

C++期末考试 2015/06/26 (8:10-9:40)

事件	时间	学号
提交开始	2016-06-08 0:0	姓名
提交结束	2017-06-01 10:0	班级

1-2 of 2

保存

第1题, class Song

Please write the following functions in class *Song*.

```
• new Song::Song(string name, Artist & artist, Album & album)
```

This problem requires around 5 lines of code.

EXAMPLE INPUT

无

EXAMPLE OUTPUT

```
-- testSong --
song#1: UNKNOWN ARTIST, UNKNOWN ALBUM
song#2: UNKNOWN ARTIST, UNKNOWN ALBUM
```

注意：请注意程序风格（共5点），将检查并扣分。

主程序

```
1 #include <iostream>
2 #include <string>
3 #include <vector>
4 using namespace std;
5
6 class Artist // 音乐艺术家(歌唱者)
7 {
8 public:
9     virtual string getName() {
10         return "UNKNOWN ARTIST";
11     }
12 };
13
14 class Album // 音乐专辑
15 {
16 public:
17     virtual string getName() {
18         return "UNKNOWN ALBUM";
19     }
20 };
21
22 ///////////////////////////////////////////////////
23 // Song
24 ///////////////////////////////////////////////////
25
26 class Song
27 {
28 public:
29     string name;
30     Artist * artist;
31     Album * album;
32
33 public:
34     Song(string name, Artist & artist, Album & album); // TODO
35
36     void print() {
37         cout << name << ": " << artist->getName() << ", " << album->getName() << endl;
38     }
39 };
40
41
42 #include "source"
43
44 ///////////////////////////////////////////////////
45 // Test
46 ///////////////////////////////////////////////////
47
```

```

48
49 void testSong() {
50     cout << "-- testSong --" << endl;
51     Artist artist;
52     Album album;
53     Song song1("song#1", artist, album);
54     Song song2("song#2", artist, album);
55     song1.print();
56     song2.print();
57 }
58
59 int main() {
60     testSong();
61 }

```

答案程序

```

1 Song::Song(string name, Artist & artist, Album & album) {
2     this->name = name;
3     this->artist = &artist;
4     this->album = &album;
5 }

```

文本编辑框

保存和测试对错 答案正确

## 第2题, class Song

注意: 本题需在 第1题 的基础上完成。

Please write the following functions in class *Song*.

- Song::Song(string name, Artist & artist, Album & album)
- new string Song::getName()
- new Artist & Song::getArtist()
- new Album & Song::getAlbum()

Note that you might want to copy some previous code to solve this problem. By using previous code, this problem requires around 15 lines of new code.

### EXAMPLE INPUT

无

### EXAMPLE OUTPUT

```

-- testSong --
song#1: UNKNOWN ARTIST, UNKNOWN ALBUM
song#2: UNKNOWN ARTIST, UNKNOWN ALBUM

```

注意: 请注意程序风格 (共5点), 将检查并扣分。

主程序

```

1 #include <iostream>
2 #include <string>
3 #include <vector>
4 using namespace std;
5
6 class Artist
7 {
8 public:
9     virtual string getName() {
10         return "UNKNOWN ARTIST";
11     }
12 };
13
14 class Album
15 {
16 public:
17     virtual string getName() {
18         return "UNKNOWN ALBUM";
19     }
20 };
21
22 //////////////////////////////////////////////////
23 // Song
24 //////////////////////////////////////////////////
25
26 class Song
27 {
28 public:

```

```

29     string name;
30     Artist * artist;
31     Album * album;
32
33 public:
34     Song(string name, Artist & artist, Album & album);
35
36     string getName(); // TODO
37
38     Artist & getArtist(); // TODO
39
40     Album & getAlbum(); // TODO
41
42 };
43
44 #include "source"
45
46 void print(Song & song) {
47     cout << song.getName() << ": " << song.getArtist().getName() << ", " << song.getAlbum().getName() << endl;
48 }
49
50 ////////////////
51 // Test
52 ////////////////
53
54 void testSong() {
55     cout << "-- testSong --" << endl;
56     Artist artist;
57     Album album;
58     Song song1("song#1", artist, album);
59     Song song2("song#2", artist, album);
60     print(song1);
61     print(song2);
62 }
63
64 int main() {
65     testSong();
66 }

