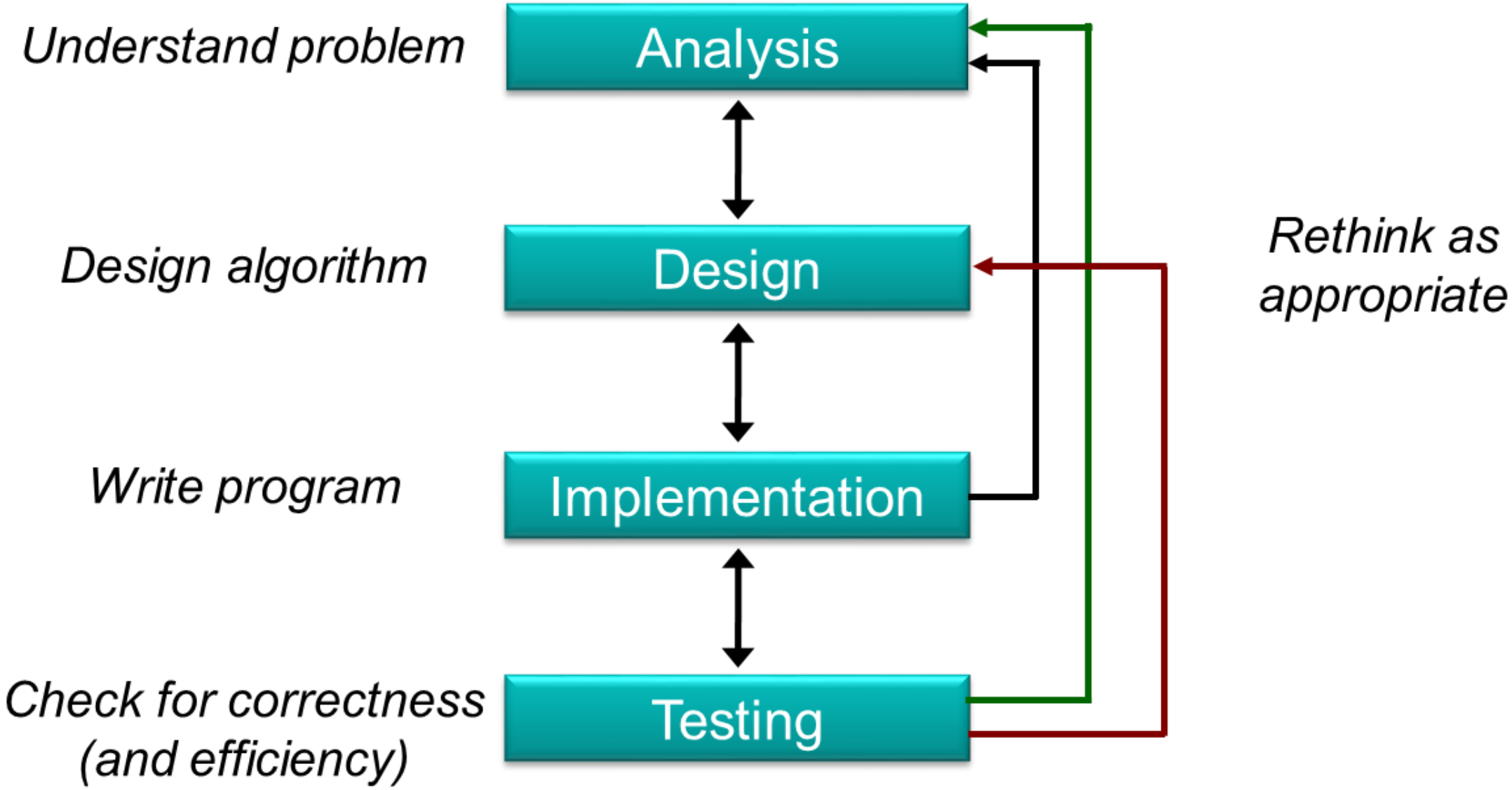
CS1010E Topic 6: MODULAR DESIGN

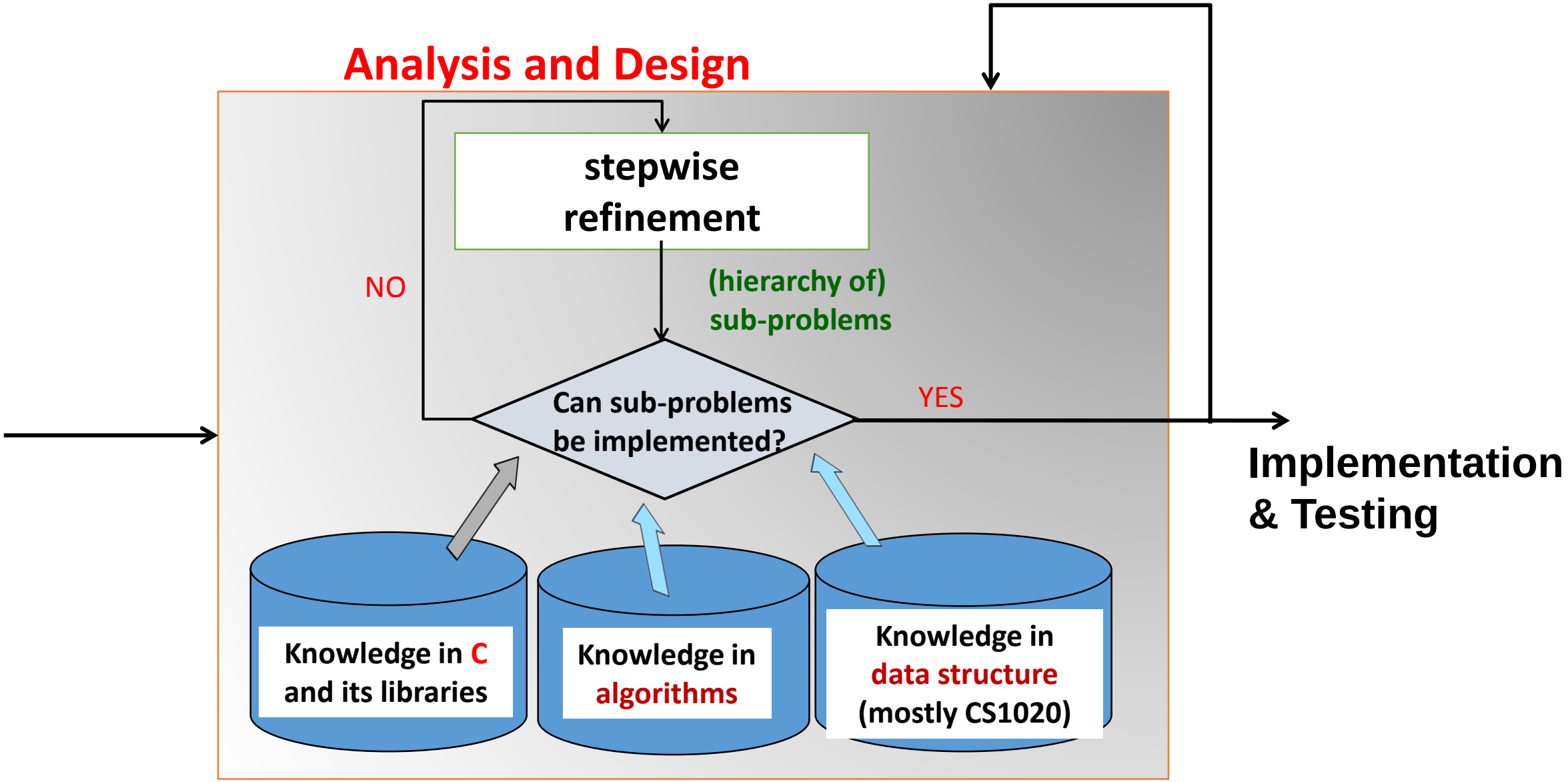
Siau-Cheng KHOO
Block COM2, Room 04-11, +65 6516 6730
www.comp.nus.edu.sg/~khoosc
khoosc@nus.edu.sg

Semester II, 2017/2018

Problem Solving in Programming



Problem Solving in Programming



Top-Down Design

- We introduced functions earlier; they are functions from C libraries and functions defined by programmers.
- Such functions provide code reusability. Once the function is defined, we can use it whenever we need it, and as often as we need it.
- In the following case study, we re-look into top-down design in approaching an algorithmic problem.
- In the process, we encounter certain tasks that are similar, hence necessitating the creation of user-defined function.

