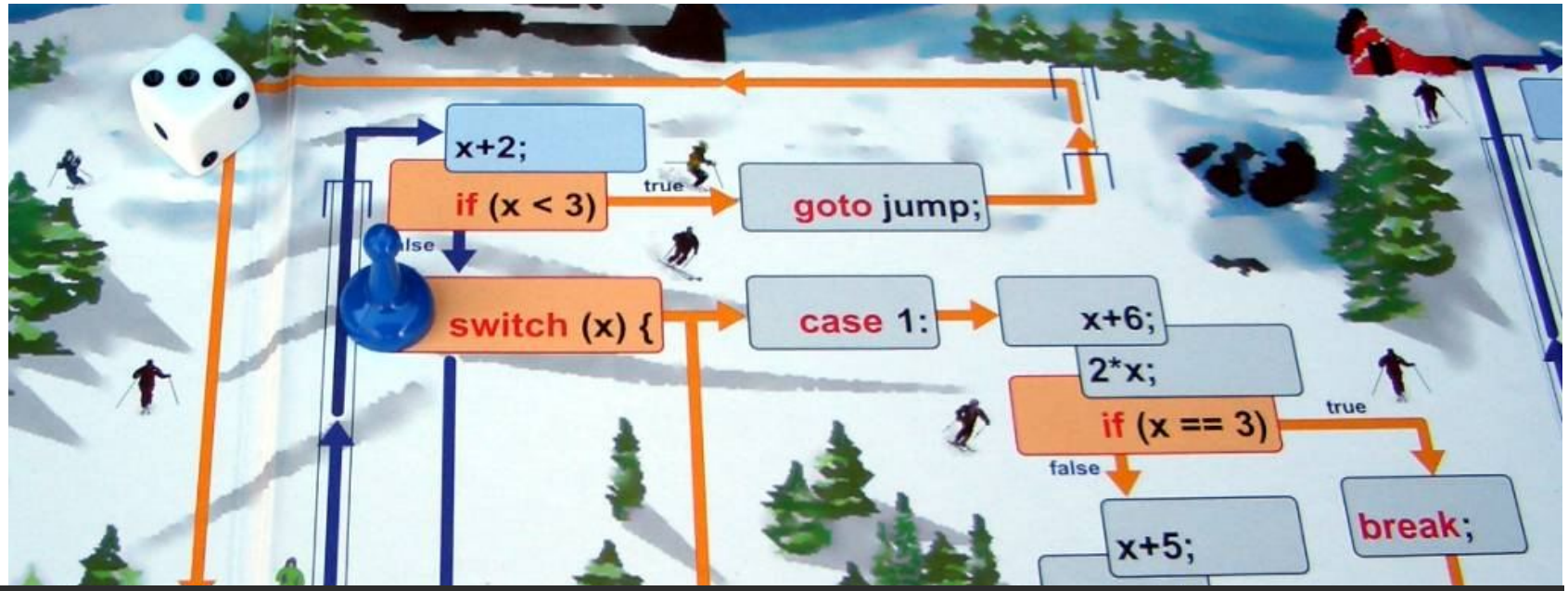


Data types / Operators / Math / Mouse Input

Instructor: Neng-Hao (Jones) Yu
Time: Mon. 6:10 – 9:10pm
Place: 商院大樓 260509
Course website: http://programming101.cs.nccu.edu.tw

Recaps

- How to write a program?
- Drawing APIs (Application Programming Interface)
  - `functionName(parameters)`
- Variables
- Naming conventions

Operators and operands

operands

```
int sumNumber = 2 + 3 * 4;
```

operators

```
// sumNumber is now 14

println(5 + 5); // 10

println("Hello"+"World"); // HelloWorld

println("5" + "5"); // ?
```

operator overloading

Arithmetic operators

+	Addition : Adds numeric expressions
-	Subtraction : Negates or subtracts numeric expressions
*	Multiplication : Multiplies two numerical expressions
/	Division : Divides expression1 by expression2
%	Modulo : Calculates the remainder of division
++	Increment : Adds 1 to a numeric expression
--	Decrement : Subtracts 1 from a numeric expression