```

答案程序

```

1 Song::Song(string name, Artist & artist, Album & album) {
2     this->name = name;
3     this->artist = &artist;
4     this->album = &album;
5 }
6
7 string Song::getName() { // TODO
8     return this->name;
9 }
10
11 Artist & Song::getArtist() { // TODO
12     return *(this->artist);
13 }
14
15 Album & Song::getAlbum() { // TODO
16     return *(this->album);
17 }

```

文本编辑框

保存和测试对错 答案正确

### 第3题, class Song

注意: 本题需在 第1-2题 的基础上完成。

Please write the following functions for class Song.

- Song::Song(string name, Artist & artist, Album & album)
- string Song::getName()
- Artist & Song::getArtist()
- Album & Song::getAlbum()
- new ostream & operator << (ostream & out, Song & song)

Note that you might want to copy some previous code to solve this problem. By using previous code, this problem requires around 5 lines of new code.

#### EXAMPLE INPUT

无

EXAMPLE OUTPUT

```
-- testSong --
song#1: UNKNOWN ARTIST, UNKNOWN ALBUM
song#2: UNKNOWN ARTIST, UNKNOWN ALBUM
```

注意：请注意程序风格（共5点），将检查并扣分。

主程序

```
1 #include <iostream>
2 #include <string>
3 #include <vector>
4 using namespace std;
5
6 class Artist
7 {
8 public:
9     virtual string getName() {
10         return "UNKNOWN ARTIST";
11     }
12 };
13
14 class Album
15 {
16 public:
17     virtual string getName() {
18         return "UNKNOWN ALBUM";
19     }
20 };
21
22 ///////////////////////////////////////////////////
23 // Song
24 ///////////////////////////////////////////////////
25
26 class Song
27 {
28 public:
29     string name;
30     Artist * artist;
31     Album * album;
32
33 public:
34     Song(string name, Artist & artist, Album & album);
35
36     string getName();
37
38     Artist & getArtist();
39
40     Album & getAlbum();
41
42 };
43
44 #include "source"
45
46 ///////////////////////////////////////////////////
47 // Test
48 ///////////////////////////////////////////////////
49
50 void testSong() {
51     cout << "-- testSong --" << endl;
52     Artist artist;
53     Album album;
54     Song song1("song#1", artist, album);
55     Song song2("song#2", artist, album);
56     cout << song1 << endl; // TODO
57     cout << song2 << endl;
58 }
59
60 int main() {
61     testSong();
62 }
```

答案程序

```
1 Song::Song(string name, Artist & artist, Album & album) {
2     this->name = name;
3     this->artist = &artist;
4     this->album = &album;
5 }
6
7 string Song::getName() { // TODO
8     return this->name;
```

```

9 }
10
11 Artist & Song::getArtist() { // TODO
12     return *(this->artist);
13 }
14
15 Album & Song::getAlbum() { // TODO
16     return *(this->album);
17 }
18
19 ostream & operator << (ostream & out, Song & song) {
20     cout << song.getName() << ": " << song.getArtist().getName() << ", " << song.getAlbum().getName() << endl;
21     return out;
22 }

```

文本编辑框

保存和测试对错

## 第4题, class Artist (音乐艺术家)

Please write the following functions for class ArtistImpl.

- **new** void ArtistImpl::addSong(Song & song)

This problem requires around 15 lines of code.