A Simple "Drawing" Program

Problem:

- Write a program to draw a rocket ship, a male stick figure, and a female stick figure

rocket

Analysis:

- No particular input needed, just draw the needed 3 figures
- There are common shapes shared by the 3 figures

male

Design:

- Algorithm (in words):
  1. Draw Rocket ship
  2. Draw Male stick figure (below Rocket ship)
  3. Draw Female stick figure (below Male stick figure)

female

A Simple "Drawing" Program

- Rocket ship: Triangle + Rectangle + Inverted V
- male stick figure : Circle + Rectangle + Inverted V
- a female stick figure : Circle + Triangle + Inverted V

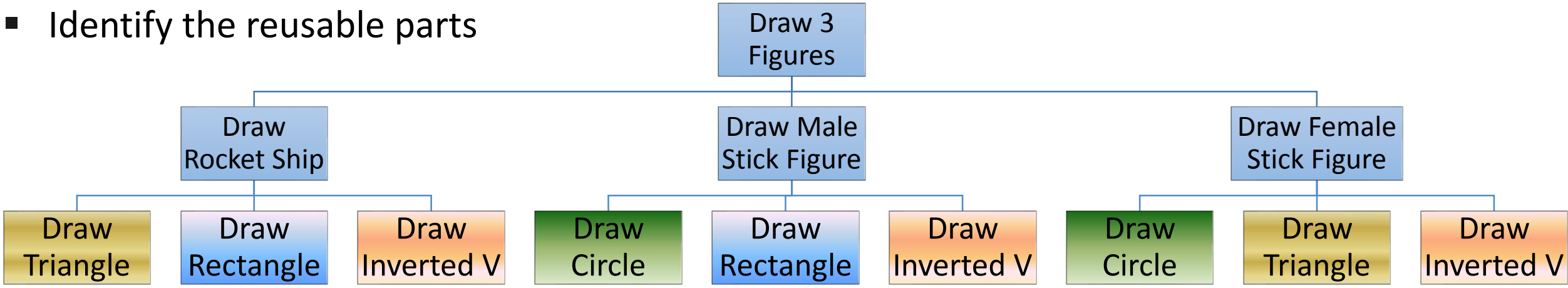
rocket

male

female

Design (Structure Chart):

