition;
- modelling;
- abstraction;
- evaluation.

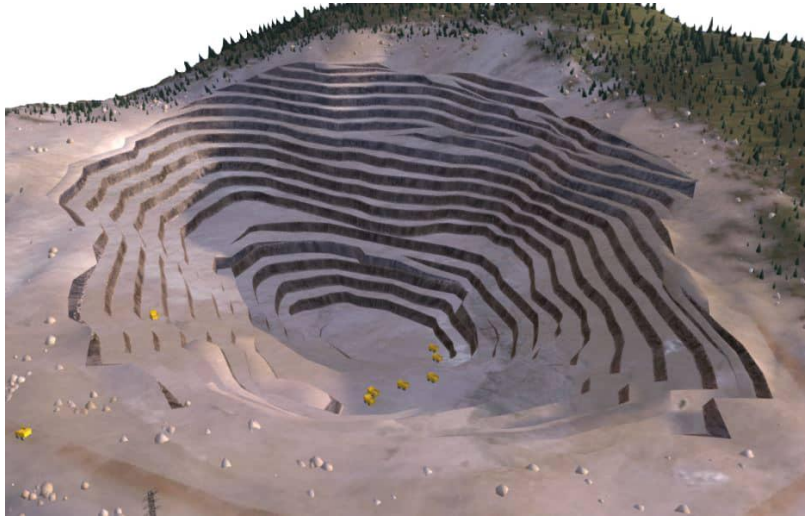




Beecher, Karl. 2017

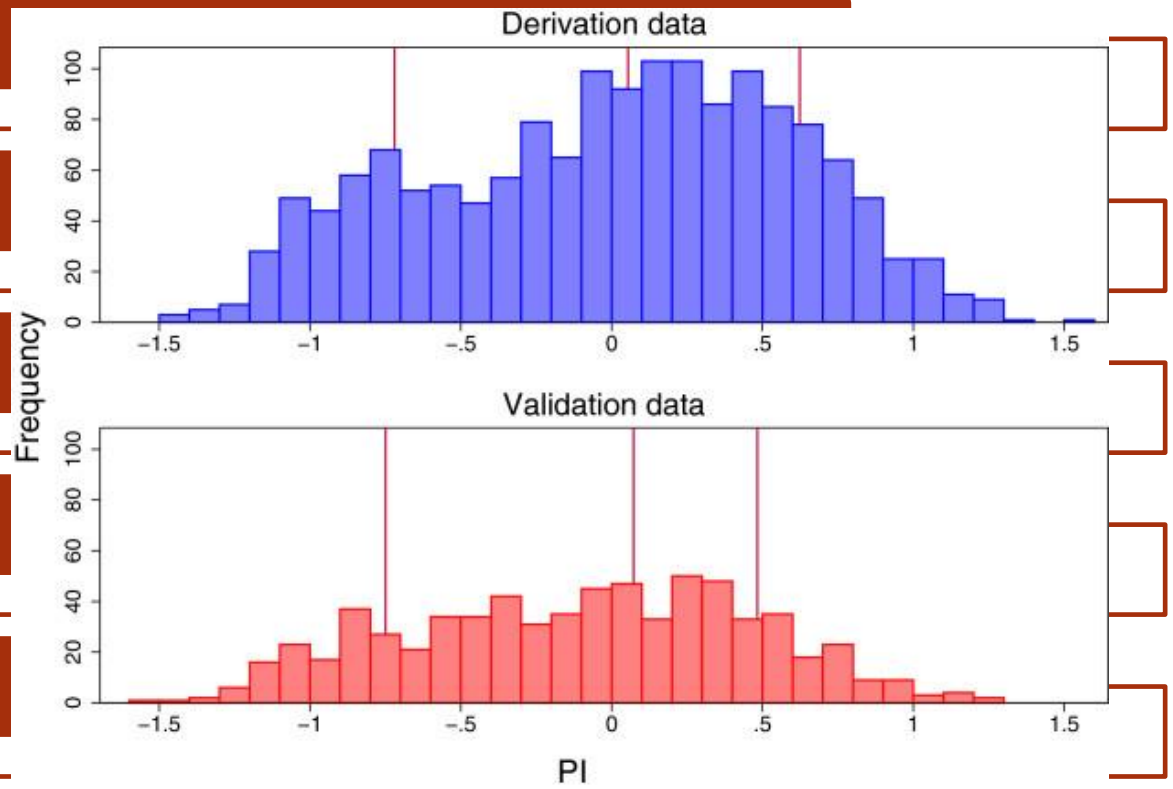
HOW IS IT BEING USED? – ABSTRACTION

- logical thinking;
- algorithmic thinking;
- decomposition;
- generalisation and pattern recognition;
- modelling;
- abstraction;
- evaluation.



Beecher, Karl. 2017

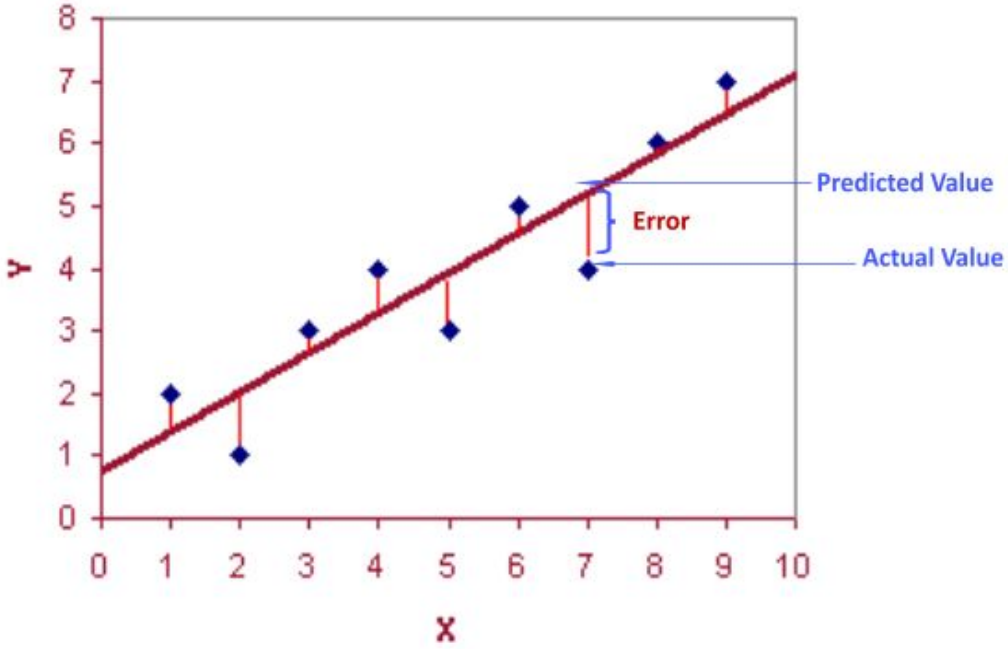
HOW IS IT BEING USED? – EVALUATION



abstraction;

evaluation.

