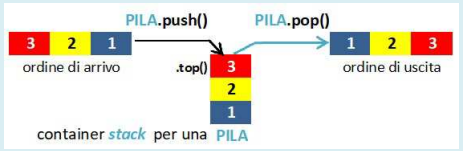
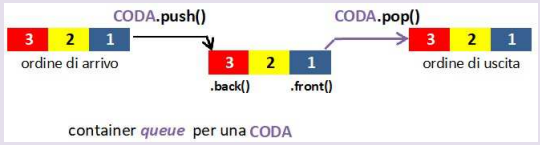
Esempio di gestione mediante *container* di una PILA *(stack)* e di una CODA *(queue)*

prof.ssa P.Grandillo



pila_stack.cpp (esempio: pila di pratiche da elaborare)

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

  stack <int> PILA;

  ifstream finp ("finp_P_C.txt");

void scriviPila (){
    cout << "\n\nin PILA pratiche: ";
    cout << PILA.size();

    cout << " - ultima: " << PILA.top();
}


int Push  (int pra) {
    PILA.push (pra);
    return 0;
}
int Pop  (int& praT) {
    if ( PILA.empty() ) {
      cout <<"\n\n-------------------------------------\n";
      cout <<   "errore Pila vuota, nessuno da servire";
      return -1;
    }
    scriviPila ();
    praT = PILA.top();   //-- lettura dell'ultimo elem. inserito
    PILA.pop ();        //-- rimuove l'ultimo elemento dalla Pila
    return 0; //per stack il metodo .pop() opera su ultimo elemento
}
```



coda_queue.cpp (esempio: fila di persone da servire)

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

  queue <int> CODA;

  ifstream finp ("finp_P_C.txt");

void scriviCoda ()  {
    cout << "\n\nin CODA persone : ";
    cout << CODA.size();
    cout << " - prima: "   << CODA.front();
    cout << " - ultima: " << CODA.back();
}


int Push  (int per) {
    CODA.push (per);
    return 0;
}
int Pop  (int& perT) {
    if ( CODA.empty() ) {
      cout <<"\n\n-------------------------------------\n";
      cout <<   "errore Coda vuota, nessuno da servire";
      return -1;
    }
    scriviCoda ();
    perT = CODA.front();  //-- lettura del primo elem. inserito
    CODA.pop();         //-- rimuove il primo elemento dalla Coda
    return 0; //per queue il metodo .pop() opera su primo elemento
}
```

Esempio di gestione mediante *container* di una PILA *(stack)* e di una CODA *(queue)*

prof.ssa P.Grandillo

```
int main(){

  int PRA, praPila;          //-- PRA pratica letta da file
  int N, esito=0;           //-- praPila prelevata da PILA

    finp >> N;

    for (int i=0; i<N && esito==0; i++) {

        finp >> PRA;

        if (PRA > 0)           //--- arriva  pratica
            esito = Push(PRA);
        else                   //---elaborata pratica
        {   esito = Pop (praPila);
            if (esito == 0)
                cout << "\nconclusa pratica: "<< praPila;
        }
    }

    if (esito == 0)

      if ( ! PILA.empty() ) {    //-- condizioni equivalenti

        cout << "\n\n-------------------------\n";
        cout <<     "errore: Pila ancora piena";
        scriviPila ();
      }
      else
        cout <<"\n\n OK - non ci sono altre pratiche nella Pila";
    cout << endl;
    finp.close();
    return 0;
}
```

```
int main(){

  int PER, perCoda;          //-- PER persona letta da file
  int N, esito=0;           //-- perCoda prelevata da CODA

    finp >> N;

    for (int i=0; i<N && esito==0; i++) {

        finp >> PER;

        if (PER > 0)           //---arriva persona
            esito = Push(PER);
        else                   //--- esce persona
        {   esito = Pop (perCoda);
            if (esito == 0)
                cout << "\nservita persona : "<< perCoda;
        }
    }

    if (esito == 0)

      if ( CODA.size() > 0 ) {    //-- condizioni equivalenti

        cout << "\n\n-------------------------\n";
        cout <<     "errore: CODA ancora piena";
        scriviCoda ();
      }
      else
        cout <<"\n\n OK - non ci sono altre persone nella Coda";
    cout << endl;
    finp.close();
    return 0;
}
```

