说明会
Briefing Session

台大盃2021 - 全国中学生程式设计竞赛

NTU Cup 2021- National Secondary School Programming Contest

拉曼大学 资讯与通讯科技学院 刘绍瑜

Liew Soung Yue

Faculty of Information and Communication Technology

Universiti Tunku Abdul Rahman

Process Flow

Workshop + Qualification Test

Every contestant must join a workshop, associated with a Qualifying Test.

The purpose of the Qualifying Test is to shortlist the qualified contestants for the Final Competition.

The first two workshops will be held on **31/5/21 (Mon)** and **7/6/21 (Mon)**

Final Competition

Date: 30/10/2021 (Sat)

Time: 08:30 – 13:00

Overview of Workshop

	Time	Details
Session 1 (8:30AM – 10:00AM)	8:30AM – 9:00AM	Basic ICT **(30 minutes)**
	9:00AM – 10:00AM	Programming Concepts 1. Basic Data Type and Input/Output **(15 mins)** 2. Control Structure (if…else / Switch…case) **(15 mins)** 3. Iteration (for, while, do…while) **(15 mins)** 4. User defined data type (Array) **(15 mins)**
Session 2 (10:00AM – 11:00AM)	10:00AM – 11:00AM	Real-World Application ICT – invited speaker **(1 hour)**

Overview of Workshop (cont’d)

	Time	Details
Session 3 (11:00AM – 12:00PM)	11:00AM – 11:30AM	1. Briefing on rules and regulations of competition **(15 mins)** 2. How to use competition platform including IDE installation **(15 mins)**
	11:30AM – 12:00PM	1. Demo on one sample question in C++ **(15 mins)** 2. Demo on one sample question in Java **(15 mins)**
Session 4 (12:00PM – 1:00PM)	12:00PM – 1:00PM	Qualifying Test Q&A in UTAR Web-based Learning Portal **(1 hour)**

Workshop

Qualification Test

Way of Conduct	MCQ (选择题), random questions generated from question bank
Conduct Mode	Online
Participant	Individual
Duration	1 hour
No. of Question	20 questions (including 5 **HOT** questions)
Types of Question	General Knowledge of Programming: Arithmetic, Logic, Algorithm, Coding (C/C++/Java)
Passing Mark	70%
Expected Outcome	20 teams to participate in Final Competition

Sample Qualifying Questions

Q1. ___________ is a set of step-by-step instructions designed to perform a specific task or to solve an instance of problem.

a) **Algorithm**
b) **Complexity**
c) **Pseudocode**
d) **Data Structure**

Q2. Find the solution for the following recurrence relation $a_n = 3n^2 \times a_{n-1}$ **, where** $a_0 = 1$

a) $a_n = (n!)^2$
b) $a_n = 3(n!)^2$
c) $a_n = 3^n(n!)^2$
d) $a_n = 3^{n+1}(n!)^2$

Sample Qualifying Questions (cont’d)

Q3. What does the following piece of code do?

```
for (int i = 0; i < array.length-1; i++)
{
  for (int j = i+1; j < array.length; j++)
  {
     if( array[i].equals(array[j]) )
     {
       System.out.println(array[i]);
      }
   }
}
```

a) Print the unique elements in the array

b) Print the duplicate elements in the array

c) Print the element with maximum frequency

d) Print the element with minimum frequency

Final Competition

Conduct Mode	Physical
Participant	3 students per team
Duration	3 hours
No. of Question	8 questions
Programming Language	C/C++, Java
Resources	- Each team will be provided **ONE** computer with a minimum spec of Intel Core i3 4th Gen, 8GB RAM, 120GB SSD. - Compiler: Dev-C++(IDE) MinGW / Visual Studio Code - Candidates are **NOT ALLOWED** to use other computing devices. - Pens and papers will be provided.

