For Each

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

int main() {
    const int N = 3;
    int scores[N] = {90, 95, 100};

    for (auto n : scores)
        cout << n << endl;
}
```

for-each-element in array:

```cpp
for (element declaration : array)
    statement;
```
#include <iostream>
using namespace std;

int main() {
    const int N = 3;
    int scores[N] = {90, 95, 100};

    for (auto &n : scores)
        cout << n << endl;
}

Use reference (pointer) instead of copying element (which may be expensive)
#include <iostream>
using namespace std;

void swap (int *a, int *b) {
    int t = *a;
    *a = *b;
    *b = t;
}

int main() {
    int x = 3, y = 5;
    swap(&x, &y);
    cout << x << ' ' << y << endl;
}
#include <iostream>
using namespace std;

void swap (int &a, int &b) {
    auto t = a;
    a = b;
    b = t;
}

int main() {
    int x = 3, y = 5;
    swap(x, y);
    cout << x << ' ' << y << endl;
}
Random Numbers

- Pseudo random number generator

```cpp
#include <iostream>
#include <cstdlib>  // c standard lib

using namespace std;

int main() {
    srand(2312); // set initial seed value
    for (int i = 0; i < 10; i++)
        cout << rand() << endl;
}
```

Pseudo random:
Sequence of numbers from rand() is the same, if seed value is the same
Random Numbers (2)

- Set seed from `time()`

```cpp
#include <iostream>
#include <cstdlib>  // c standard lib
#include <ctime>    // c time lib
using namespace std;

int main() {
    srand((time(NULL)));
    for (int i = 0; i < 10; i++)
        cout << rand() << endl;
}
```

time(NULL) returns the number of seconds since 00:00 hours, Jan 1, 1970 UTC
[Lab – Practice #1]

- int randRange(min, max)
  - Command line: number of random numbers
  - Input: min, max (check the values if min <= max)
  - Output: random numbers in the range min to max inclusive (min <= output <= max)

```
$ ./randRange 5
min? 10
max? 100
6
97
34
21
59
```
[Project #1] Poker

- Implement Seven Poker Game
  - 4 players ("player 0" is your avatar)
  - Lastly, player who has the largest score is winner

- 5 files
  - Makefile / main.cpp / poker.cpp / poker.h / poker_AI.o
  - You should only change "poker.cpp"
[Project #1] Poker

- **poker.h**
  - You can make new variables between /*****/
  - Formula (Pair ~ Royal Straight Flush)
  - Classes
    - Card : Number / shape
    - Condition : State of formula
    - Player : Name / state of cards(hand, table)
    - Poker : Card deck / functions()
[Project #1] Poker

- **poker.cpp**
  - Shuffle_deck
    - Shuffle cards (52 cards)
    - Random seed is decided
    - Use “push_back()” to deck variable and use from behind
  - First_draw
    - Hand out 3 cards to players (0, 1, 2, 3 order)
  - Verify_top
    - Overturn your top card
    - In verify_top_AI(), AIs overturn their cards
[Project #1] Poker

- **poker.cpp**
  - **Verify_table**
    - Check all cards in table and update players’ state
  - **Verify_mine**
    - Check cards in my hand and update my state
  - **Show_table**
    - Show all cards in tables
    - Should not display dead player
    - Example

```
Player 0 : xx xx AS 8C 4C 8S 5S
Player 1 : xx xx AD JH 7D 5H 4H
Player 2 : xx xx 9D JC 8H QD 4S
Player 3 : xx xx KD JD 10H AC 3D
```
[Project #1] Poker

- **poker.cpp**
  - **Draw**
    - Hand out 1 card to players
    - You should find top player in table
      - Hand out the card counterclockwise
      - If player 2 is top, $2 \rightarrow 3 \rightarrow 0 \rightarrow 1$ order
  - **Go_or_die**
    - You can choose go or die
    - Make your algorithm in here
    - In `go_or_die_AI()`, AIs decide their play
[Project #1] Poker

- poker.cpp
  - Show_result
    - Show result of game
    - Calculate score
    - Example

```
< Result >
Player 0 : xx xx AS 8C 4C 8S 5S
Player 2 : xx xx 9D JC 8H QD 4S
Winner is 'Player 0'!!
```