Meeting 12: Data types and Polymorphism

Today

Qur 3

Data types and Polymorphism (H941)

P₀ programs all syntactically valid programs were type ok

P₁ programs there are syntactically valid programs that are type inconsistent
generated code:

```
if check to see if type ok then
do the op
else
exit (-1)
```
P1

input C  - still just input integers
print    - print all types
print [1, 2]

Compiler

P1

T

print (2 if input() else [1, input()])

Flatlining

\[
\text{tmp2 = input(C)}
\]

\[
\text{if tmp2 \geq}
\]

\[
\text{tmp0 = input(C)}
\]

\[
\text{tmp1 = 2 + tmp0}
\]

\[
\text{else}
\]

\[
\text{tmp3 = input(C)}
\]

\[
\text{tmp1 = [1, tmp3]}
\]

\[
\text{if statement needed}
\]

\[
\text{tmp5 = tmp1 if tmp2 else tmp5}
\]

\[
\text{print tmp5}
\]
print \(1 + 3 + 4\) ( if else )

\(+ ( \text{if else} )\)

---

What is Polymorphism?

Multiple data types and a variable can hold any of them.

- Integer — 32-bit value that fits in a register
- List
- Array
- Dictionary — hash table
- Pointers to these things

A value is pointer or integer.

Tell what type a value is at runtime.
Possible Types

`INT`, `BOOL`, `LIST`, `DICT`

```
struct {
    enum INT, BOOL, LIST, DICT {
        tag;
    }

    union {
        int i;
        bool b;
        PyObject * pobj;
        gpointer * list;
    }

    PyObject * obj;
}
```

32-bit word

3

```
\[ \begin{array}{c}
\text{00 INT} \\
\text{01 BOOL} \\
\text{11 BIG (LIST, DICT)}
\end{array} \]
```