- A documentation tool that shows the relationship among the sub-tasks
- Identify the reusable parts



A Simple "Drawing" Program

Implementation (partial program)

```c
#include <stdio.h>
void draw_rocket_ship();
void draw_male_stick_figure();
void draw_circle();
void draw_rectangle();

int main(void) {
   draw_rocket_ship();
   printf("\n\n");

   draw_male_stick_figure();
   printf("\n\n");

   return 0;
}
```

```c
void draw_male_stick_figure() {
   draw_circle();
   draw_rectangle();
   ...
}

void draw_circle() {
   printf("   **   \n");
   printf(" *    * \n");
   printf(" *    * \n");
   printf("   **   \n");
}

void draw_rectangle() {
   printf(" ****** \n");
   printf(" *    * \n");
   printf(" *    * \n");
   printf(" *    * \n");
   printf(" ****** \n");
}
```

Function Prototypes

- It is a good practice to put function prototypes at the top of the program, before the `main()` function, to inform the compiler of the functions that your program may use and their return types and parameter types.
- Function definitions to follow after the `main()` function.
- Without function prototypes, you will get error/warning messages from the compiler, as it assumes the default (implicit) return type of int for function `draw_circle`, which conflicts with the function header of `draw_circle ()` when the compiler encounters the function definition later

Default Return Type

- A ‘type-less’ function has default return type

```
 1 #include <stdio.h>
 2
 3 int main(void) {
 4     draw_circle() ;
 5     return 0;
 6 }
 7
 8 draw_circle() {
 9     printf("   **   \n");
10     printf(" *    * \n");
11     printf(" *    * \n");
12     printf("   **   \n");
13 }
```

```
warning: implicit declaration of function 'draw_cir
(due to absence of function prototype)
warning: return type defaults to 'int' ← line 8

In function 'draw_circle':
warning: control reaches end of non-void function
```

Default Return Type

- Tips
  - Provide function prototypes for all programmer-defined functions
  - Explicitly specify the function return type for all functions

```
1 #include <stdio.h>
 2
 3 void draw_circle() ;
 4
 5 int main(void) {
 6     draw_circle() ;
 7     return 0;
 8 }
 9
10 void draw_circle() {
11     printf("   **   \n");
12     printf(" *    * \n");
13     printf(" *    * \n");
```

