COL100 Lecture 12

Note Title 06-09-2018

Review:

- pass/call-by-value
- call-by-reference : cannot pass non-variables
- const parameters : can pass non-variables
  behaves as call-by-value in this case
- output parameters
- good decomposition
- Strings
  - C++ string. eg. `string s;`
  - C string. eg. `char s[10];`

long double
long long int

long int

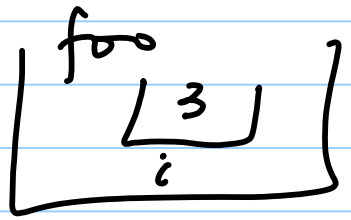
float

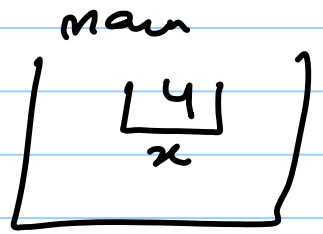
```
void foo (const int & i)
{
  ...
}

int main ()
{
  int x = 3;
  foo (x+1);
}
```

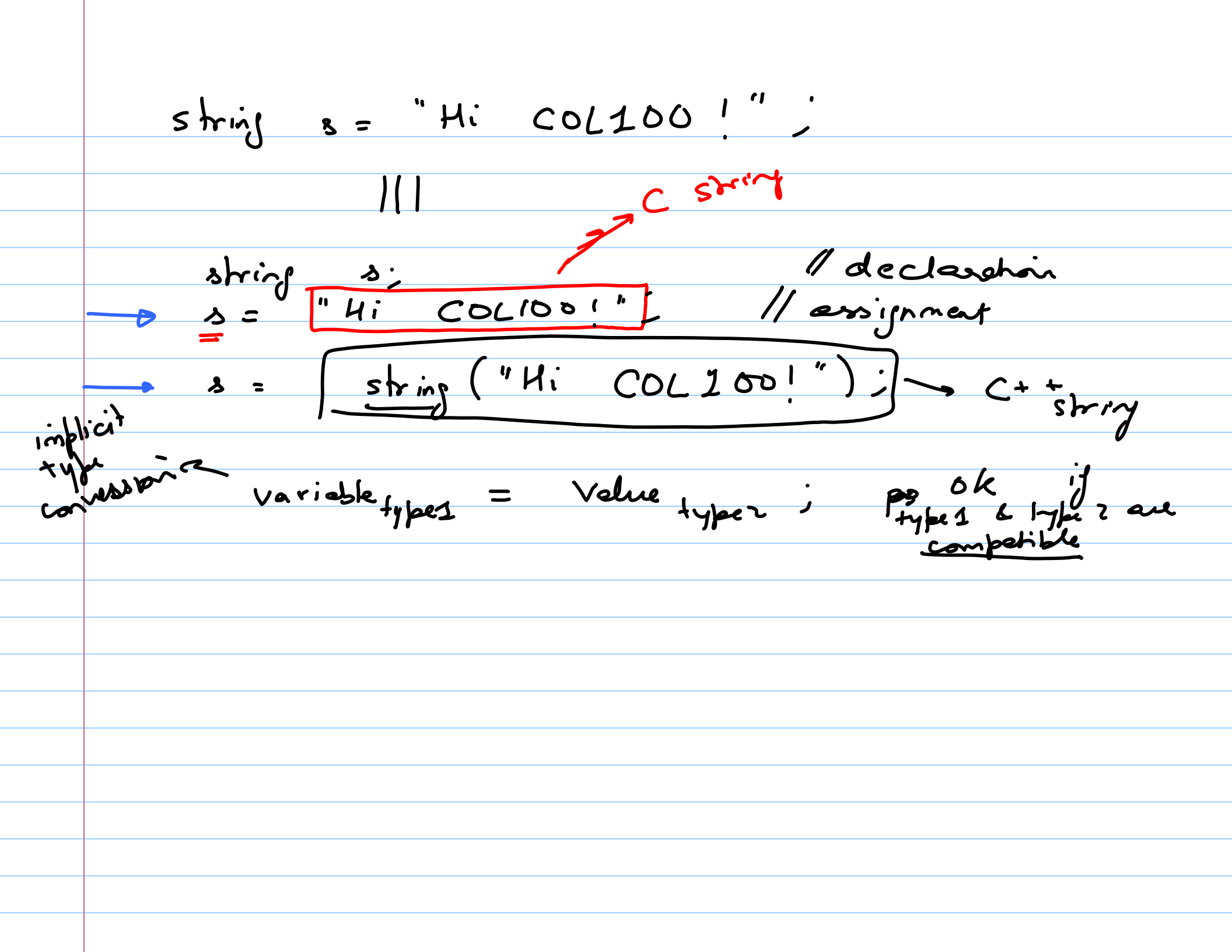
x++

value





```
string s = "Hi COL100!";
```



variable$_{type1}$ = value$_{type2}$; ok if $type1$ & $type2$ are compatible

index	0	1	2	3	4	5	6	7	8	9
value	H	i	_	C	O	L	1	0	0	!

```
char c1 = s[3];
cout << c1; // 'C'
```

'C'

```
char c2 = s.at(7);
cout << c2; // 0
```

'0'

Characters have ASCII encodings

eg:

Character representation		ASCII representation
'0'	⟶	21
'1'	⟶	22

```
int c = int('0');
```

ASCII value of this character

type-casting

```
int i = (int) '0';
```

type-casting

↓

change the type of the value to TYPE

(type) value

value

(int) '0' .

Strings

#include <string>

string s = " .... ";

s.at(3)

↳ member function of string type

"at" is a **member function**

s = "Hi" ; str = "COLIOO"

s = "HiCOLIOO".

	s.append(str)	add text to the end of a string.
int	s.compare(str)	returns <0, 0, or >0 depending on relative ordering
	s.erase(index, length)	deletes text from a string starting at index.
	s.find(str) s.rfind(str)	returns the first/last index where the start of "str" appears in the string s; string::npos

s1 = "SZ" ;
s2 = "hi i . . . . " ;

s1. compare (s2) :

```
string name = "Thomas Edison";
```

0 1 2 3 4 5 6 7

```
name.erase(7, 6);
cout << name; // Thomas␣
```

find

string::npos

```
       0 1 2 3 4 5 6 7
string name = "Thomas Edison";

int p = name.find("Edi");
int q = name.rfind("Edi");
cout << p;   // 7
string h = "hi";
name.find(h);

name.find("Th");

name.rfind("o");
```

```
if (s.find(str) != string::npos)
{
    // do something

}
```

- erase

```
string s = "cat";
s.erase(0,3);
cout << s;        " "
```

```
string s;
cout << s;          // ""

string s = "Thomas Edison";
cout << s.find("o");   // 2
```

string s = "hi";

string s = "\73\74"

"hi"

s.insert(index, str);	add text into s at the given index
s.length() or s.size()	returns the number of characters in s
~~s.replace(index~~	

```
string s = "Hello";
s.insert(3, "FOO");
cout << s;  // HelFOOlo
```

Number of digits in input

```
int i = getInteger("prompt"); //assume positive
string s = integerToString(i);
    cout << s.length();
    cout << s; // get back the input
```