RMSE



Beecher, Karl. 2017

WHAT IS COMPUTATIONAL THINKING NOT?

Computational thinking **is not the same** as teaching computer science.

THE DEBATE

Jeannette Wing, have said that computational thinking is 'thinking like a computer scientist'. However, others **have criticised this phrase** (Denning, 2009; Hemmendinger, 2010) because it leaves in place a strong association with computer science, which runs **the risk of people seeing it not as an everyday skill, but merely as a repackaging of CS**.

THE AIM

CT teaches **an approach to problem-solving** where the ultimate aim is to **provide a solution** whose form means **it is ready to be programmed into a computer**.

CT takes a relatively **small subset of concepts** – which just happen to be **important to CS** – and uses them **to construct a widely applicable, problem-solving approach**.

Beecher, Karl. 2017

CURRENT LIMITATIONS

Computational thinking **is not the same** as teaching computer science.

MATURITY

Cynthia Selby wrote an important paper **proposing a definition for CT** (**Selby, 2013**)
a **focused approach to problem solving**, incorporating thought processes that utilize **abstraction, decomposition, algorithms, evaluation, and generalizations** (**2013**)

EFICACY

It has so far **not been taught widely** for an extended time.

PERCEIVED IMPERIALISM

CT **risk** being **imperialistic and off-putting**
It should be noted that **many practices in CT are not original to the subject**
They **do not simply become computational by virtue of finding application** in computing

Beecher, Karl. 2017

SUMMARY

*"Computational thinking is **an approach to problem-solving** that involves using a set of **practices and principles from computer science** to **formulate a solution** that's executable by a computer."*

"It's not just for programmers. In fact, it's applicable in a diverse array of fields."

Beecher, Karl. 2017

SEMESTER SYLLABUS

1. Introduction
2. Apa itu computational thinking ?
3. Berpikir Logika (*Logical Thinking*) dan Algoritma (*Algorithmic Thinking*)
4. Berpikir Algoritma (*Algorithmic Thinking*) dan Penyelesaian Masalah *(Problem Solving)*
5. Dekomposisi (*Decomposition*) dan Abstraksi (*Abstraction*)
6. Model dan Error

Mid Term

1. Siklus Operasi Penambangan (Batubara)
2. Siklus Operasi Penambangan (Mineral)
3. Kebidangan dalam Pertambangan
4. Berpikir Komputasi dalam Software Penambangan
5. Case Study
6. Case Study
7. Case Study

Final Term

Assignment T2

Jelaskan yang anda ketahui mengenai berpikir logika dan algoritma! (sampaikan sitasi)

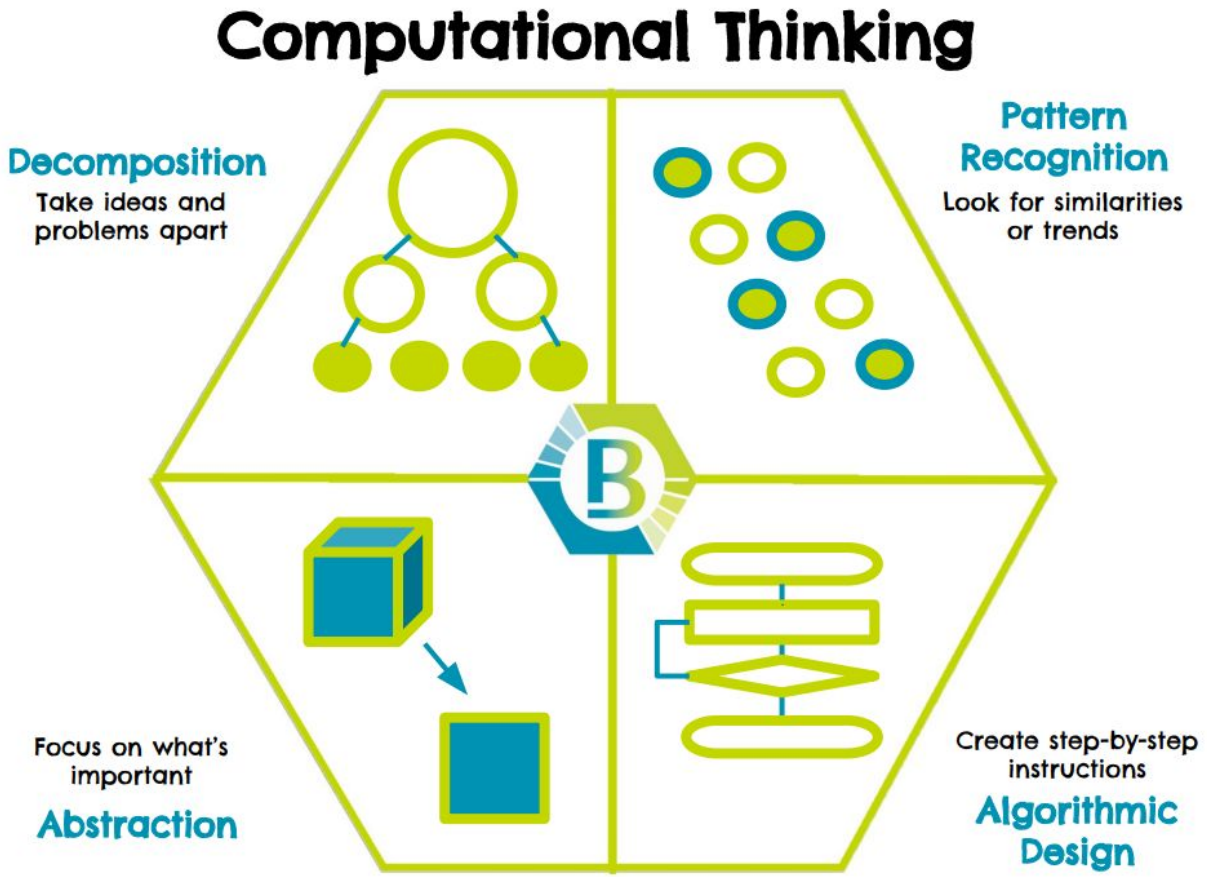
UNIVERSITAS TRISAKTI
PRODI TEKNIK PERTAMBANGAN

LOGICAL THINKING

MFU6207 - KOMPUTASI DASAR
(*COMPUTATIONAL THINKING*)

3rd session
1.5 hours

Lecturer :
Danu Putra, S.T., M.T.



www.wcpss.net

SEMESTER SYLLABUS

1. Introduction
2. Apa itu computational thinking ?
3. Berpikir Logika (*Logical Thinking*) dan Algoritma (*Algorithmic Thinking*)
4. Berpikir Algoritma (*Algorithmic Thinking*) dan Penyelesaian Masalah *(Problem Solving)*
5. Dekomposisi (*Decomposition*) dan Abstraksi (*Abstraction*)
6. Model dan Error

Mid Term

1. Siklus Operasi Penambangan (Batubara)
2. Siklus Operasi Penambangan (Mineral)
3. Kebidangan dalam Pertambangan
4. Berpikir Komputasi dalam Software Penambangan
5. Case Study
6. Case Study
7. Case Study

Final Term

WHAT IS LOGIC?