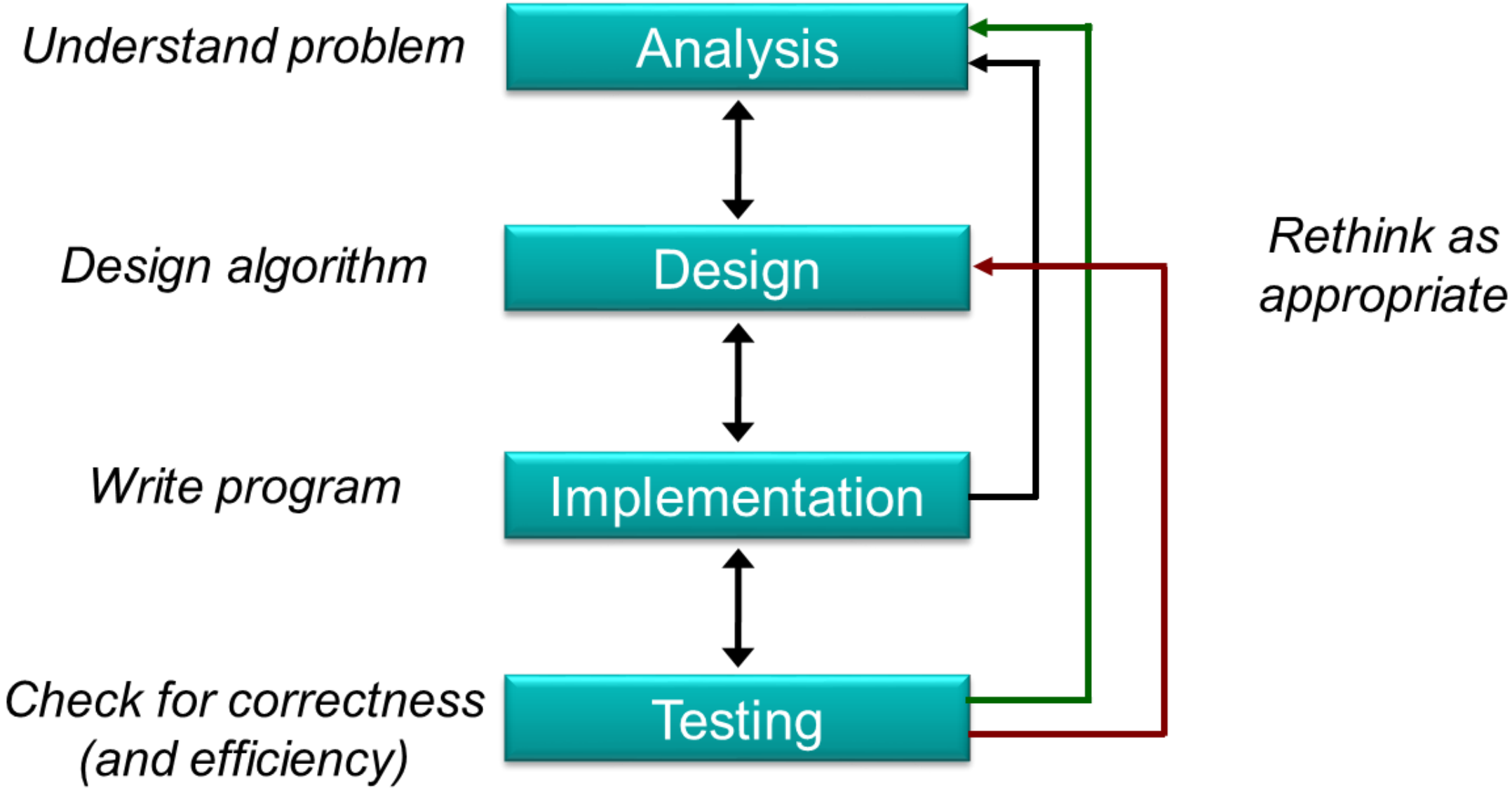
Writing Functions

- A program is a collection of functions (modules) to transform inputs to outputs
- In general, each box in a structure chart is a sub-problem which is handled by a function
- In mathematics, a function maps some input values to a single (possibly multiple dimensions) output
- In C, a function maps some input values to zero or more output values
  - No output: void func(…) { … }
  - One output, e.g., double func(…) { …; return value; }
  - More outputs through changing input values (through pass-by-reference)
- Return value (if any) from function call can (but need not) be assigned to a variable.

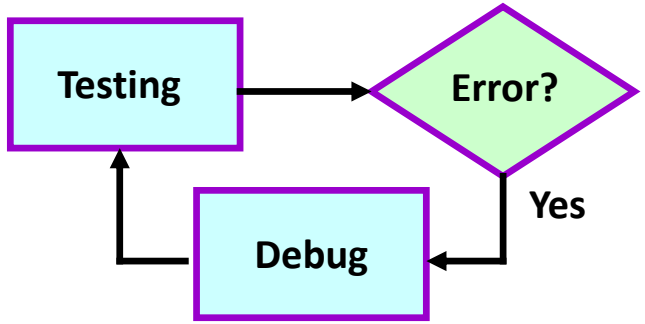
Writing Functions

- Use of functions allow us to manage a complex (abstract) task with a number of simple (specific) ones.
  - This allows us to switch between abstract and go to specific at ease to eventually solve the problem.
- Function allows a team of programmers working together on a large program – each programmer will be responsible for a particular set of functions.
- Function is good mechanism to allow re-use across different programs. Programmers use functions like building blocks.
- Function allows incremental implementation and testing (with the use of driver function to call the function and then to check the output)
- Acronym NOT summarizes the requirements for argument list correspondence. (N: number of arguments, O: order, and T: type).