Sample Final Question

Q1. The Day of the Week – Gregorian Calendar:

Description

The Gregorian Calendar （公历）is the most used solar dating system in the world. It was introduced in October 1582 by Pope Gregory XIII, and it spaces **leap years （闰年）** to make its average year 365.2425 days long, that is determined by the Earth's revolution around the Sun.

The rule for leap years is

- **The year must be divisible by 4**
- If the year can also be divided by 100, **it is *NOT* a leap year, UNLESS**
- The year is also divisible by 400. **Then it *is* a leap year.**

For example, the years 1700, 1800, and 1900 are not leap years, but the years 1600 and 2000 are.

Sample Final Question (cont’d)

In a normal year, the number of days in each month is as follows:

Month 1 (January):	31 days
Month 2 (February):	28 days
Month 3 (March):	31 days
Month 4 (April):	30 days
Month 5 (May):	31 days
Month 6 (June):	30 days
Month 7 (July):	31 days
Month 8 (August):	31 days
Month 9 (September):	30 days
Month 10 (October):	31 days
Month 11 (November):	30 days
Month 12 (December):	31 days

(in total 365 days)

However, in a leap year, the month of February has 29 days instead of 28.

Sample Final Question (cont'd)

On the other hand, a week is composed of seven days, namely (in sequence) "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", and "Saturday".

The same sequence repeats from a week to the next week.

Question

Given that the first day of the Gregorian calendar was **15th October 1582**, which was a **Friday**, write a program to accept a specific date with the format **[Day, Month, Year]** as the input, and calculate **which day of the week** corresponding this date as the output.

Sample Final Question (cont'd)

Requirements:

- Your program must fulfill the following:
  1. Input format [Day, Month, Year], where
     1≤ Day ≤31,
     1≤ Month ≤12, and
     1582 ≤ Year ≤ 9999
  2. If the input is a valid date, the correct output is **the day of the week** corresponding to the date.
  3. If the input is not a valid date, the correct output is **"This is not a valid date. Please key in another date"**, then return to (i).

Examples:

- If the input is [17, 4, 2021], then the correct output is "**Saturday**".
- If the input is [29, 2, 2100], then the correct output is "**This is not a valid date. Please key in another date**".

Workshop | Qualification Test | **Final Competition**

Format of Competition	• The competition will consist of 8 questions. • The questions may be solved in any order.
Submission Procedures	• For each question, participants can submit their solution for judging whenever they are ready. • All solutions will be submitted online, in the form of executable file (C++: *.exe* file or Java: *.jar* file). • Timestamps are electronically recorded for each submission.
Judging Criteria	• The judges will run the executable file with some test input data, and check whether the output is correct. • Competitors will receive notice from the judges, indicating whether or not the solution is correct. • Each submitted solution that is correct will be counted. • Each team is allowed to resubmit the solution multiple times, however penalty is imposed for each incorrect submission.