Logic is a system used for distinguishing between correct and incorrect arguments.

Logic includes a set of principles that, when applied to arguments, allow us to demonstrate what is true :

1. Socrates is a man.
2. All men are mortal.
3. Therefore, Socrates is mortal.

We don't always carry it out correctly, which can lead us to form wrong conclusions. And since we use computers essentially to automate our reasoning, we must **learn to perform logic correctly before writing a computer solution.**

In a sense, applying logic is a way of developing and testing a hypothesis

PREMISES

In a logical argument, **each individual thing you already know (or assume) is called a premise**. A premise is like any ordinary statement you or I might make, except that it can be evaluated to obtain an answer of 'true' or 'false'.

A premise, therefore, has a truth value.

1. Socrates is a man.
2. All men are mortal.

} **premise**

3. Therefore, Socrates is mortal. → **conclusion**

Once all the premises are stated, the next step is to analyse them and react accordingly with a conclusion.

Most of the magic lies in this step and this is what we will focus on.

Beecher, Karl. 2017

INDUCTIVE AND DEDUCTIVE

It's important to realise that some logical arguments are stronger than others. In fact, you can categorise arguments based on their certainty. **The two best-known categories are deductive and inductive**

A deductive argument is the strongest form of reasoning because its conclusion necessarily follows from its premises (so long as it has been constructed properly and the premises are incontrovertibly true).

While deductive arguments are strong, **they have very strict standards**, which makes them hard to construct.

A deductive argument can fail in one of two ways :

1. one of its premises could turn out to be false
   Missie is a dog.
   All dogs are brown.
   Therefore, Missie is brown.

2. the conclusion doesn't necessarily follow from the premises
   All tennis balls are round.
   The Earth is round.
   Therefore, the Earth is a tennis ball.

'garbage in, garbage out'.

faulty logic

This is why we have inductive reasoning,

Deals in **probabilities** rather than hard, black and white rules.

Beecher, Karl. 2017

INDUCTIVE AND DEDUCTIVE (CONT.)

The premises in an inductive argument are not unquestionably true. Rather, we have some level of confidence in them. The form of the argument doesn't guarantee that the conclusion is true, but it probably results in a trustworthy conclusion.

A bag contains 99 red balls and one black ball.
100 people each drew one ball from the bag.
Sarah is one of those 100 people.
Therefore, Sarah probably drew a red ball.

These aspects of reasoning are important in CT because computers are involved.

Beecher, Karl. 2017

INDUCTIVE AND DEDUCTIVE (CONT.)

*"**The answer a computer gives is only as reliable as its reasoning**, and since a computer is automating your reasoning, **it's your responsibility to make sure :***

*1. that reasoning is **valid**;*
*2. you give the computer **reliable input**;*
*3. you **know how to interpret** what conclusion the computer reports, that is, is the result unquestionably true (the reasoning was deductive) or probably true (the reasoning was inductive)?"*

Beecher, Karl. 2017

BOOLEAN LOGIC

Even though much of our reasoning is inductive, **computers are not well equipped to deal with shades of grey**
In order to instruct computers to make logical decisions, we need a system of logic that maps well onto this way of things

PROPOSITIONS

Statements in Boolean logic are also known as propositions, which have several basic properties

- a proposition can only have one value at any one time
  whereas real-life problems often present us with probabilities, **the basic Boolean world deals in certainties.**

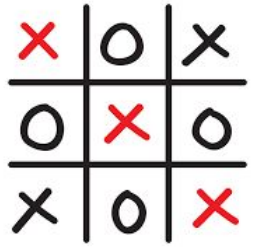
"Some of your efforts will go into mapping real-world, grey areas onto Boolean black and white"

- propositions must have clear and unambiguous meaning
  It is travelling **fast** <> It is travelling fast, where "fast" is **70 mph or greater.**
- it's possible to combine individual propositions to make more complex ones (called compound propositions)

Jenny is wearing the shirt ***and*** the shirt is red

Beecher, Karl. 2017

BOOLEAN LOGIC



LOGICAL OPERATORS
(AND - CONJUCTION)

"If the weather is sunny ***and*** i'm on holiday, then i'm going to lie in the Garden."

propositions **propositions**

Logical operators

(OR - DISJUNCTION)

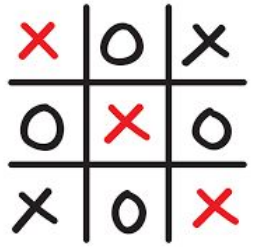
"If player 1 achieves a row ***or*** player 2 achieves a row, then the game is over."

propositions **propositions**

Logical operators

Beecher, Karl. 2017

BOOLEAN LOGIC (CONT.)



(NOT - NEGATION)

"If a square is ***not*** occupied, then a player may add their symbol to that square."

propositions

Logical operators

(IF THEN - IMPLICATION)

"***If*** a player achieves a row, then the game is over."

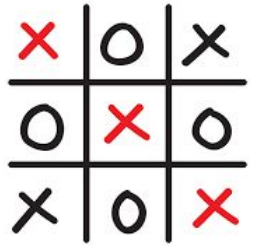
propositions

Logical operators

Beecher, Karl. 2017

BOOLEAN LOGIC (CONT.)

(IF AND ONLY IF - BICONDITIONAL)



"***If and only if*** all squares are occupied, then no more moves are possible.

propositions

Logical operators

Beecher, Karl. 2017

SYMBOLIC LOGIC (CONT.)

Operator name	Symbol	Example
AND	$\wedge$	$A \wedge B$
OR	V	$A \vee B$
NOT	$\neg$	$\neg A$
IMPLIES	$\rightarrow$	$A \rightarrow B$
IF AND ONLY IF	$\leftrightarrow$	$A \leftrightarrow B$

Beecher, Karl. 2017

SYMBOLIC LOGIC – TRUTH TABLE

(AND)

P	*Q*	*P* AND *Q*
True	True	True
True	False	False
False	True	False
False	False	False

(NOT)

P	NOT *P*
True	False
False	True

(OR)

P	*Q*	*P* OR *Q*
True	True	True
True	False	True
False	True	True
False	False	False

(IMPLIES)

P	*Q*	*P* IMPLIES *Q*
True	True	True
True	False	False
False	True	True
False	False	True

.”