### EXAMPLE INPUT

无

### EXAMPLE OUTPUT

```

-- testArtist --
artist#1: song#1
artist#1: song#1, song#2
caught: SongExistedException

```

注意: 请注意程序风格 (共5点), 将检查并扣分。

主程序

```

1 #include <iostream>
2 #include <string>
3 #include <vector>
4 using namespace std;
5
6 class Artist
7 {
8 public:
9     virtual string getName() {
10         return "UNKNOWN ARTIST";
11     }
12 };
13
14 class Album
15 {
16 public:
17     virtual string getName() {
18         return "UNKNOWN ALBUM";
19     }
20 };
21
22 class Song
23 {
24 public:
25     string name;
26
27 public:
28     Song(string name) {
29         this->name = name;
30     }
31
32     string getName() {
33         return this->name;
34     }
35 };
36 };
37
38 ///////////////////////////////////////////////////
39 // Artist
40 ///////////////////////////////////////////////////
41

```

```

42 class SongExistedException {};
43
44 class ArtistImpl : public Artist
45 {
46 private:
47     string name;
48     vector<Song> songs;
49
50 public:
51     ArtistImpl(string name) {
52         this->name = name;
53     }
54
55     virtual string getName() {
56         return this->name;
57     }
58
59 public:
60
61     void print() {
62         cout << this->getName() << ": ";
63         for (int i = 0; i < this->songs.size(); ++ i) {
64             cout << (i == 0 ? "" : ", ") << this->songs[i].getName();
65         }
66         cout << endl;
67     }
68
69     void addSong(Song & song); // TODO, may throw SongExistedException
70 };
71
72 #include "source"
73
74 ///////////////////////////////////////////////////
75 // Test
76 ///////////////////////////////////////////////////
77
78 void testArtist() {
79     cout << "-- testArtist --" << endl;
80     ArtistImpl artist1("artist#1");
81     Song song1("song#1");
82     Song song2("song#2");
83
84     try {
85         artist1.addSong(song1);
86         artist1.print();
87         artist1.addSong(song2);
88         artist1.print();
89         artist1.addSong(song2);
90         artist1.print();
91     } catch (SongExistedException & ex) {
92         cout << "caught: SongExistedException" << endl;
93     }
94
95 }
96
97 int main() {
98     testArtist();
99 }

```

答案程序

```

1 void ArtistImpl::addSong(Song & song) {
2     // if song is in songs, throw an exception
3     for (int i = 0; i < songs.size(); ++ i) {
4         if (songs[i].getName() == song.getName()) {
5             throw SongExistedException(); // return
6         }
7     }
8     // else add song into songs
9     songs.push_back(song);
10 }

```

文本编辑框

保存和测试对错 答案正确

## 第5题, class Artist (音乐艺术家)

注意: 本题需在 第4题 的基础上完成。

Please write the following functions for class ArtistImpl.

- void ArtistImpl::addSong(Song & song)
- new Song ArtistImpl::removeSong(Song & song)

Note that you might want to copy some previous code to solve this problem. By using previous code, this problem requires around 15 lines of new code.

#### EXAMPLE INPUT

无

#### EXAMPLE OUTPUT

```
-- testArtist --
artist#1: song#1, song#2
artist#1: song#1
caught: SongNotFoundException
```

注意：请注意程序风格（共5点），将检查并扣分。

主程序

```
1 #include <iostream>
2 #include <string>
3 #include <vector>
4 using namespace std;
5
6 class Artist
7 {
8 public:
9     virtual string getName() {
10         return "UNKNOWN ARTIST";
11     }
12 };
13
14 class Album
15 {
16 public:
17     virtual string getName() {
18         return "UNKNOWN ALBUM";
19     }
20 };
21
22 class Song
23 {
24 public:
25     string name;
26
27 public:
28     Song(string name) {
29         this->name = name;
30     }
31
32     string getName() {
33         return this->name;
34     }
35
36 };
37
38 ///////////////////////////////////////////////////
39 // Artist
40 ///////////////////////////////////////////////////
41
42 class SongExistedException {};
43
44 class SongNotFoundException {};
45
46 class ArtistImpl : public Artist
47 {
48 private:
49     string name;
50     vector<Song> songs;
51
52 public:
53     ArtistImpl(string name) {
54         this->name = name;
55     }
56
57     virtual string getName() {
58         return this->name;
59     }
60
61 public:
62
```

```

63     void print() {
64         cout << getName() << ": ";
65         for (int i = 0; i < songs.size(); ++ i) {
66             cout << (i == 0 ? "" : ", ") << songs[i].getName();
67         }
68         cout << endl;
69     }
70
71     void addSong(Song & song); // may throw SongExistedException
72
73     Song removeSong(Song & song); // TODO, return the removed song, may throw SongNotFoundException
74 };
75
76 #include "source"
77
78 //////////////////////////////////////////////////
79 // Test
80 //////////////////////////////////////////////////
81
82 void testArtist() {
83     cout << "-- testArtist --" << endl;
84     ArtistImpl artist1("artist#1");
85     Song song1("song#1");
86     Song song2("song#2");
87
88     try {
89         artist1.addSong(song1);
90         artist1.addSong(song2);
91         artist1.print();
92         artist1.removeSong(song2);
93         artist1.print();
94         artist1.removeSong(song2);
95         artist1.print();
96     } catch (SongNotFoundException & ex) {
97         cout << "caught: SongNotFoundException" << endl;
98     }
99
100 }
101
102 int main() {
103     testArtist();
104 }