Problem Solving in Programming



Testing and Debugging

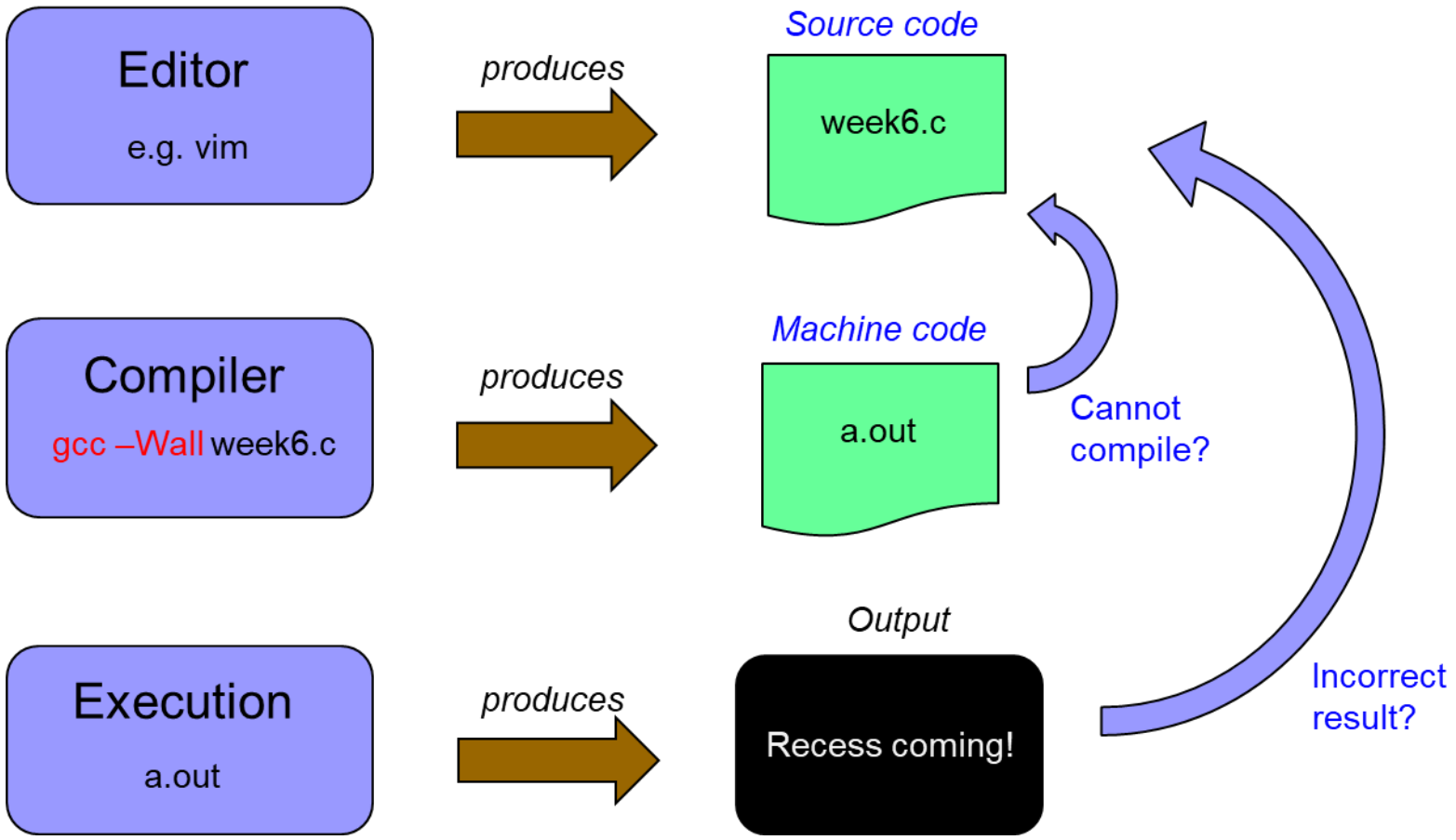


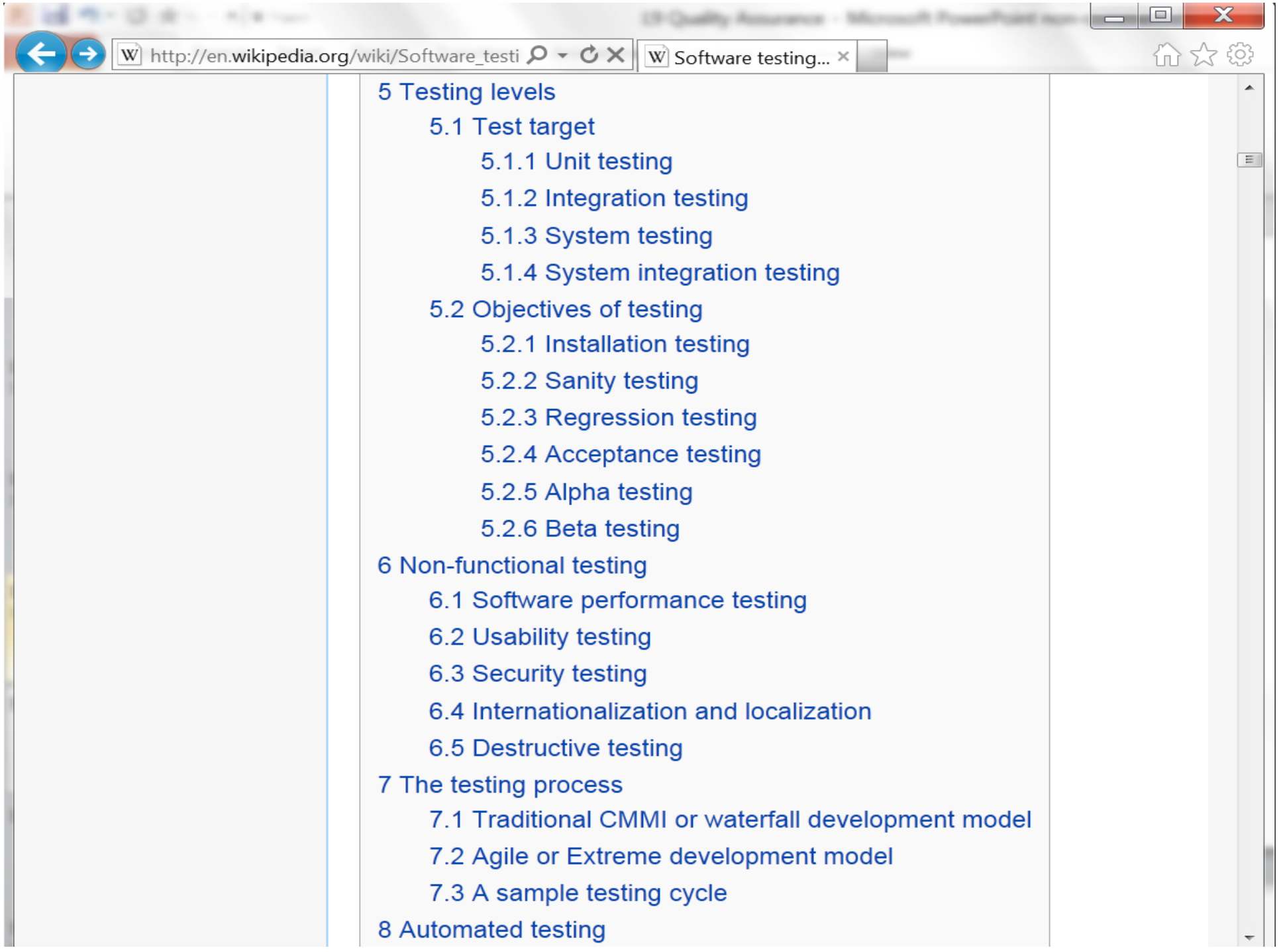
- Testing
  - To check if a code contains errors.
- Debugging 
  - To locate the errors (“bugs”) and fix them.
- After debugging, the program is not necessarily error-free.
  - It just means that whatever errors remain are harder to find. ☺

