ESc101 : Fundamental of Computing

I Semester 2008-09

Lecture 11

Types

- Type conversion during assignment
- Type casting
- Many examples : $-)$

Type conversion during assignment statement

type conversion in Assignment

`i` : a variable of type `t1`

`E` : an expression (of possibly different type).

what happens when we execute statement `i = E;` **?**

Steps :

1. Evaluate the expression `E`. Let val be its value and `t2` be its type.
2. If `t1` is **wider** than that of `t2` :

   $\Rightarrow$: type of val gets promoted to `t1` and copied to `i`;

Otherwise **compilation error !!**

type casting

type cast : *an operator to explicitly convert the type of an expression*

`t` : a numeric data type.

E : an expression (possibly of type **wider** than `t1`).

`(t1)(E)`

convert the type of (value of) E into `t1`.

Note : type cast has higher priority than any arithmetic operator.

type casting from wider to narrower range

Information may get lost

1. (`int`)(1.34) : loss of information (*fractional part gets lost*)

2. (`byte`)(1234): loss of information

3. (`byte`)(123) : no loss of information

type casting from narrower to wider

Information may get lost here also

1. `long(123421)` : no loss of information.
2. (float)12345678998765432 1L : loss of information

   is equal to 1.2345679E17, which is $1.2345679 * 10^{17}$
3. (long)((float) 12345678998765432 1L) : loss of information

   is equal to 12345679051908 7104

Reason : the number of floating points possible using 32 bits float is less than the number of integers represented by long.

type casting in asignment statements

```
int i;      float x = 4;      float y=3;

                  i = (int)(x/y);
```

What is value of `i` ?

type casting in asignment statements

```
int i;      float x = 4;      float y=3;

                  i = (int)(x/y);
```

the value of `i` is 1.

(Revisiting) Assigning literals to numeric variables

```
x = C;
```

x is a variable of numeric data type, C is a literal of numeric type.

- *if* x *is integer data type :*

  If C is integer literal, follow the same rule mentioned in Lecture 10,

  else compilation error.

- *if* x *is floating point data type :*

  Same as assigning arithmetic expression to float, so

  follow the rule of type conversion inassignment (slide 3)

Summary of steps to evaluate expression

Evaluating expression `(t)`(`E`)

1. parenthesize E
2. replace variables by their values
3. evaluate the expression (with type cast operator being of maximumprecedence)
4. let `val` be the value of expression.
5. apply type cast operator `(t)` on `val`

Summary of execute assignment statement

```
i = (t)(E)
```

1. first evaluate `(t)`(`E`)
2. let `value` be the value of `(t)`(`E`).
3. follow the rule of type conversion (**slide 3**) and copy `value` into `i`.

For examples and practice problems on types and expression evaluations, please go through the file practice.pdf available on course website