```

答案程序

```

1 void ArtistImpl::addSong(Song & song) {
2     // if song is in songs, throw an exception
3     for (int i = 0; i < songs.size(); ++ i) {
4         if (songs[i].getName() == song.getName()) {
5             throw SongExistedException(); // return
6         }
7     }
8     // else add song into songs
9     songs.push_back(song);
10 }
11
12 Song ArtistImpl::removeSong(Song & song) {
13     // if song not exists, throw exception
14     int index = -1;
15     for (int i = 0; i < songs.size(); ++ i) {
16         if (songs[i].getName() == song.getName()) {
17             index = i;
18             break;
19         }
20     }
21     if (index == -1) throw SongNotFoundException();
22     // else remove song
23     song = songs[index];
24     for (int i = index + 1; i < songs.size(); ++ i) {
25         songs[i - 1] = songs[i];
26     }
27     songs.pop_back();
28     return song;
29 }

```

文本编辑框

保存和测试对错 答案正确

## 第6题, class Artist (音乐艺术家)



注意：本题需在 第4-5题 的基础上完成。

Please write the following functions for class `ArtistImpl`.

- `void ArtistImpl::addSong(Song & song)`
- `Song ArtistImpl::removeSong(Song & song)`
- `new int ArtistImpl::numOfSongs()`
- `new Song & ArtistImpl::getSong(int index)`
- `new ostream & operator << (ostream & out, ArtistImpl & artist)`

Note that you might want to copy some previous code to solve this problem.  
By using previous code, this problem requires around 20 lines of new code.

#### EXAMPLE INPUT

无

#### EXAMPLE OUTPUT

```
-- testArtist --
artist#1: song#1
artist#1: song#1, song#2
artist#1: song#2
caught: SongNotFoundException
```

注意：请注意程序风格（共5点），将检查并扣分。

主程序

```
1 #include <iostream>
2 #include <string>
3 #include <vector>
4 using namespace std;
5
6 class Artist
7 {
8 public:
9     virtual string getName() {
10         return "UNKNOWN ARTIST";
11     }
12 };
13
14 class Album
15 {
16 public:
17     virtual string getName() {
18         return "UNKNOWN ALBUM";
19     }
20 };
21
22 class Song
23 {
24 public:
25     string name;
26
27 public:
28     Song(string name) {
29         this->name = name;
30     }
31
32     string getName() {
33         return this->name;
34     }
35 };
36 };
37
38 ///////////////////////////////////////////////////
39 // Artist
40 ///////////////////////////////////////////////////
41
42 class SongExistedException {};
43
44 class SongNotFoundException {};
45
46 class ArtistImpl : public Artist
47 {
48 private:
49     string name;
50     vector<Song> songs;
51
52 public:
53     ArtistImpl(string name) {
54         this->name = name;
```

```

55     }
56
57     virtual string getName() {
58         return this->name;
59     }
60
61 public:
62
63     void addSong(Song & song); // may throw SongExistedException
64
65     Song removeSong(Song & song); // return the removed song, may throw SongNotFoundException
66
67     Song & getSong(int index); // TODO, may throw SongNotFoundException
68
69     int numOfSongs(); // TODO
70 };
71
72 #include "source"
73
74 ///////////////////////////////////////////////////
75 // Test
76 ///////////////////////////////////////////////////
77
78 void testArtist() {
79     cout << "-- testArtist --" << endl;
80     ArtistImpl artist1("artist#1");
81     Song song1("song#1");
82     Song song2("song#2");
83
84     try {
85         artist1.addSong(song1);
86         cout << artist1 << endl; // TODO
87         artist1.addSong(song2);
88         cout << artist1 << endl;
89         artist1.removeSong(song1);
90         cout << artist1 << endl;
91         artist1.getSong(artist1.numOfSongs());
92     } catch (SongNotFoundException & ex) {
93         cout << "caught: SongNotFoundException" << endl;
94     }
95 }
96
97 int main() {
98     testArtist();
99 }