Programming Errors : Taxonomy (1/2)

- Syntax Error (and warning)
  - Reported by the compiler.
  - Program does not obey C grammar such as invalid expression, missing semi-colon, uninitialized variable, etc.
- Run-time Error
  - Reported at run-time.
  - Program terminates unexpectedly (i.e. crashes) due to illegal operation, such as dividing a number by zero or pointer exception.
- Logic Error
  - Program finish execution but doesn't produce expected result (wrong algorithm).

Programming Errors : Taxonomy (2/2)




http://en.wikipedia.org/wiki/Software_testi
Software testing...
5 Testing levels
5.1 Test target
5.1.1 Unit testing
5.1.2 Integration testing
5.1.3 System testing
5.1.4 System integration testing
5.2 Objectives of testing
5.2.1 Installation testing
5.2.2 Sanity testing
5.2.3 Regression testing
5.2.4 Acceptance testing
5.2.5 Alpha testing
5.2.6 Beta testing
6 Non-functional testing
6.1 Software performance testing
6.2 Usability testing
6.3 Security testing
6.4 Internationalization and localization
6.5 Destructive testing
7 The testing process
7.1 Traditional CMMI or waterfall development model
7.2 Agile or Extreme development model
7.3 A sample testing cycle
8 Automated testing

Windows

A fatal exception 0E has occurred at 0028:C0011E36 in VXD VMM(01) +
00010E36. The current application will be terminated.

* Press any key to terminate the current application.
* Press CTRL+ALT+DEL again to restart your computer. You will
  lose any unsaved information in all applications.

Press any key to continue

Infamous Programming Errors

- **Ariane 5 story (1996)**
  - **Cost**: $370 million
  - **Disaster**: Flight 501, which took place on June 4, 1996, was the first, and unsuccessful, test flight of the European Ariane 5 expendable launch system. As it was an unmanned flight, there were no victims, but the breakup caused the loss of four cluster mission spacecraft.
  - **Cause**: Due to an error in the software design (inadequate protection from integer overflow), the rocket veered off its flight path 37 seconds after launch and was destroyed by its automated self-destruct system.

Infamous Programming Errors

- **Mars Climate Crasher (1998)**
  - **Cost**: $125 million
  - **Disaster**: After a 286-day journey from Earth, the Mars Climate Orbiter fired its engines to push into orbit around Mars. The engines fired, but the spacecraft fell too far into the planet's atmosphere, likely causing it to crash on Mars.
  - **Cause**: The software that controlled the Orbiter thrusters used imperial units (pounds of force), rather than metric units (Newtons) as specified by NASA.

DBS Bank suffers massive IT failure

Tags: DBS Bank IBM IT failure Singapore Monetary Authority system outage

By **Sumner Lemon** | Jul 7, 2010

1 comments Facebook LinkedIn Delicious Digg Email Print

DBS, one of Singapore's biggest banks suffered a major IT outage on Monday that took down its computer systems for seven hours.

The outage knocked DBS Bank's back-end computer systems offline, leaving its customers unable to withdraw cash from ATM machines on Monday morning.

"We first knew of the problem at 3:00 a.m. (Singapore time) and by 10:00 a.m., all our branches and ATMs were fully

DBS Bank suffers massive IT failure

Tags: DBS Bank IBM IT failure Singapore Monetary Authority system outage

By Sumner Lemon | Jul 7, 2010

1 comments Facebook LinkedIn Delicious Digg Email Print

said David Gledhill, managing director and head of group technology and operations at DBS,

It wasn't immediately clear why the bank's backup systems didn't prevent the outage.

DBS signed a 10-year outsourcing deal with IBM valued at S$1.2 billion in November 2002. Th

The DAO was launched on April 30th, 2016, and had raised over $100 million by May 16th, 2016.

The DAO (Decentralized Autonomous Organization) was proposed to operate like a venture capital fund for cryptocurrencies in decentralized blockchain.

It had become the largest contract on the Ethereum blockchain. There was a lot of hype in the Ethereum community because the vision of Ethereum and the use of smart contracts were becoming a reality. The news had hit the front page of The New York Times and the community /r/ethtrader rejoiced: ("ARE YOU SERIOUS!?!? FRONT PAGE!!!")


"All the News That's Fit to Print"

The New York Times

Late Edition

Today, clouds and sunshine, a shower or thunderstorm, high 69. Tonight, an evening shower, low 56. Tomorrow, a thunderstorm, high 70. Weather map appears on Page 20.

