

Types, Part I

Programming Languages

CS 214



Types

A *type* is _____.

Examples from C++:

- The _____ type:

$V = \{\text{INT_MIN}, \dots, -1, 0, 1, \dots, \text{INT_MAX}-1, \text{INT_MAX}\}$

$O = \{\ll, \gg, +, -, *, /, \%, =, ++, --, \dots\}$

- The _____ type:

$V = \{\text{NUL}, \dots, '0', \dots, '9', \dots, 'A', \dots, 'Z', \dots, 'a', \dots, 'z', \text{DEL}\}$

$O = \{\ll, \gg, =, ++, --, \text{isupper}(), \text{islower}(), \text{toupper}(), \dots\}$

- The _____ type:

$V = \{\text{"", "A", "B", "C", \dots, "AA", "AB", "AC", \dots, "AAA", \dots}\}$

$O = \{\ll, \gg, +, +=, [], \text{find}(), \text{substr}(), \dots\}$



Fundamental Types

Let's assume the existence of some basic types:

Name	V	C++	Ada	Smalltalk	Lisp
_____	false, true	<i>bool</i>	<i>boolean</i>	<i>Boolean</i>	<i>boole</i>
_____	the set of chars	<i>char</i>	<i>character</i>	<i>Character</i>	<i>character</i>
_____	the integers	<i>int</i>	<i>integer</i>	<i>Integer</i>	<i>integer</i>
_____	the reals	<i>double</i>	<i>float</i>	<i>Float</i>	<i>real</i>

Given these, new types can be created via _____.

Each constructor has 3 components:

- The syntax used to denote that constructor;
- The set of elements produced by that constructor; and
- The operations associated with that constructor.



Set Constructor I: _____

The product constructor is the basis for _____.

- The product of two sets A and B is denoted _____.
- $A \times B$ consists of all ordered pairs (a, b) : $a \in A, b \in B$.
 $A \times B \times C$ consists of all ordered triples (a, b, c) : $a \in A, b \in B, c \in C$.
 $A \times B \times \dots \times N$ consists of all ordered n -tuples (a, b, \dots, n) :
 $a \in A, b \in B, \dots, n \in N$.

Example: the set _____ has 256 elements:

$\{ \dots, (\text{true}, 'A'), (\text{false}, 'A'), (\text{true}, 'B'), (\text{false}, 'B'), \dots \}$.

- Operations associated with product are the _____ operations:
 - *first*, applied to an n -tuple (s_1, s_2, \dots, s_n) returns _____.
 - *second*, applied to an n -tuple (s_1, s_2, \dots, s_n) returns _____.
 - *nth*, applied to an n -tuple (s_1, s_2, \dots, s_n) returns _____.



Product Example: C++ structs

```
struct Student
{
    int id;
    double gpa;
    char gender;
};
Student aStudent;
```

Formally, a Student consists of:

Formally, a particular Student:

```
aStudent.id = 12345;
aStudent.gpa = 3.75;
aStudent.gender = 'F';
```

is the 3-tuple: _____.

The C++ _____ is a projection operation:

```
cout << aStudent.id           // extract id
     << aStudent.gpa          // extract gpa
     << aStudent.gender        // extract gender
     << endl;
```



Set Constructor II: _____

The function constructor is the basis for _____.

- The set of all functions from a set A to a set B is denoted _____.
- A particular function f mapping A to B is denoted _____.

Examples:

- The set $(\text{char}) \rightarrow \text{bool}$ contains all functions that map char values into bool values, some C examples of which include:

- The set $(\text{char}) \rightarrow \text{char}$ contains all functions that map char values into char values, some C examples of which include:



Function and Product

What does this set contain? $(\text{int} \times \text{int}) \rightarrow \text{int}$

– _____.

Examples? _____

Suppose we define an aggregate named *IntPair*:

```
struct IntPair {  
    int a,  
    b;  
};
```

and then define a function named `add()`:

```
int add(IntPair ip) {  
    return ip.a + ip.b;  
};
```

`add()` is a member of the set: _____

– The function constructor let us create _____.



Function Arity

Product serves to denote an aggregate or an argument-list.

What does this set contain? $(\text{int} \times \text{int}) \rightarrow \text{bool}$

- All functions that map _____.

Examples? _____

Definition:

The number of operands an operation requires is its _____.

- Operations with 1 operand are _____ operations, with *arity-1*.
- Operations with 2 operands are _____ operations, with *arity-2*.
- Operations with 3 operand are _____ operations, with *arity-3*.
- ...



Example Ternary Operation

The C/C++ conditional expression has the form:

$\langle \text{expr} \rangle_0 ? \langle \text{expr} \rangle_1 : \langle \text{expr} \rangle_2$

producing $\langle \text{expr} \rangle_1$ if $\langle \text{expr} \rangle_0$ is true, and
producing $\langle \text{expr} \rangle_2$ if $\langle \text{expr} \rangle_0$ is false.