```

答案程序

1

文本编辑框

保存和测试对错

## 第7题, class Album (音乐专辑)

注意: 本题需在 第1-6题 的基础上完成。

Please write the following functions for class AlbumImpl.

- new AlbumImpl::AlbumImpl(string albumName)

Note that you might want to copy some previous code to solve this problem.  
By using previous code, this problem requires around 5 lines of new code.

### EXAMPLE INPUT

无

### EXAMPLE OUTPUT

```

-- testAlbum --
album#1
Songs (0)

```

注意: 请注意程序风格 (共5点), 将检查并扣分。

主程序

```

1 #include <iostream>
2 #include <string>
3 #include <vector>
4 using namespace std;

```

```

5
6 class Artist
7 {
8 public:
9     virtual string getName() {
10         return "UNKNOWN ARTIST";
11     }
12 };
13
14 class Album
15 {
16 public:
17     virtual string getName() {
18         return "UNKNOWN ALBUM";
19     }
20 };
21
22 //////////////////////////////////////////////////
23 // Song
24 //////////////////////////////////////////////////
25
26 class Song; // TODO: need to include your complete class Song later.
27
28 //////////////////////////////////////////////////
29 // Album
30 //////////////////////////////////////////////////
31
32 class AlbumImpl : public Album
33 {
34 private:
35
36     /*
37     Hint: We will use class ArtistImpl to store songs
38     */
39     Artist * album; // it is used to store all songs & the name of the album
40     vector<Artist *> artists; // it is used store artists
41
42 public:
43
44     virtual string getName();
45
46     int numOfSongs(); // return the number of songs
47
48     ~AlbumImpl() {
49         delete this->album;
50         for (int i = 0; i < artists.size(); ++ i) {
51             delete this->artists[i];
52         }
53     }
54
55     AlbumImpl(string albumName); // TODO
56
57 };
58
59 /*
60 TODO: Please also include
61 - class Song
62 - class ArtistImpl
63 */
64 #include "source"
65
66 string AlbumImpl::getName() {
67     ArtistImpl * album = (ArtistImpl *)this->album;
68     return album->getName();
69 }
70
71 int AlbumImpl::numOfSongs() { // return the number of songs
72     ArtistImpl * album = (ArtistImpl *)this->album;
73     return album->numOfSongs();
74 }
75
76 void print(ostream & out, AlbumImpl & album) {
77     cout << album.getName() << endl;
78     cout << "Songs (" << album.numOfSongs() << ")" << endl;
79 }
80
81 //////////////////////////////////////////////////
82 // Test
83 //////////////////////////////////////////////////
84
85 void testAlbum() {

```

```
86     cout << "-- testAlbum --" << endl;
87     AlbumImpl album1("album#1");
88     print(cout, album1);
89
90 }
91
92 int main() {
93     testAlbum();
94 }
```

答案程序

1

文本编辑框

保存和测试对错

## 第8题, class Album (音乐专辑)

注意: 本题需在 第1-7题 的基础上完成。

Please write the following functions for class AlbumImpl.

- AlbumImpl::AlbumImpl(string albumName)
- new Song & AlbumImpl::operator [] (int songIndex)
- new void AlbumImpl::addSong(string songName, string artistName)

Note that you might want to copy some previous code to solve this problem.  
By using previous code, this problem requires around 30 lines of new code.