Beecher, Karl. 2017

SYMBOLIC LOGIC – TRUTH TABLE

(IF AND ONLY IF)

P	*Q*	*Q* IF AND ONLY IF *P*
True	True	True
True	False	False
False	True	False
False	False	True

.”

Beecher, Karl. 2017

SEMESTER SYLLABUS

1. Introduction
2. Apa itu computational thinking ?
3. Berpikir Logika (*Logical Thinking*) dan Algoritma (*Algorithmic Thinking*)
4. Berpikir Algoritma (*Algorithmic Thinking*) dan Penyelesaian Masalah *(Problem Solving)*
5. Dekomposisi (*Decomposition*) dan Abstraksi (*Abstraction*)
6. Model dan Error

Mid Term

1. Siklus Operasi Penambangan (Batubara)
2. Siklus Operasi Penambangan (Mineral)
3. Kebidangan dalam Pertambangan
4. Berpikir Komputasi dalam Software Penambangan
5. Case Study
6. Case Study
7. Case Study

Final Term

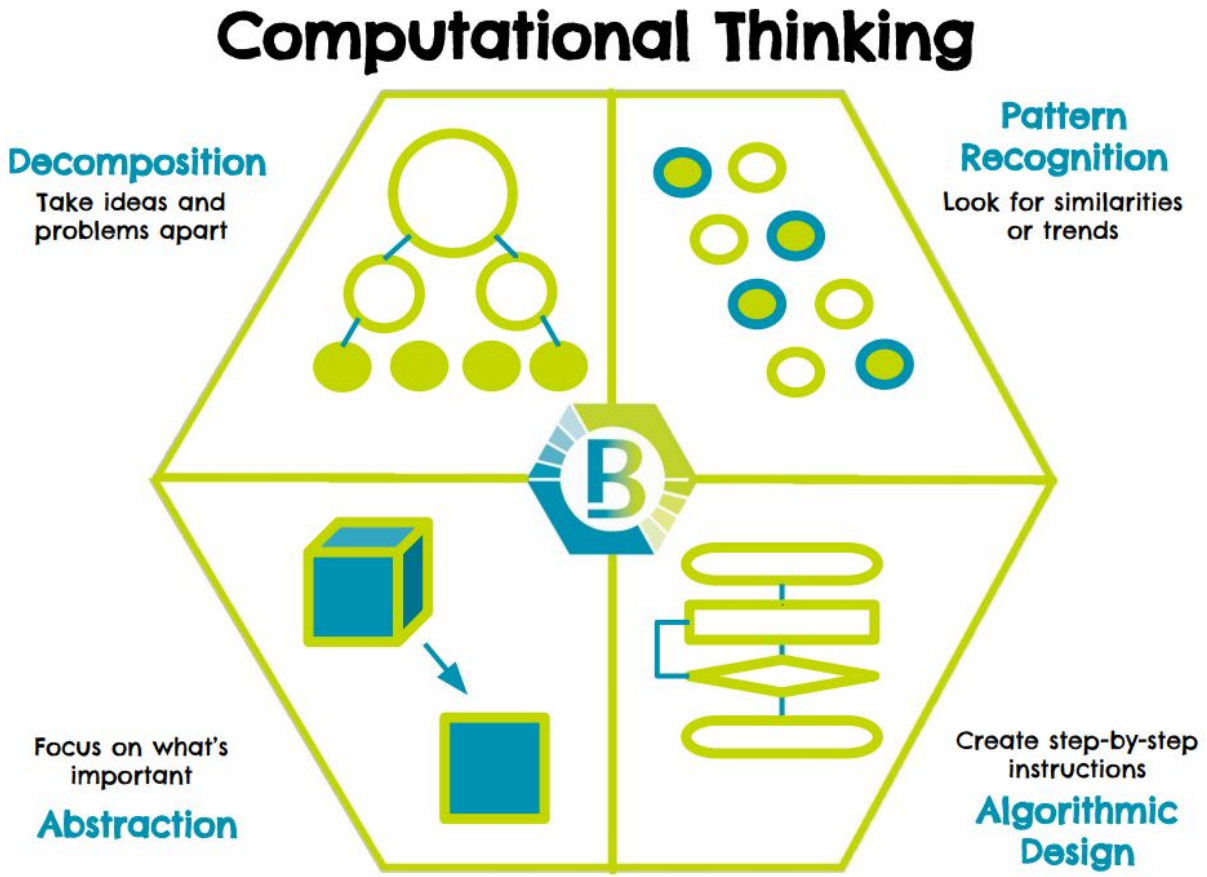
UNIVERSITAS TRISAKTI
PRODI TEKNIK PERTAMBANGAN

ALGORITHMIC THINKING

MFU6207 - KOMPUTASI DASAR
(*COMPUTATIONAL THINKING*)

4th session
1.5 hours

Lecturer :
Danu Putra, S.T., M.T.



www.wcpss.net

SEMESTER SYLLABUS

1. Introduction
2. Apa itu computational thinking ?
3. Berpikir Logika (*Logical Thinking*) dan Algoritma (*Algorithmic Thinking*)
4. Berpikir Algoritma (*Algorithmic Thinking*) dan Penyelesaian Masalah *(Problem Solving)*
5. Dekomposisi (*Decomposition*) dan Abstraksi (*Abstraction*)
6. Model dan Error

Mid Term

1. Siklus Operasi Penambangan (Batubara)
2. Siklus Operasi Penambangan (Mineral)
3. Kebidangan dalam Pertambangan
4. Berpikir Komputasi dalam Software Penambangan
5. Case Study
6. Case Study
7. Case Study

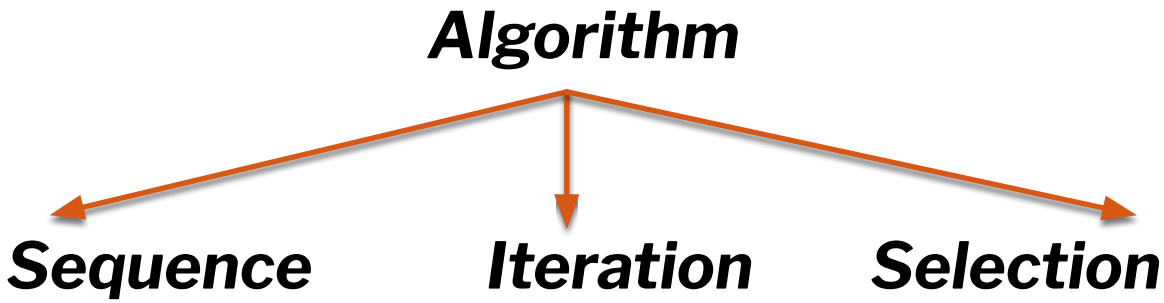
Final Term

APPROACH

The good news is: humans already have an innate, intuitive understanding of both logic and algorithms.