Modulo (%) → **NOT** Percentage

- Calculates the remainder when one number is divided by another.
  - `int a = 5 % 4;  // Sets 'a' to 1`
  - `int row = 33 % 10;  // calculate row position`
  - `int col = 33 / 10;  // calculate column position`
- Modulo is extremely useful for keeping numbers within a boundary such as keeping a shape on the screen.
  - `int newPosition = x % width;`

Type of operators

- Interfix: 5 + 3 - 2
- Prefix: -55
- Postfix: foo++ ➔ foo = foo + 1

- Unary → ++ (one operand)
- Binary → + - * / (two operands)
- Ternary → ?: (three operands)

Prefix vs Postfix

```
int xNum = 0;

println(++xNum); // 1

println(xNum); // 1


int yNum = 0;

println(yNum++); // 0

println(yNum); // 1
```

Compound assignment operators

```
var foo=5;

foo += 5;   // foo = foo + 5;

-=

*=

/=

%=
```

Operator precedence

```
int first = (1 + 2) * 7;

int second = 5 * --first % 3;

println(second);

println(first);
```

Name	Symbol	Examples
Parentheses	`()`	`a * (b + c)`
Postfix, Unary	`++ -- !`	`a++ --b !c`
Multiplicative	`* / %`	`a * b`
Additive	`+ -`	`a + b`
Relational	`> < <= >=`	`if (a > b)`
Equality	`== !=`	`if (a == b)`
Logical AND	`&&`	`if (mousePressed && (a > b))`
Logical OR	`||`	`if (mousePressed || (a > b))`
Assignment	`= += -= *= /= %=`	`a = 44`

Data types

- The way values of that type can be stored.
  - `int: integer`
  - `float: floating-point number`
  - `char: single character`
  - `String: string of words`
- Advantages:
  - less bugs
  - efficient memory allocation

Data types

- Size and value range of data types:

	Name	Description	Range of values
32bits	*int*	Integers (whole numbers)	–2,147,483,648 to 2,147,483,647
32bits	*float*	Floating-point values	–3.40282347E+38 to 3.40282347E+38
1bit	*boolean*	Logical value	true or false
16bits	*char*	Single character	A–z, 0–9, and symbols
	String	Sequence of characters	Any letter, word, sentence, and so on
	PImage	PNG, JPG, or GIF image	N/A
	PFont	VLW font; use the Create Font tool to make	N/A
	PShape	SVG file	N/A

Type conversions

- **Implicit** conversion, also called **coercion**, is sometimes performed at runtime. It happens in
  - In assignment statements
  - In expressions using certain operators, such as the addition (+) operator

- **Explicit** conversion, also called **casting**, occurs when your code instructs the compiler to treat a variable of one data type as if it belongs to a different data type.

```
int a;                  b= a / 2.0;

float b;                // or

a= 55;                  b= (float)a / 2;

b= a / 2;               // or

println(b);             b= float(a) / 2;

// what is the value of b?
```

http://processing.org/examples/datatypeconversion.html

Constants

- Must be initialized
- Cannot change the data in it
- Constant names are all caps with words separated by an underscore, e.g.

```
final float MILES_KM_CONVERSION_VALUE = 1.61;
final float PI = 3.14;
```

```
// convert mile to km
final float MILES_KM_CONVERSION_VALUE = 1.61;

float km = 30 * MILES_KM_CONVERSION_VALUE;

println(km);
```

Exercise (5mins):
Write a program to calculate the area of a circle given a radius

Build-in constants and variables

```
size (200,200);
float x = width/2;
float y = height/2;
float d = width * 0.8;
arc(x, y, d, d, 0, QUARTER_PI);
arc(x, y, d-20, d-20, 0, HALF_PI);
arc(x, y, d-40, d-40, 0, PI);
arc(x, y, d-60, d-60, 0, TWO_PI);
```