### EXAMPLE INPUT

无

### EXAMPLE OUTPUT

```
-- testAlbum --
album#1
Songs (1):
  song#1: artist#1, album#1
Artists (1):
  artist#1

album#1
Songs (2):
  song#1: artist#1, album#1
  song#2: artist#1, album#1
Artists (1):
  artist#1

album#1
Songs (3):
  song#1: artist#1, album#1
  song#2: artist#1, album#1
  song#3: artist#2, album#1
Artists (2):
  artist#1
  artist#2

caught: SongExistedException
```

注意: 请注意程序风格 (共5点), 将检查并扣分。

主程序

```
1 #include <iostream>
2 #include <string>
3 #include <vector>
4 using namespace std;
5
6 class Artist
7 {
8 public:
9     virtual string getName() {
10         return "UNKNOWN ARTIST";
11     }
12 };
13
14 class Album
15 {
16 public:
17     virtual string getName() {
18         return "UNKNOWN ALBUM";
19     }
20 };
```

```
21
22 //////////////////////////////////////////////////
23 // Song
24 //////////////////////////////////////////////////
25
26 class Song;
27
28 //////////////////////////////////////////////////
29 // Album
30 //////////////////////////////////////////////////
31
32 class AlbumImpl : public Album
33 {
34 private:
35
36     Artist * album; // use to store all songs & the name of the album
37     vector<Artist *> artists;
38
39 public:
40
41     virtual string getName();
42
43     int numOfSongs(); // return the number of songs
44
45
46     ~AlbumImpl() {
47         delete this->album;
48         for (int i = 0; i < artists.size(); ++ i) {
49             delete this->artists[i];
50         }
51     }
52
53     AlbumImpl(string albumName);
54
55     Song & operator [] (int songIndex); // TODO, may throw SongNotFoundException
56
57     void addSong(string songName, string artistName); // TODO, may throw SongExistedException
58
59     void print(); // NEW
60 };
61
62 /*
63 Please also include
64 - class Song
65 - class ArtistImpl
66 */
67 #include "source"
68
69
70 string AlbumImpl::getName() {
71     ArtistImpl * album = (ArtistImpl *)this->album;
72     return album->getName();
73 }
74
75 int AlbumImpl::numOfSongs() { // return the number of songs
76     ArtistImpl * album = (ArtistImpl *)this->album;
77     return album->numOfSongs();
78 }
79
80 void AlbumImpl::print() {
81     cout << this->getName() << endl;
82     cout << "Songs (" << this->numOfSongs() << "):" << endl;
83     for (int i = 0; i < this->numOfSongs(); ++ i) {
84         cout << " " << this->operator [] (i) << endl;
85     }
86     cout << "Artists (" << this->artists.size() << "):" << endl;
87     for (int i = 0; i < this->artists.size(); ++ i) {
88         ArtistImpl * artist = (ArtistImpl *) (this->artists[i]);
89         cout << " " << artist->getName() << endl;
90     }
91     cout << endl;
92 }
93
94 //////////////////////////////////////////////////
95 // Test
96 //////////////////////////////////////////////////
97
98 void testAlbum() {
99     cout << "-- testAlbum --" << endl;
100     AlbumImpl album1("album#1");
101 }
```

```

102     try {
103         album1.addSong("song#1", "artist#1");
104         album1.print();
105         album1.addSong("song#2", "artist#1");
106         album1.print();
107         album1.addSong("song#3", "artist#2");
108         album1.print();
109         album1.addSong("song#2", "artist#1");
110         album1.print();
111     } catch (SongExistedException & ex) {
112         cout << "caught: SongExistedException" << endl;
113     }
114
115 }
116
117 int main() {
118     testAlbum();
119 }

```

答案程序

```

1 int main() {
2     testAlbum();
3 }

```

文本编辑框

保存和测试对错

## 第9题, class Album (音乐专辑)

注意: 本题需在 第1-8题 的基础上完成。

Please write the following functions for class AlbumImpl.

