Computer Programming Lab 1

2023/10/3 Tim Chen

Outline

- Data Types
- Operators
- `for` Loop
- Pre-increment v.s. Post-increment

Data Types

1. Implicit type conversion

```c
int number = 10;
printf("%d\n", number / 3); // 3
```

Data Types (cont.)

2. Explicit type conversion

```
int num = 3;
float fnum = 3.5;
float sum;
sum = (float)num + fnum;
```

Operators

Arithmetic Operator

- `=`

  e.x. `x = 1`, `y = 2`
- `+`, `-`, `*`, `/`

  e.x. `x = a + 1`, `x /= 2`
- `%` Module operator

  e.x. `x = 5 % 3`
- `++`, `--` Increase/Decrease operator

  e.x. `i++`, `--j`

Operators (cont.)

Relational Operator

- `==`, `!=` Check if two operands are equal

  e.x. `a == b`, `x != y`
- `>`, `<` Check if value of left operand is greater/less than the right

  e.x. `a < b`
- `>=`, `<=` Check if it's greater/less than or equal to

  e.x. `a >= b`

Operators (cont.)

Logical Operator

- `&&` Logical AND

  e.x. `a && b`
- `||` Logical OR

  e.x. `x || y`
- `!` Logical NOT

  e.x. `!(a && b)`

`for` Loop

Recap `while` loop

```
int product = 2;
while ( product <= 100) {
   product = 2 * product;
} /* end while */
```

`for` Loop (cont.)

```
for (initialization; condition; update) {}
```

e.x.

```
for (int i = 0; i <= 100; ++i) {
    printf("%d", i);
} /* end for */
```

`for` Loop (cont.)

```
for (int i = 0; i <= 100; ++i) {
   /* do something in for loop */
} /* end for */
```

```
int i = 0;
while ( i <= 100 ) {
   /* do something in while loop */
   ++i;
} /* end while */
```

Pre-increment v.s. Post-increment

Pre-increment

- e.x. `++a`

```c
int a = 0;
printf("%d\n", ++a); // 1
```

Pre-increment v.s. Post-increment (cont.)

Post-increment

- e.x. `a++`

```c
int a = 0;
printf("%d\n", a++); // 0
printf("%d\n", a); // 1
```

Pre-increment v.s. Post-increment (cont.)

```
int a = 0; // global variable
```

```
int aplusplus () {
    int tmp = a;
    a = a + 1;
    return tmp;
}
```

```
int plusplusa () {
    a = a + 1;
    return a;
}
```

Which one is faster ?

Thank you