Here is a simple *minimum()* function using it:

```
int minimum(int first, int second) {  
    return (first < second) ? first : second;  
};
```

The C/C++ conditional expression is a ternary operation,
which in this case is a member of the set:



Operator Positioning

Operators are also categorized by their position relative to their operands:

- _____ operators appear *between* their operands: 1 + 2
- _____ operators appear *before* their operands: + 1 2
- _____ operators appear *after* their operands: 1 2 +

_____ ≡ _____ ≡ _____

Prefix, infix, and postfix notation are different conventions for the same thing; a language may choose any of them:

C++ Expr	Category	Value	Lisp Expr	Category	Value
$x < y$	_____	true, false	$(< x y)$	_____	true, false
$++x$	_____	$x+1$	$(incf x)$	_____	$x+1$
$11 + 12$	_____	23	$(+ 11 12)$	_____	23
$!flag$	_____	neg. of <i>flag</i>	$(not\ flag)$	_____	neg. of <i>flag</i>
$cout << x$	_____	cout	$(princ\ x\ str)$	_____	<i>x</i>
$x++$	_____	<i>x</i>	None		



Set Constructor III: _____

Kleene Closure is the basis for representing _____.

- The Kleene Closure of a set A is denoted _____.
- The Kleene Closure of a set is the set of all tuples that can be formed using elements of that set.

Example: The Kleene Closure of `bool` -- `bool*` -- is the infinite set:

{ _____
_____ ... }

- For a tuple $t \in A^*$, the operations include:

$\text{null}(A^*) \rightarrow \text{bool}$	<code>null()</code>	\rightarrow _____
	<code>null(false)</code>	\rightarrow _____
	<code>null(true)</code>	\rightarrow _____

$\text{first}(A^*) \rightarrow A$	<code>first(true, false)</code>	\rightarrow _____
	<code>first(false, true)</code>	\rightarrow _____

$\text{rest}(A^*) \rightarrow A^*$	<code>rest(true, true, false)</code>	\rightarrow _____
	<code>rest(false, true, true)</code>	\rightarrow _____



Kleene Closure Examples

If *char* is the set of ASCII characters, what is *char** ?

- The infinite set of _____.
(AKA the set of all _____).

The C/C++ notation: "Hello"
is just a different syntax for: ('H', 'e', 'l', 'l', 'o')

Thus, *int** denotes a sequence (array, list, ...) of _____;

```
int intStaticArray[32];  
int * intDynamicArray = new int[n],  
vector<int> intVec;  
list<int> intList;
```

*real** denotes a sequence (array, list, ...) of _____;
and so on.



Sequence Operations

Sequence operations can be built via *null()*, *first()*, and *rest()*

- An output operation can be defined like this (pseudocode):

```
void print(ostream out, int * a) {  
    if ( !null(a) ) {  
        out << first(a) << ' ';  
        print(out, rest(a));  
    }  
};
```

- A subscript operation can be defined like this (pseudocode):

```
char & operator[](int * a, int i) {  
    if (i > 0)  
        return operator[](rest(a), i-1);  
    else  
        return first(a);  
};
```

In Lisp:

first is called _____

rest is called _____.



Practice Using Constructors

Give formal descriptions for:

- The *logical and* operation ($\&\&$):

- How many operands does it take? _____

- What types are its operands? _____

- What type of value does it produce? _____

- So $\&\&$ is a member of _____

- The C++/STL *substring* operation ($str.substr(i,n)$):

- How many operands does it take? _____

- What types are its operands? _____

- What type of value does it produce? _____

- So $substr()$ is a member of: _____

- For you: The *logical negation* operation (!):



More Practice

- For you: this *C++ record*:

```
struct Student {  
    int myID;  
    string myName;  
    bool iAmFullTime;  
    double myGPA;  
};
```

- For you: an *accessor* method:

```
struct Student {  
    int myID;  
    int id() const ;  
    string myName;  
    bool iAmFullTime;  
    double myGPA;  
};
```

- How does this affect our *Student* description?



More Practice (ii)

- For you: A “complete” class:

```
class Student {  
    public:  
        Student();  
        Student(int, string, bool, double);  
        int getId() const;  
        string getName() const;  
        bool getFullTime() const;  
        double getGPA() const;  
        void read(istream &);  
        void print(ostream &) const;  
    private:  
        int    myID;  
        string myName;  
        bool    iAmFullTime;  
        double myGPA;  
};
```



Summary

A type consists of _____ and _____.

The set constructors:

- _____,
- _____, and
- _____

provide a formal way to represent types:

- Use the *product* and *Kleene closure* to represent the _____;
- Use the *function* constructor to represent the _____ on the type.