The bad news is : they are both mathematical concepts in nature.

*"**You can't rely solely on your intuition** when dealing with these topics, otherwise you'll make mistakes."*



IN BRIEF : ALGORITHM

Logic and algorithms **are not the** same.

Algorithms build on logic because, as part of their work, they make logical decisions. The other part of their work is **'stitching' those decisions together.**

Case 

"The rules of the game were laid out as logical statements. Sometimes the propositions were true, sometimes false. By working out their truth values, we could come to conclusions about how the game would be in different situations."

Objective

We want to build functioning systems based on rules

Logic isn't sufficient

Need **something** that can integrate all these rules
and execute actions based on the outcomes of evaluating them

Algorithm

Beecher, Karl. 2017

WHAT IS ALGORITHM ?

*"a sequence of **clearly defined steps** that describe a process to follow a **finite set** of unambiguous instructions with **clear start and end points**."*

Beecher, Karl. 2017

DEFINING ALGORITHM

1. An algorithm is **a collection of individual steps.**
2. Following on from that property is **definiteness.**
   - Every step must be **precisely defined**
   - Each step in an algorithm can have **one and only one meaning**

     “some sugar” or “cook it for a while” □ **ambiguous**

3. Algorithms are also **sequential**
   - **The process must be carried out in the order specified**

     “Dicing an onions □ frying an onions”

     “Frying an onions□ dicing an onions”

     different { 1 + 2 * 5 = ??

     (1 + 2) * 5 = ??

Beecher, Karl. 2017

DETOUR : STATE IN ALGORITHM

State

"the current configuration of all information kept track of by a program at any one instant in time."

Note :

*there is **no 'global view'** when it comes to algorithms.*
*individual steps are **executed one by one and only a single step** can be under consideration at any one time. Once a step has been executed, the computer forgets all about it and moves on to the next.*

Solutions : (*Variables*)

- some way of **'remembering' things** that happened in previous steps
- variables in programming **do not correspond** exactly to variables from mathematics
- It can also have its values updated throughout an algorithm's execution (this is called **assignment**)

Beecher, Karl. 2017

CONTROLLING ALGORITHM EXECUTION

Iteration (Looping) & Selection

Iteration allows you to **repeat a series of steps over and over**, without the need to write out each individual step manually

Example :

99 bottles of coke on the wall,
99 bottles of coke.
Take one down, pass it around,
98 bottles of coke on the wall.

98 bottles of coke on the wall,
98 bottles of coke.
Take one down, pass it around,
97 bottles of coke on the wall.

keeps repeating until the last verse

1 bottle of coke on the wall,
1 bottle of coke.
Take it down, pass it around,
No more bottles of coke on the wall.

simplified

Looping statement

X stands for the number of bottles. At the start, X is 99. Sing the verse:

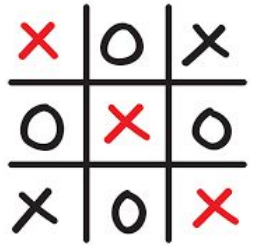
X bottles of coke on the wall,
X bottles of coke.
Take one down, pass it around,
X-1 bottles of coke on the wall.

Subtract 1 from X.
Repeat the verse **if X is greater than 0**, otherwise finish.

selection statement

Beecher, Karl. 2017

EXAMPLES - PSEUDOCODES



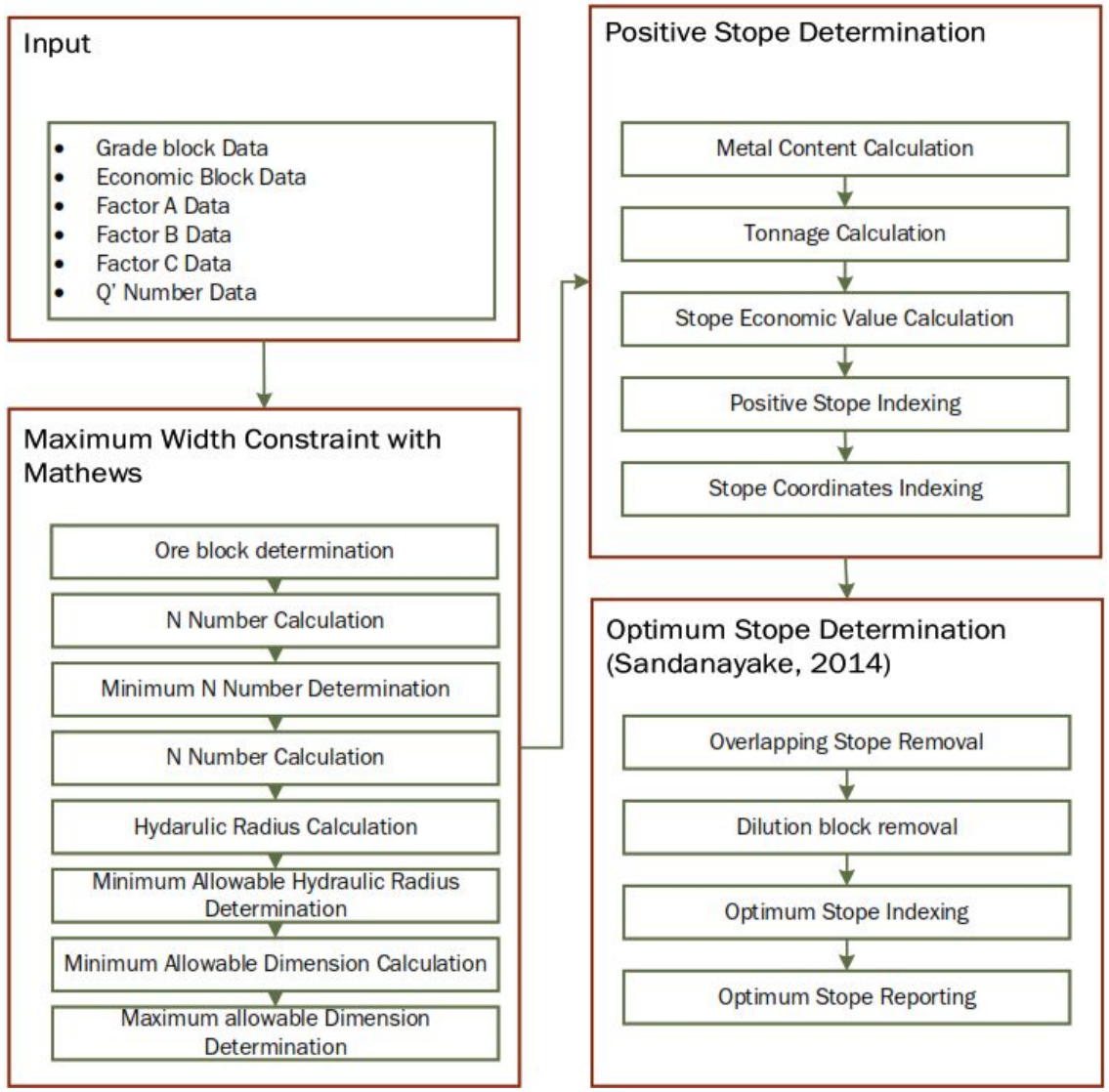
1. game is **started** → **state**
2. begin loop:
    3. prompt player to choose a square
    4. if chosen square is not occupied, then put player's symbol in that square → **selection**
    5. check board to see if a row has been achieved
    6. if a row has been achieved, then game is won → **selection**
    7. if a row has not been achieved and no squares are available, then game is drawn → **selection**
    8. switch to other player
9. exit loop if game is won or game is drawn → **selection**
10. display message 'Game over'

