به نام خدا

کلاسی به نام unique pairs براي ذخيره سازي مجموعه اي نامحدود از داده ها طراحی کنید که داده ها در آن به صورت زوج هاي مرتب(کلید، مقدار) هستند. در واقع در این مجموعه، کلید نقش اندیس را بازي می کند و می توان بر اساس کلید به عنصر مقدار دسترسی پیدا کرد. قوانینی براي این مجموعه وجود دارد:

- کلید منحصر به فرد است، یعنی امکان ندارد دو زوج با کلید یکسان در مجموعه وجود داشته باشد.
- نوع کلید و نوع مقدار براي همه عناصر یک مجموعه واحد است. یعنی مثلا یک مجموعه داریم که کلید همه عناصر از نوع int و مقدار همه عناصر از نوع double است. اما می توان مجموعه ي ديگري با نوع کلید float و مقدار string هم داشت.

-نيازي به مرتب کردن داده ها بر حسب کلید نیست.

این کلاس باید حاوي امکانات زیر باشد:

- Insert: یک زوج داده اي را دریافت کرده و در صورت یکتا بودن کلید، آن را به مجموعه اضافه کند. در صورت تکراري بودن کلید، زوج جدید جایگزین زوج قبلی می شود .
- Remove: یک کلید را دریافت کرده و در صورت وجود در مجموعه، زوج داده اي آن را حذف می کند .
- تابع size: تعداد زوج های موجود در مجموعه را نشان می دهد.
- اپراتور چاپ: همه داده های مجموعه به صورت زوج (کلید، مقدار) یکی یکی چاپ شوند.
- اپراتور []: امکان دسترسی به فیلد مقدار با کمک کلید. در صورتی که آن کلید وجود نداشته باشد ،یک زوج جدید با آن کلید و مقدار پیش فرض ایجاد می شود.
- اپراتور - : تفاضل دو مجموعه را محاسبه می کند. یعنی A-B برابر مجموعه A بدون زوج های (کلید، مقدار) مشترک با مجموعه B می شود.

کلاس دیگری به نام pairs طراحی کنید که مانند کلاس قبل شامل زوج های )کلید، مقدار( باشد، اما در این کلاس شرط یکتا بودن کلید وجود ندارد. امکانات این کلاس به شرح زیر است:

- Insert: یک زوج داده ای را دریافت کرده و آن را به مجموعه اضافه کند. در صورت تکراری بودن زوج (کلید ،مقدار) زوج جدید اضافه نمی شود.
- Remove: یک کلید را دریافت کرده و در صورت وجود در مجموعه، همه زوج داده های آن را حذف می کند.

- تابع size: تعداد زوج های موجود در مجموعه را نشان می دهد.
- اپراتور چاپ: مجموعه داده های مجموعه به صورت زوج (کلید، مقدار) یکی یکی چاپ شوند.
- اپراتور - : تفاضل دو مجموعه را محاسبه می کند. یعنی A-B برابر مجموعه A بدون زوج های (کلید،مقدار) مشترک با مجموعه B می شود.

کلاس های خود را به گونه ای طراحی کنید که با استفاده از آن ها بتوان کد زیر را اجرا کرد. در طراحی کلاس ها از امکانات شی گرایی به گونه ای استفاده کنید که کد شما حداقل باشد .

```
int main()
{
        uniquePairs<char, int> first;

       first['x'] = 8;   //first= {('x',8)}
       first['y'] = 16;  //first= {('x',8), ('y',16)}
        first['z'] = 32;     //first= {('x',8), ('y',16), ('z',32)}
       first.insert('a', 100); //first= {('x',8), ('y',16), ('z',32), ('a',100)}
       first.insert('x', 200);  //first= {('x',200), ('y',16), ('z',32), ('a',100)}
       first.remove('x'); //first= {('y',16), ('z',32), ('a',100)}
       first.remove('a'); //first= {('y',16), ('z',32)}

        uniquePairs<char, int> second;
        second = first;                // second={('y',16), ('z',32)}

        cout << second;     //console output: {('y',16), ('z',32)}
      std::cout << "Size of first: " << first.size() << '\n';  //output: 2
      std::cout << "Size of second: " << second.size() << '\n'; //output: 2

       second.insert('b', 40);
       second.insert('x', 70);
       second.insert('a', 100); // second={('y',16), ('z',32), ('b',40), ('x',70), ('a',100)}
       second = second - first; //second=( ('b',40), ('x',70), ('a',100)}

       first.insert('x', 8);
       const uniquePairs<char, int> third = first;

       second['a']=third['y'];
       cout << third;

       pairs<string, string> phoneBook1;
       phoneBook1.insert("ali", "123");
       phoneBook1.insert("ali", "456");
       phoneBook1.insert("mina", "5678");
       pairs<string, string> phoneBook2;
       phoneBook2["ahmad"] = "1234567";
       phoneBook2["mina"] = "1234567";
      phoneBook2["mina"] = "567";
```

```
    const pairs<string, string>phoneBook3 = phoneBook2;

    phoneBook1.remove("ali"); // phoneBook1 = { ("mina", "567") }

    cout << phoneBook3;

}
```