VOL. CLXV ... No. 57,240 © 2016 The New York Times NEW YORK, SUNDAY, MAY 22, 2016 $5.00

ethereum wiki ETHEREUM BITCOIN ALTCOINS PRICES NEWS

The DAO was launched on April 30th, 2016, and had raised over $100 million by May 16th, 2016.

www.ethereumwiki.com/ethereum/night-dao-attack-pictures/

The DAO crowdfund went smoothly and contracts and proposals were created and voted on yet the warnings before the DAO launch were there. Posts on reddit issued warnings about the security firm who did a poor audit on the code. And hours before the close of the token sale, Vitalik Buterin tweeted this:

contracts were becoming a reality. The news had hit the front page of The New York Times and the community /r/ethtrader rejoiced: ("ARE YOU SERIOUS!?!? FRONT PAGE!!!")


"All the News That's Fit to Print"

The New York Times

Late Edition

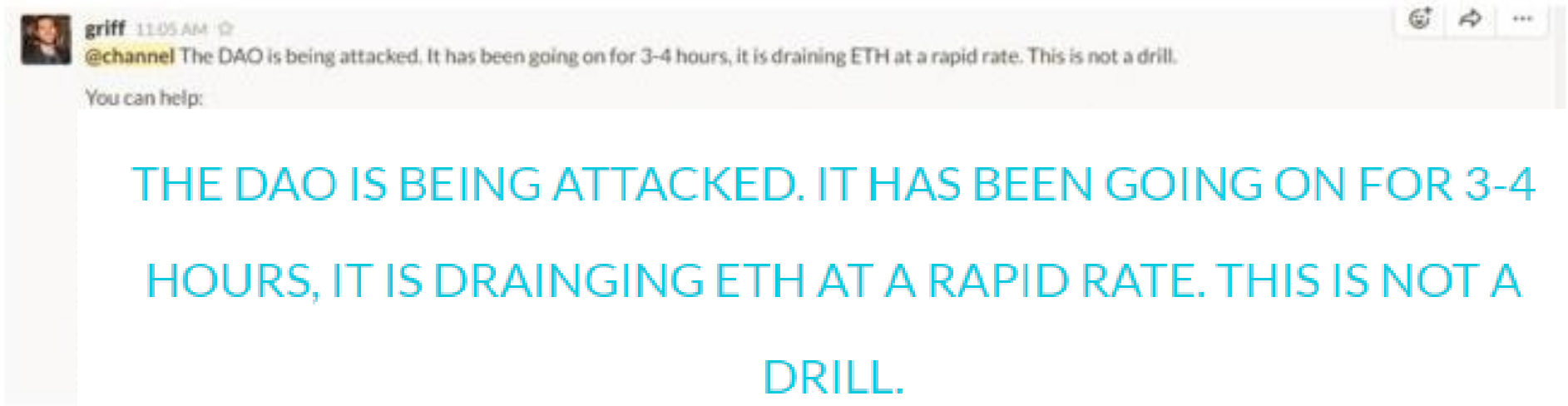
Today, clouds and sunshine, a shower or thunderstorm, high 69. Tonight, an evening shower, low 56. Tomorrow, a thunderstorm, high 70. Weather map appears on Page 20.

VOL. CLXV ... No. 57,240 © 2016 The New York Times NEW YORK, SUNDAY, MAY 22, 2016 $5.00

www.ethereumwiki.com/ethereum/night-dao-attack-pictures/

On June 17th, 2016, a flaw in the code allowed someone to move 3,641,694 Ether into a sub-DAO that they alone control.

Early into the attack, community organizer for slock.it and The DAO, Griff Green began messaging alerts to the community Slack group asking for help and that it wasn't a drill.



THE DAO IS BEING ATTACKED. IT HAS BEEN GOING ON FOR 3-4 HOURS, IT IS DRAINGING ETH AT A RAPID RATE. THIS IS NOT A DRILL.

Days later, the supposed hacker releases a note which can be read here and concludes with:

I HOPE THIS EVENT BECOMES AN VALUABLE LEARNING EXPERIENCE FOR THE ETHEREUM COMMUNITY AND WISH YOU ALL THE BEST OF LUCK.

The attacker has never been found and search to find him/her continues today.

https://www.coindesk.com/dao-attacked-code-issue-leads-60-million-ether-theft/

DAO Attack Explained

```
3  contract IGlogalBank {
4
5      mapping (address => uint) p
6
7      function IGlogalBank() publ
8      }
9
10     function getBalance() const
11         return this.balance;
12     }
13
14     function getAddress() const
15         return this;
16     }
17
18     function deposit() public p
19         if (customers[msg.sende
```