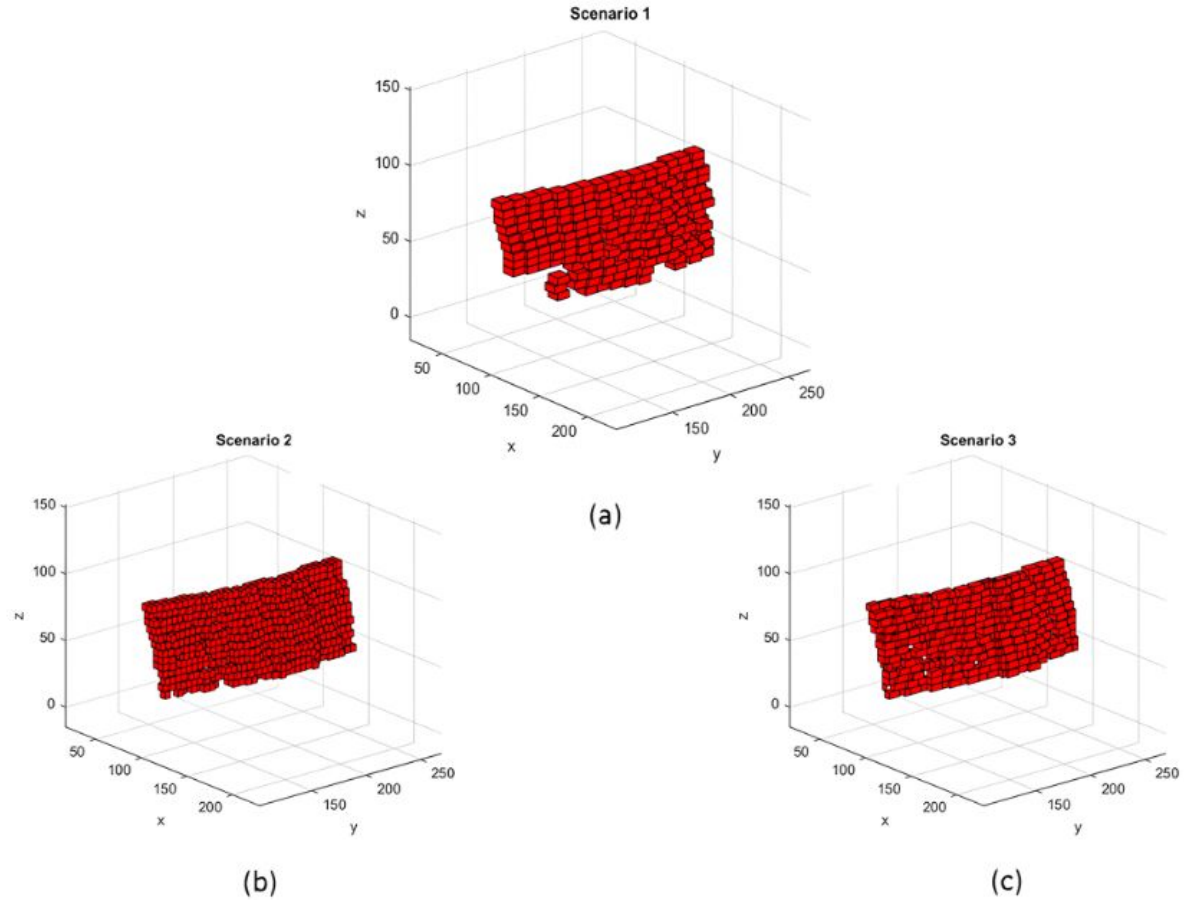
sequence

iteration

Beecher, Karl. 2017

EXAMPLES - RISET





Putra, D,. 2024

CONTROLLING ALGORITHM EXECUTION

Iteration (Looping) & Selection

Iteration allows you to **repeat a series of steps over and over**, without the need to write out each individual step manually

Example :

99 bottles of coke on the wall,
99 bottles of coke.
Take one down, pass it around,
98 bottles of coke on the wall.

98 bottles of coke on the wall,
98 bottles of coke.
Take one down, pass it around,
97 bottles of coke on the wall.

keeps repeating until the last verse

1 bottle of coke on the wall,
1 bottle of coke.
Take it down, pass it around,
No more bottles of coke on the wall.

simplified

Looping statement

X stands for the number of bottles. At the start, X is 99. Sing the verse:

X bottles of coke on the wall,
X bottles of coke.
Take one down, pass it around,
X-1 bottles of coke on the wall.

Subtract 1 from X.
Repeat the verse **if X is greater than 0**, otherwise finish.

selection statement

Beecher, Karl. 2017

KEEP IN MIND

The need for clarity and meticulousness

A couple of facts about this are worth pointing out:

- A computer **will do exactly as it is told**. If it is told to do something impossible, it will crash.
- A computer has **no innate intelligence of its own**. It will not do anything that it has not been instructed to do.
- A computer has **no common sense**.
  - It **will not try to interpret your instructions** in different ways.
  - It **will not make assumptions** or fill in the obvious blanks in an incomplete algorithm.

The goal ,

a solution that a computer can execute. That means an algorithm with **precise, unambiguous meaning** that **requires neither creative intelligence nor common sense** to understand fully.