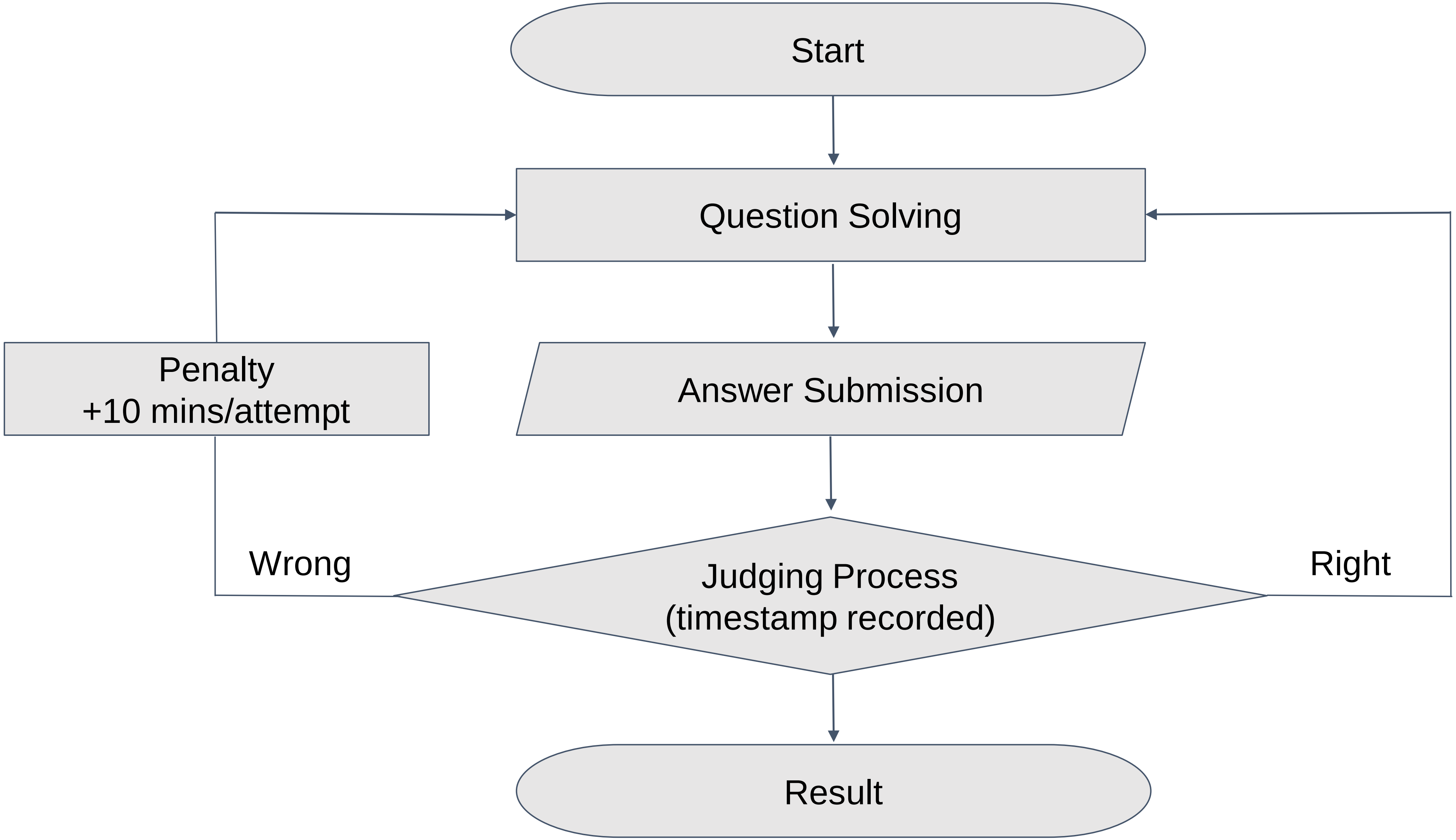
Scoring

- The winning team will be the team which answers the most questions correctly in the shortest amount of time.
- The total time is the sum over all solutions of the times at which the participant submitted the solutions correctly, plus additional penalty time for incorrect submissions.
- If a participant submitted a solution correctly several times, the first time will be counted.
- Each incorrect submission for a question that the team eventually gets correct will incur a penalty of **10 MINS PER ATTEMPT**.
- Incorrect solutions to questions which are never answered correctly do not add to the total time, neither do incorrect solutions submitted after correct solutions.

Example 1: Team 1 answers Question A incorrectly after 30 minutes, and correctly after 35 minutes. Also, the team answers Question B correctly after 62 minutes, and incorrectly after 64 minutes. The total time is 35 + 10 + 62 = 107 minutes.

Example 2: Team 2 answers Question B incorrectly after 10 minutes, and correctly after 18 minutes. Also, the team answers Question A incorrectly after 60 minutes, and correctly after 65 minutes. The total time is 18 + 10 + 65 + 10 = 103 minutes.

Competition Flow



Start

Question Solving

Answer Submission

Judging Process
(timestamp recorded)

Wrong

Penalty
+10 mins/attempt

Right

Result