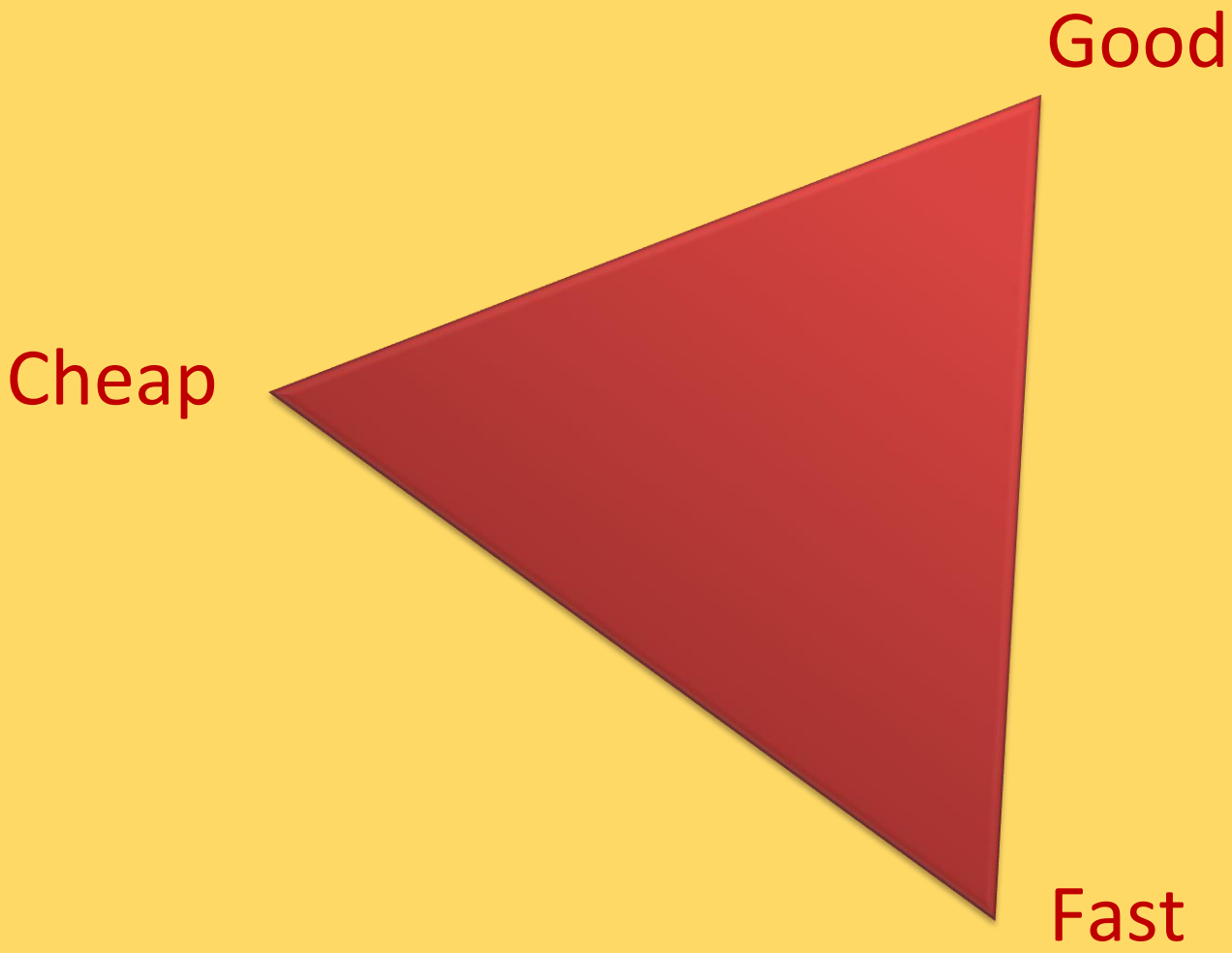
```
contract Attacker {
    address public target;

    function Attacker(address _target) public payable
        target = _target;
    }

    function() public payable {
        if (target.call(bytes4(keccak256("withdraw()")

        }
    }

    function getBalance() constant public returns (uin
        return this.balance;
    }

    function getAddress() constant public returns (add
```

Difficulty of Software Testing

- The difficulty in software testing stems from the complexity of software: we can not completely test a program of moderate complexity.

- Software testing is a trade-off between budget, time and quality.

Good, cheap, fast: select any two



Testing & Debugging in Small-Scale (1/3)

- Philosophical notes on program design, debugging and testing
  - A good design is important
    - A good design results in a high quality program. A low quality design cannot be “debugged into” high quality.
  - Do not optimize for speed too early
    - It is possible to make a correct program run faster.
    - It is much more difficult to make a fast (but wrong) program run correctly.

Testing & Debugging in Small-Scale (2/3)

- Tips:
  - Start off with a working algorithm.
  - Simplify your design and logic.
  - Incremental coding, test early, fix bugs once you find them.
  - Recognize common errors (e.g. integer division).
  - Recompile the program for every single change.
  - Test boundary values.
    - Example: in prime number test, did you test your program with boundary values 1 and 2?
  - Explain the bug to someone else.
  - Pause for a while and take a deep breadth!

Testing & Debugging in Small-Scale (3/3)

- How to do debugging?

- Manual walkthroughs
  - Verbal walkthroughs.
  - Tracing with pencil-and-paper.

- Use printf statements to check intermediate results
  - May provide information on:
    - Which functions have been called.
    - The value of parameters and local variables at strategic points

Example

```
  int number, digit2, digit3, digit4, digit5, digit6;
  int step1, step2, step3;

  ...

  digit6 = number%10; number /= 10;
  digit5 = number%10; number /= 10;
  digit4 = number%10; number /= 10;
  digit3 = number%10; number /= 10;
  digit2 = number%10;

  step1 = digit2*2 + digit3*6 + digit4*2 + digit5*4 +
digit6;


  step2 = step1 % 13;


  step3 = 13 - step2;


 ...
```

Using a Debugger

- Debugger usually provides the following
  - Stepping
  - Breakpoint
  - Watches (inspecting variables)

- On UNIX, you may want to use gnu debugger gdb.
  - See video http://www.youtube.com/watch?v=Z6zMxp6r4mc