Types

CSE2013-17F

New Types

- Basic types in C
  - `char, short, int, long, float, double`
- New types in C++ standard
  - `wchar_t`: wide character (e.g., 8, 16, 32 bits)
  - `bool`: true / false
  - `long long`: 64-bit integer (“조”, “경”)
  - `long double`: more precise than `double`

New Variable Initialization

- 2 new ways of variable initialization

```
#include <iostream>
using namespace std;

int main() {
    int a = 5;          // c-like init
    int b(3);           // constructor init
    int c{2};           // uniform init

    cout << a << b << c << endl;
}
```

Auto **Keyword**

- The C++ compiler automatically decides types of variables

```
int a = 0;
auto b = a;          // int b
auto c = 1.0;        // double c


decltype(a) d;       // the same type as a
```

`Constant` **Keyword**

- A variable declared with `Constant` is ‘read-only’

```
const double pi = 3.14159265359;
const char tab = '\t';

double r = 10.0;     // radius

double area = pi * r * r;
double circle = 2 * pi * r;

cout << area << tab << circle << endl;
```

std::string

- std::string
  - #include <string>
- Initialization
  - std:string str;
- getline(cin, str)
  - cin a line of input >> variable 'str'

```
#include <string>
…
std::string str1("C++ string");
std::string str2 = "another way of init";

char str3[] = "c-style string";
```

```
int main() {
    string s;
    getline(cin,s);
    cout << "s = " << s << endl;
}
```

std::string ii

- Copy (=), concatenation (+), length (str.length())

```
#include <iostream>
#include <string>
using namespace std;

int main() {
    string s1("one");
    string s2("two");
    string s3;

    s3 = s1;          // string copy
    s3 = s1 + s2;     // string concatenation
    cout << "s1 + s2 = " << s3 << endl;
    cout << "s3 length = " << s3.length() << endl;
    cout << s3[0] << " " << s3[3] << endl;
}
```

std::string iii

- string into stream

```
#include <iostream>
#include <string>
#include <sstream>
using namespace std;

int main() {
    string str = "one two three";
    stringstream ss(str);

    string tok;
    while (ss >> tok)
        cout << tok << endl;
}
```

std::vector i

- std::vector - dynamic array
  - #include <vector>
  - a[idx] , a.at(idx)
  - a.size( )

```
#include <vector>
…
std::vector<int> a1 = { 0, 1, 3 };
std::vector<double> f1 = { 0.1, 3.14 };

a1[2] = 2;              // element access as if array
f1.at(0) = 1.44;        // or use at()

cout << "size = " << a1.size() << endl;
```

std::vector ii

- int arr[10];
- vector<int> arr;
- arr[0] = 1; // error: out of bound
- vector<int> arr1(10, 0); // size 10 and initialized with zero
- arr1[0] = 1; arr2[1] = 100; …
- vector<int> arr2 = arr1; // arr2 has 10 same elements

`std::vector` - **stack**

- push_back(), pop_back()
  - push/pop an element on/off the stack
- back()
  - return value at the top of the stack

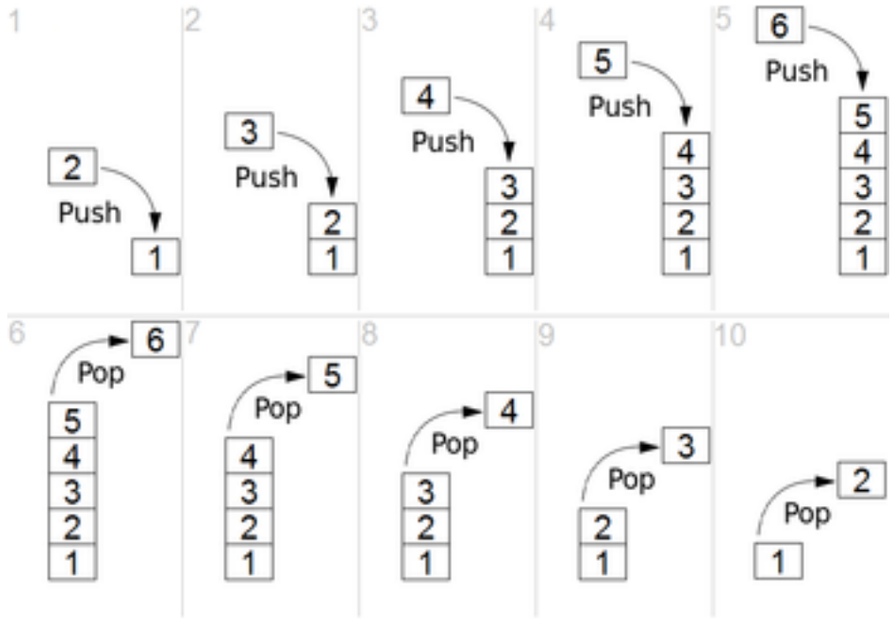

image from wikipedia

[Lab - Practice #1]

- Split a string line into words and print them in reverse order (*must use std::vector)
    - Input: string (e.g. “Test my skill”)
    - Output:
        - Original input string
        - One word per line in reverse order

```
$ ./rssplit
input: Test my skill
skill
my
Test
```