Una `SegreteriaStudenti` contiene 3 sportelli. Gli sportelli possono essere liberi o occupati. Quando uno studente arriva nella segreteria studenti, se tutti gli sportelli sono occupati, si mette in coda; altrimenti, occupa uno degli sportelli liberi. Ogni studente è identificato nella coda di attesa da un numero intero progressivo. La coda può contenere un qualsivoglia numero di studenti.

Realizzare le operazioni seguenti che possono essere effettuate su di una `SegreteriaStudenti`:

- ✓ **`SegreteriaStudenti s();`**

Costruttore che inizializza una segreteria studenti `s`. Inizialmente, la segreteria ha tutti gli sportelli liberi e non ci sono clienti in coda.

- ✓ **`~SegreteriaStudenti();`**

Distruttore.

- ✓ **`s.arrivo();`**

Operazione che implementa l'arrivo di uno studente nella segreteria studenti `s`. Se tutti gli sportelli sono occupati, lo studente si mette in coda; altrimenti, occupa uno degli sportelli liberi.

- ✓ **`s.abbandona(j);`**

Operazione che implementa l'abbandono della coda da parte dello studente `j`. La funzione restituisce `false` se lo studente `j` non è nella coda di attesa; altrimenti restituisce `true`.

- ✓ **`s.libera(k)`**

Operazione che libera lo sportello `k`. Quando si libera uno sportello, lo studente che attende da più tempo esce dalla coda e occupa lo sportello. La funzione restituisce `false` se l'indice dello sportello è errato oppure lo sportello era già libero; altrimenti restituisce `true`.

- ✓ **`cout << s`**

Operatore di uscita per il tipo `SegreteriaStudenti`. L'operatore stampa, per ogni sportello, se lo sportello è libero o occupato, e, se esistono studenti in coda, il numero a loro assegnato.

Per esempio, per una segreteria studenti con i 3 sportelli occupati e con 2 studenti in coda (lo studente con il numero 3 e lo studente con il numero 5) l'operatore produrrà la stampa seguente:

```
Sportelli
1: occupato
2: occupato
3: occupato
Studenti in coda:
3
5
```

Mediante il linguaggio C++, realizzare il tipo di dato astratto `SegreteriaStudenti`, definito dalle precedenti specifiche. Individuare eventuali situazioni di errore, e metterne in opera un corretto trattamento.

Di seguito viene riportato un esempio di main.cpp e la relativa uscita attesa.

```
// file main.cpp
#include "compito.h"

int main(){

    {
        //test costruttore e distruttore
        SegreteriaStudenti s;
    }

    SegreteriaStudenti s;

    //test arrivo studenti: 3 studenti vengono serviti, 5 studenti vanno in coda
    for(int i=0; i<8;i++){
        s.arrivo();
    }

    //test stampa
    cout << s << endl;

    //test abbandona
    s.abbandona(1); //via lo studente in coda con il numero 1
    s.abbandona(3);
    cout << s << endl;


    //test libera
    s.libera(1); // lo sportello numero 1 diventa libero
                 // e serve lo studente in testa alla coda

    s.libera(1); // lo sportello numero 1 diventa libero
                 // e serve lo studente in testa alla coda

    s.libera(1); // lo sportello numero 1 diventa libero
                 // e serve lo studente in testa alla coda

    s.libera(1); // lo sportello numero 1 diventa libero
                 // e serve lo studente in testa alla coda

    //test stampa
    cout << s << endl;

    return 0;
}
```

USCITA ATTESA

```
Sportelli
1: occupato
2: occupato
3: occupato
Studenti in coda:
      1
      2
      3
      4
      5

Sportelli
1: occupato
2: occupato
3: occupato
Studenti in coda:
      2
      4
      5

Sportelli
1: occupato
2: libero
3: occupato
Studenti in coda:
```