Static mode vs Dynamic mode

- Dynamic mode:

run once at the beginning

```
void setup() {
  size(300, 300);
}
```

update forever

```
void draw() {
  //background(0);
  ellipse(mouseX, mouseY, 80, 80);
}
```

build-in variables

Exercise (6mins)

- draw a ball
- keep it moving from left to right

- Bonus: moving alien

Math

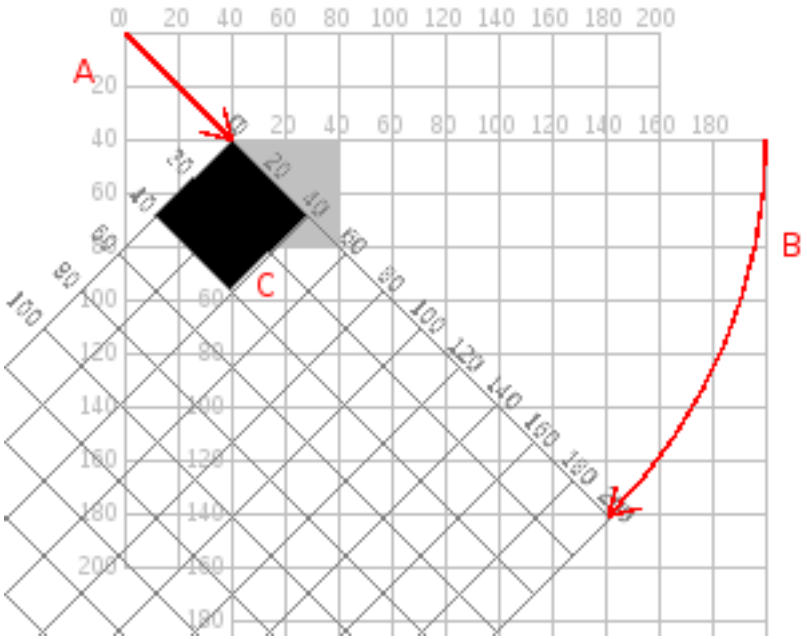
- abs()
- ceil()
- floor()
- round()
- sq()
- pow()
- min()
- max()
- norm()
- constrain()
- map()
- dist()

http://processing.org/reference/

https://gist.github.com/jonesfish/7ab805f1da4f7ba1bf6a

Math

- `translate()`
- `rotate()`
- `scale()`
- `pushMatrix()`
- `popMatrix();`



https://gist.github.com/jonesfish/3e93c7c11b08795654bc

Math

- `radians()`
- `degrees()`
- `sin()`
- `atan2(y, x)`

Exercise (5mins) : draw a sin wave

https://gist.github.com/jonesfish/1c6b2c76652e8a963f91

Random Numbers

```
float r = random(50);

// generate a random number from 0.0 to 50.0

// starting at 0, and up to(but not including) 50.0


int dice = int(random(6));

// roll a dice
```

https://gist.github.com/jonesfish/6b462cce0f24f92cfaf1

Assign1 – Slot Machine

- A: 出現 777的機率為 10%
- B:
  - 每次按下 "Roll" 扣50分
  - 按下 "Stop" 後以水果重複出現次數計算倍率分數（該水果數量＊該水果的基本分數）
- C:
  - 不同水果代表不同分數
  - 開始時有500分，按下 "Roll" 啟動遊戲
  - 按下 "Stop" 後隨機指派九宮格內的水果
  - 結算分數並顯示（加總每一格水果所代表的分數）


Slot Machine
Score : 0
7
7
7
Roll

Slot Machine APIs

fid: 0	1	2	3	4	5
					
score: 60	10	20	30	40	50

```
mahcine.setSlotFruit(int x, int fid);
mahcine.getSlotScore(int x);
machine.getFruitCount(int fid);
machine.probability(float p);
```