*NOTA: sia per il container **stack** che per il container **queue**, le condizioni* **! nome.empty()** *e* **nome.size() > 0** *sono equivalenti*

per approfondimenti consultare :

https://www.cplusplus.com/reference/stack/stack/

https://www.cplusplus.com/reference/queue/queue/

Esempio di gestione mediante *container* di una PILA *(stack)* e di una CODA *(queue)*

p.3/3 **casi di Test** prof.ssa P.Grandillo

Entrambi i programmi leggono dal File di Testo il numero N di ingressi ed uscite dalla **PILA**/**CODA**; poi leggono gli N valori successivi: un numero positivo indica il progressivo della **pratica**/**persona** in arrivo (1 2 ...) che va inserita nel ***container stack*/*queue*** mediante il metodo ***.push()***; il valore **-1** indica che una **pratica**/**persona** è stata **elaborata**/**servita** e che, mediante il metodo ***.pop()***, dalla **PILA** può essere eliminato **l'ultimo elemento dalla fine** mentre la **CODA** può scorrere **eliminando il primo elemento dall'inizio**.

OUTPUT con *pila_stack.exe*	**OUTPUT con *coda_queue.exe***
caso 1 – file di input corretto →	**8** **1** **2** **-1** **3** **4** **-1** **-1** **-1**
Prompt dei comandi >> >>pila_stack.exe in PILA pratiche: 2 - ultima: 2 conclusa pratica: 2 in PILA pratiche: 3 - ultima: 4 conclusa pratica: 4 in PILA pratiche: 2 - ultima: 3 conclusa pratica: 3 in PILA pratiche: 1 - ultima: 1 conclusa pratica: 1 OK - non ci sono altre pratiche nella Pila >>	Prompt dei comandi >>coda_queue.exe in CODA persone : 2 - prima: 1 - ultima: 2 servita persona : 1 in CODA persone : 3 - prima: 2 - ultima: 4 servita persona : 2 in CODA persone : 2 - prima: 3 - ultima: 4 servita persona : 3 in CODA persone : 1 - prima: 4 - ultima: 4 servita persona : 4 OK - non ci sono altre persone nella Coda >>
caso 2 – input errato (uscite inferiori a ingressi) →	**6** **1** **2** **3** **4** **-1** **-1**
Prompt dei comandi >>pila_stack.exe in PILA pratiche: 4 - ultima: 4 conclusa pratica: 4 in PILA pratiche: 3 - ultima: 3 conclusa pratica: 3 ---------- errore: Pila ancora piena in PILA pratiche: 2 - ultima: 2 >>	Prompt dei comandi >>coda_queue.exe in CODA persone : 4 - prima: 1 - ultima: 4 servita persona : 1 in CODA persone : 3 - prima: 2 - ultima: 4 servita persona : 2 ---------- errore: CODA ancora piena in CODA persone : 2 - prima: 3 - ultima: 4 >>
caso 3 – input errato (uscita precede ingresso) →	**6** **1** **2** **-1** **-1** **-1** **3**
Prompt dei comandi >>pila_stack.exe in PILA pratiche: 2 - ultima: 2 conclusa pratica: 2 in PILA pratiche: 1 - ultima: 1 conclusa pratica: 1 ---------- errore Pila vuota, nessuno da servire >>	Prompt dei comandi >>coda_queue.exe in CODA persone : 2 - prima: 1 - ultima: 2 servita persona : 1 in CODA persone : 1 - prima: 2 - ultima: 2 servita persona : 2 ---------- errore Coda vuota, nessuno da servire >>