- AlbumImpl::AlbumImpl(string albumName)
- Song & AlbumImpl::operator [] (int songIndex)
- void AlbumImpl::addSong(string songName, string artistName)
- new void AlbumImpl::removeSong(string songName)
- new ostream & operator << (ostream & out, AlbumImpl & album)

Note that you might want to copy some previous code to solve this problem. By using previous code, this problem requires around 30 lines of new code.

### EXAMPLE INPUT

无

### EXAMPLE OUTPUT

```

-- testAlbum --
album#1
Songs (3):
  song#1: artist#1, album#1
  song#2: artist#1, album#1
  song#3: artist#2, album#1
Artists (2):
  artist#1: song#1, song#2
  artist#2: song#3

album#1
Songs (2):
  song#2: artist#1, album#1
  song#3: artist#2, album#1
Artists (2):
  artist#1: song#2
  artist#2: song#3

caught: SongNotFoundException

```

注意: 请注意程序风格 (共5点), 将检查并扣分。

主程序

```

1 #include <iostream>
2 #include <string>
3 #include <vector>
4 using namespace std;
5
6 class Artist
7 {
8 public:
9     virtual string getName() {
10         return "UNKNOWN ARTIST";
11     }

```

```

12 };
13
14 class Album
15 {
16 public:
17     virtual string getName() {
18         return "UNKNOWN ALBUM";
19     }
20 };
21
22 //////////////////////////////////////////////////
23 // Song
24 //////////////////////////////////////////////////
25
26 class Song;
27
28 //////////////////////////////////////////////////
29 // Album
30 //////////////////////////////////////////////////
31
32 class AlbumImpl : public Album
33 {
34 private:
35
36     Artist * album; // use to store all songs & the name of the album
37     vector<Artist *> artists;
38
39 public:
40
41     virtual string getName();
42
43     int numOfSongs(); // return the number of songs
44
45
46     ~AlbumImpl() {
47         delete this->album;
48         for (int i = 0; i < artists.size(); ++ i) {
49             delete this->artists[i];
50         }
51     }
52
53     AlbumImpl(string albumName);
54
55     Song & operator [] (int songIndex); // may throw SongNotFoundException
56
57     void addSong(string songName, string artistName); // may throw SongExistedException
58
59     void removeSong(string songName); // TODO, may throw SongNotFoundException
60
61 };
62
63 /*
64 Please also include
65 - class Song
66 - class ArtistImpl
67 */
68 #include "source"
69
70 string AlbumImpl::getName() {
71     ArtistImpl * album = (ArtistImpl *)this->album;
72     return album->getName();
73 }
74
75 int AlbumImpl::numOfSongs() { // return the number of songs
76     ArtistImpl * album = (ArtistImpl *)this->album;
77     return album->numOfSongs();
78 }
79
80 //////////////////////////////////////////////////
81 // Test
82 //////////////////////////////////////////////////
83
84 void testAlbum() {
85     cout << "-- testAlbum --" << endl;
86     AlbumImpl album1("album#1");
87
88     try {
89         album1.addSong("song#1", "artist#1");
90         album1.addSong("song#2", "artist#1");
91         album1.addSong("song#3", "artist#2");
92         cout << album1 << endl; // TODO

```

```
93         album1.removeSong("song#1");
94         cout << album1 << endl;
95         album1.removeSong("song#1");
96     } catch (SongNotFoundException & ex) {
97         cout << "caught: SongNotFoundException" << endl;
98     }
99
100 }
101
102 int main() {
103     testAlbum();
104 }
```

答案程序

```
1 asdfjas;dfjas
2 fasdfasdf
3 a
```

文本编辑框

保存和测试对错

\*\*\*\*\* 测试 \*\*\*\*\*

请使用以下快捷键盘操作在程序框之间拷贝程序

- Control + c 复制
- Control + v 粘贴

主程序

```
1 #include "source"
```

答案程序

```
1 #include <iostream>
2 using namespace std;
3
4 int main() {
5     int age;
6     double speed;
7     cin >> age;
8     cin >> speed;
9     cout << age << ", " << speed;
10 }
```

文本编辑框

测试数据

保存和测试

测试输出

保存于 9:6.55