Beecher, Karl. 2017

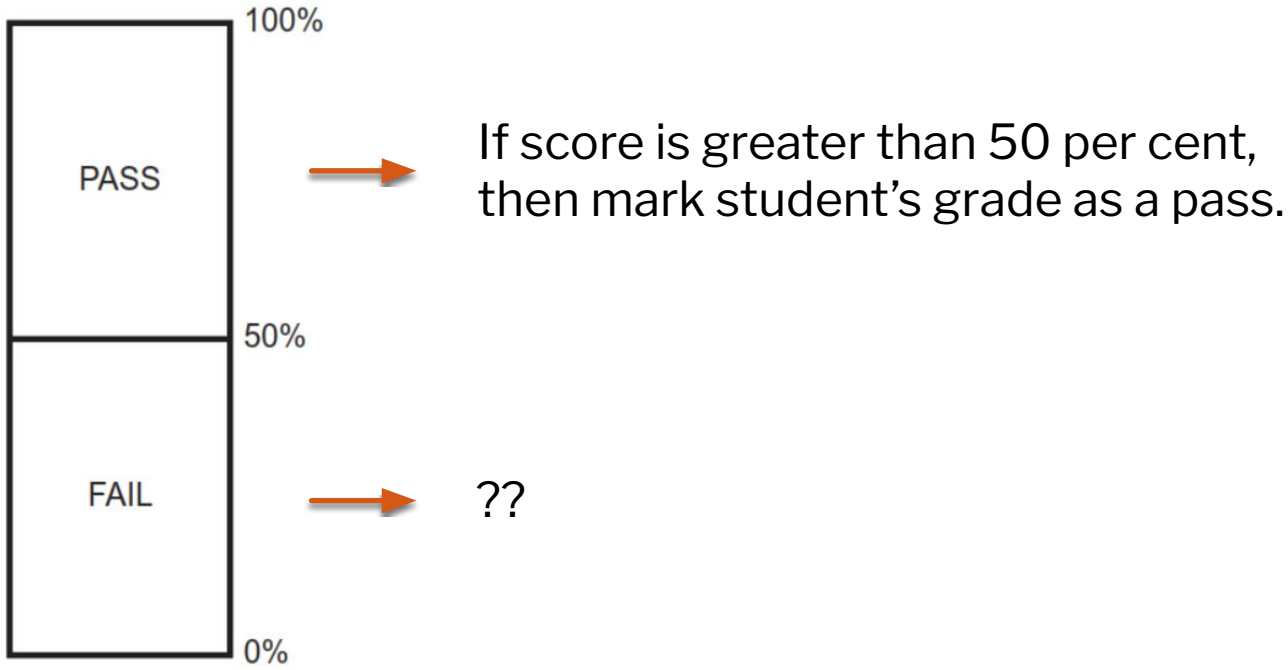
KEEP IN MIND

Incorrect use of logical operators

‘Everyone whose surname begins with A ***and*** B is assigned to Group 1.’

‘The player took their turn and the game was finished.’

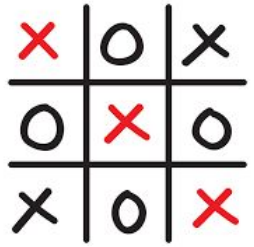
Missing certain eventualities

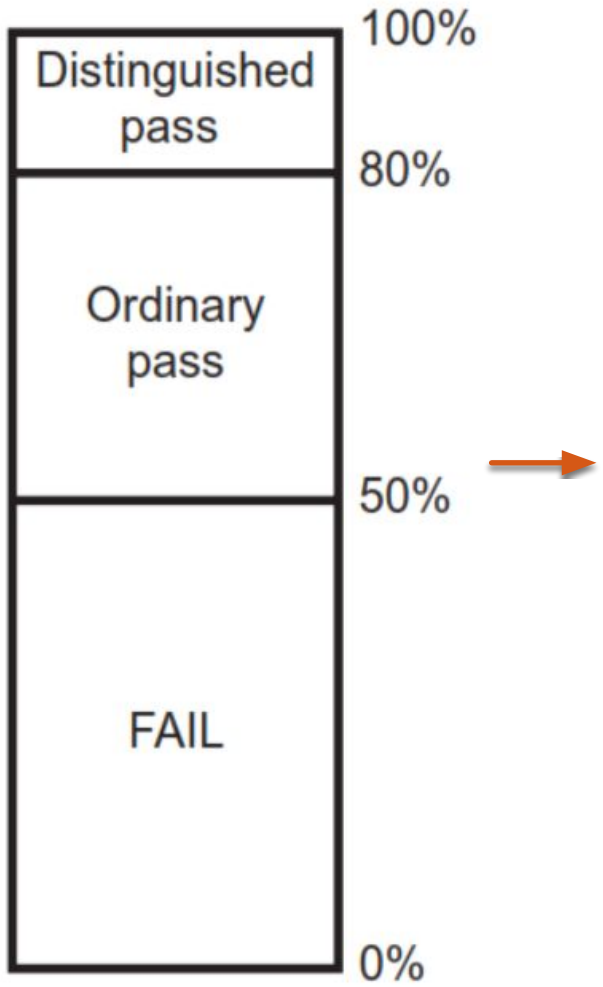


Beecher, Karl. 2017

BOOLEAN LOGIC (CONT.)

Complex conditional





if score is not less than 51 per cent or greater than 80 percent,

then mark student's grade as an ordinary pass

...

Is it good?

Beecher, Karl. 2017

SEMESTER SYLLABUS

1. Introduction
2. Apa itu computational thinking ?
3. Berpikir Logika (*Logical Thinking*) dan Algoritma (*Algorithmic Thinking*)
4. Berpikir Algoritma (*Algorithmic Thinking*) dan Penyelesaian Masalah *(Problem Solving)*
5. Dekomposisi (*Decomposition*) dan Abstraksi (*Abstraction*)
6. Model dan Error

Mid Term

1. Siklus Operasi Penambangan (Batubara)
2. Siklus Operasi Penambangan (Mineral)
3. Kebidangan dalam Pertambangan
4. Berpikir Komputasi dalam Software Penambangan
5. Case Study
6. Case Study
7. Case Study

Final Term

Assignment T2

Jelaskan yang anda ketahui mengenai proses penyelesaian masalah! (sampaikan sitasi)

