```cpp
// Connor Petri CIS 22C, 11/28/23

siDset * siDset::orderedUnion(const siDset *ptr)
{
    siDset *returnSet = new siDset();

    for (int i = 0, j = 0; i < this->getSize() && j < ptr->getSize();)
    {
        // if the i-th element of [this] is equal to the j-th element of ptr,
        // then append it to the return set and increment both iterator variables.
        if (this->get(i).id == ptr->get(j).id)
        {
            returnSet->append(this->get(i));
            i++;
            j++;
        }
        // else add the smallest value to the return set
        else if (this->get(i).id < ptr->get(j).id)
        {
            returnSet->append(this->get(i));
            i++;
        }
        else
        {
            returnSet->append(ptr->get(j));
            j++;
        }

        // If i or j are out of range, dump the remaining elements from the
        // remaining array and break from loop
        if (i == this->getSize())
        {
            for (; j < ptr->getSize(); j++)
            {
                returnSet->append(ptr->get(j));
            }
            break;
        }

        if (j == ptr->getSize())
        {
            for (; i < this->getSize(); i++)
            {
                returnSet->append(this->get(i));
            }
            break;
        }
    }

    return returnSet;
}

siDset * siDset::unorderedUnion(const siDset *ptr)
{
    // copy constructor called to copy all contents from current object to the return set
    siDset *returnSet = new siDset(this);

    // Copy all elements that are not duplicates from ptr to returnSet
    for (int i = 0; i < ptr->getSize(); i++)
    {
        // Search through each element to confirm ptr->get(i) does not exist within [this]
        bool isCopy = false;
        for (int j = 0; j < this->getSize(); j++)
        {
            if (ptr->get(i).id == this->get(j).id)
```

```
66              {
67                  isCopy = true;
68                  break;
69              }
70          }
71
72          if (!isCopy) // append if not a copy
73          {
74              returnSet->append(ptr->get(i));
75          }
76      }
77
78      return returnSet;
79  }
```