```
from dataclasses import dataclass
from typing import List, Dict, Tuple
import numpy as np
import datetime


@dataclass
class Person(object):
    cin: int
    person_type: str
    full_name: str
    email_address: str
    is_happy: bool = True

    def __str__(self):
        return f"{self.person_type}: {self.full_name} ({email_address})"


class ClassRoster(object):
    def __init__(self, instructor: Person):
        self.instructor = instructor
        self.date_created = datetime.datetime.now()
        self.students = []  # List[Person]

    def add_student(self, student: Person):
        """
        add_student(x) adds student x to the roster.

        :param student: the student (Person) object to add
        :raises: ValueError if student is not a Student or other error
        """
        if student.person_type != 'Student':
            raise ValueError(f"{student} is not a Student!")
        self.students.append(student)

    def find_student(self, cin: int = 0, email: str = '') -> int:
        """
        find_student(x) finds student with cin or email x on the roster.

        :param cin: kwarg for removal by CIN
        :param email: kwarg for removal by email address
        :return: index of student in self.students, otherwise -1 if not found
        """
        for idx, student in enumerate(self.students):
            if student.cin == cin or student.email_address == email:
                return idx
        return -1
```

```
    def remove_student(self, cin: int = 0, email: str = '') -> bool:
        """
        remove_student(x) removes student with cin or email x from the roster.

        :param cin: kwarg for removal by CIN
        :param email: kwarg for removal by email address
        :return: True if student removed, False if student not found
        """
        if cin == 0 and email == '':
            return False
        else:
            student_idx = self.find_student(cin, email)
            if student_idx == -1:
                return False
            return bool(self.students.pop(student_idx))

    def __str__(self):
        return ', '.join([str(student) for student in self.students])



def main():
    additional_input = 'hello'
    print('Hello!)


if __name__ == '__main__':
    main()
```