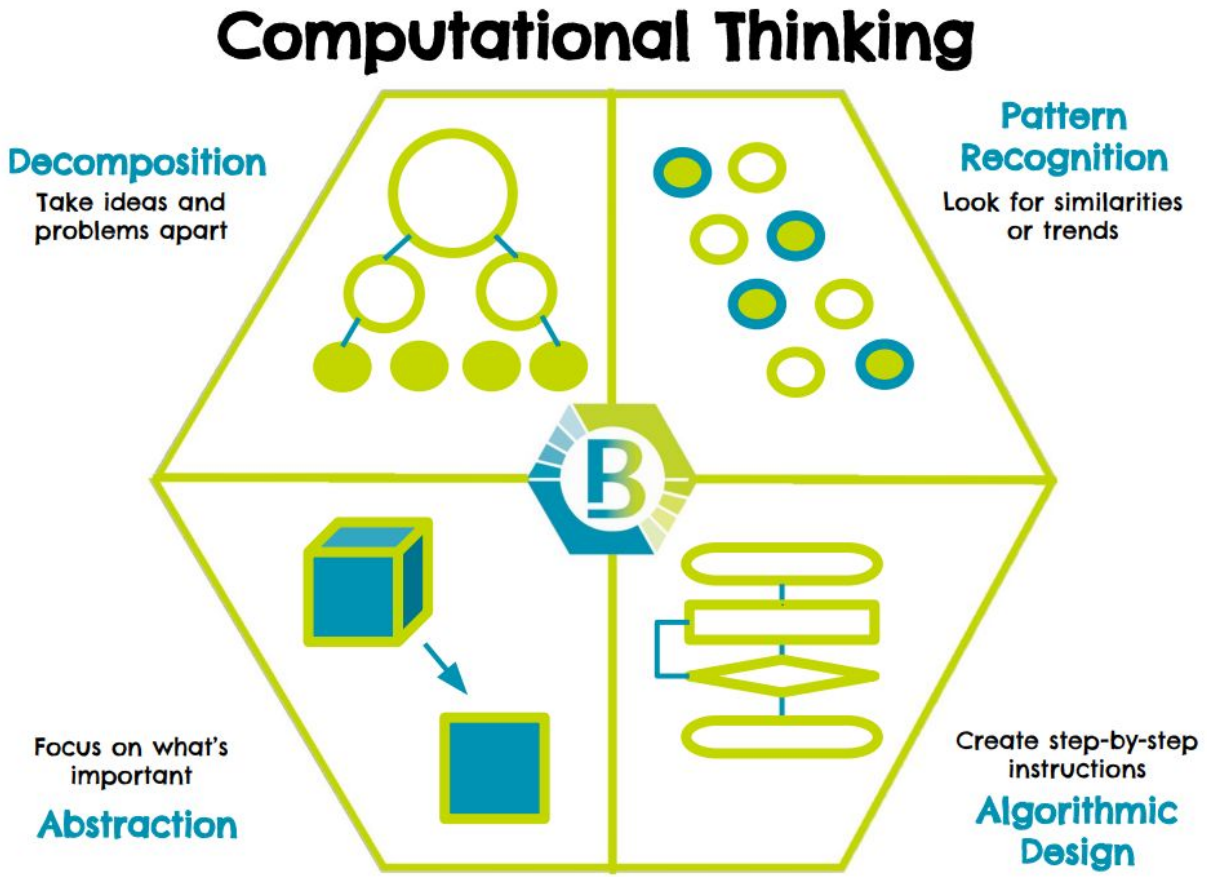
UNIVERSITAS TRISAKTI
PRODI TEKNIK PERTAMBANGAN

PROBLEM SOLVING

MFU6207 - KOMPUTASI DASAR
(*COMPUTATIONAL THINKING*)

5th session
1.5 hours

Lecturer :
Danu Putra, S.T., M.T.



www.wcpss.net

SEMESTER SYLLABUS

1. Introduction
2. Apa itu computational thinking ?
3. Berpikir Logika (*Logical Thinking*) dan Algoritma (*Algorithmic Thinking*)
4. Berpikir Algoritma (*Algorithmic Thinking*) dan Penyelesaian Masalah *(Problem Solving)*
5. Dekomposisi (*Decomposition*) dan Abstraksi (*Abstraction*)
6. Model dan Error

Mid Term

1. Siklus Operasi Penambangan (Batubara)
2. Siklus Operasi Penambangan (Mineral)
3. Kebidangan dalam Pertambangan
4. Berpikir Komputasi dalam Software Penambangan
5. Case Study
6. Case Study
7. Case Study

Final Term

INTRODUCTION

Where to start

Problem-solving is partly a **creative process**

How to Solve It takes an approach to problem-solving inspired by the best traditions of mathematical and natural sciences.

Whereas the scientific method follows the steps:

- form hypothesis;
- plan experiment;
- execute experiment;
- evaluate results.

Pólya (1973) advocates:

- understand the problem;
- devise a plan;
- execute the plan;
- review and extend.

Karl Breecher, 2017

INTRODUCTION

Don’t panic !

First,
don’t be put off by perceived size and complexity

Second,
resist the urge to jump straight into writing a solution

Karl Breecher, 2017

DEFINING THE PROBLEM

Reasons

Problem-solving involves **transforming** an **undesirable state of affairs (the start point)** into **a desirable state of affairs (the goal)**. The start point and the goal are intimately **linked.**

Some reason you might have :

- Maybe your **current process is too slow**, in which case your goal will involve making measurable improvements to the process's speed.
- Maybe you **regularly need to make certain decisions**, but you have too much data to handle. This implies your goal may be to somehow automate your decision-making strategy or filter information before you analyse it.
- Maybe you have **missing information about how something behaves**. This suggests producing a model or simulation of it.

Karl Breecher, 2017

DEFINING THE PROBLEM

Something to keep in mind

Pólya (1973) tells us that 'it is foolish to answer a question that you do not understand', and offers the following advice:

- If someone else gave it to you, **try restating the problem** in your own words.
- Try and **represent the problem using pictures and diagrams**. Humans deal better with visual representations.
- There will be knowns and unknowns at the start. You should ensure that enough information is known for you to form a solution. If there isn't, **make the unknowns explicit.**

"Whatever the problem, the key thing to remember is that a goal ***defines what needs to be done and not how it should be done***"

"be clear and specific"

Karl Breecher, 2017

DEVISING A SOLUTION

Try to do this

Once you have a finished problem definition complete with goal, you can consider the strategy for finding **a solution.**

Quality

There are usually multiple solutions: some good, some terrible and others somewhere in-between. You

should focus on finding **the best solution you can**.

For the **overall** problem, there is likely **no perfect solution.** On the other hand, individual parts of a problem may be 'perfected', in as much as their role in the overall solution might be optimisable.

Collaboration

Brainstorming sessions thrive on spontaneity. All ideas, however radical they seem, should be

recorded, and you should reject nothing out of hand.

Iteration

Go back and repeat some of the previous steps in an attempt to improve your current solution.

Karl Breecher, 2017

SEMESTER SYLLABUS

1. Introduction
2. Apa itu computational thinking ?
3. Berpikir Logika (*Logical Thinking*) dan Algoritma (*Algorithmic Thinking*)
4. Berpikir Algoritma (*Algorithmic Thinking*) dan Penyelesaian Masalah *(Problem Solving)*
5. Dekomposisi (*Decomposition*) dan Abstraksi (*Abstraction*)
6. Model dan Error

Mid Term

1. Siklus Operasi Penambangan (Batubara)
2. Siklus Operasi Penambangan (Mineral)
3. Kebidangan dalam Pertambangan
4. Berpikir Komputasi dalam Software Penambangan
5. Case Study
6. Case Study
7. Case Study

Final Term

Assignment T3

Jelaskan yang anda ketahui mengenai proses dekomposisi dan abstraksi! (sampaikan sitasi)