```
 1:
 2: #include <iostream>
 3: #include <thread>
 4: #include<chrono>
 5: #include "pac.h"
 6: #include "ghost.h"
 7: #include "maze.h"
 8: #include "gameText.h"
 9: #include "fruit.h"
10:
11: using namespace std;
12: using namespace sf;
13:
14: class Starter
15: {
16:  public:
17:      Event sfEvt;
18:      Maze maze;
19:      Pac<Starter>* pac;
20:      Ghost<Starter> *Blinky, *Pinky, *Inky,*
21:      thread ghStatusThread;
22:      thread loopThread;
23:      int attackInterval=15; // 15 sec
24:      int scaterInterval=7; // 7 sec
25:      int blueInterval = 6;
26:      int delay;
27:      int curTime;
```

```
28:       Texture backText;
29:       Texture backFlashText;
30:
31:       Sprite backSpr;
32:       bool intro = false;
33:       bool isCollid = false;
34:       bool lifeWin = false;
35:       bool toNextLevel = false;
36:       GameSound *gmSound;
37:       GameText *gameText;
38:       Fruit* fruit;
39:       RenderWindow* window;
40:
41:       Starter( RenderWindow *win, Texture* sp
42:       {
43:           cout << "Starter class OK - Start g
44:           window = win;
45:           gmSound = GameSound::getInstance();
46:           gameStatus = Demo;
47:           backText.loadFromFile("texture/map1
48:           backFlashText.loadFromFile("texture
49:           backSpr.setTexture(backText);
50:           maze.initMaze();
51:           pac = new Pac<Starter>(sprTexture,t
52:           Blinky = new Ghost<Starter>(sprText
53:           Pinky = new Ghost<Starter>(sprTextu
54:           Inky = new Ghost<Starter>(sprTextur
```

```
55:           Clyde = new Ghost<Starter>(sprTextu
56:           gameText = new GameText();
57:           fruit = new Fruit(sprTexture);
58:
59:           loopThread = thread ( &Starter::loo
60:           while (win->isOpen())
61:           {
62:               //if (pac->dotsEat == 20) { nex
63:               if ( CntrGame::dotsEat == maze.
64:               while (win->pollEvent(sfEvt))
65:               {
66:                   if (sfEvt.type == Event::Cl
67:                   else if (sfEvt.type == Even
68:                   {
69:                       if (gameStatus == Demo
70:                       {
71:                           startGame();
72:                       }
73:                       pac->rotation(sfEvt.key
74:                   }
75:               }
76:           }
77:       };
78:
79:       ~Starter()
80:       {
81:           if (loopThread.joinable()) { loopThr
```

```
 82:         if (ghStatusThread.joinable()) { ghS
 83:         delete  Blinky;
 84:         delete Pinky;
 85:         delete Inky;
 86:         delete Clyde;
 87:         //delete pac;
 88:         //delete gmSound;
 89:         //delete gameText;
 90:     }
 91:
 92:     //---
 93:     void drawLife( RenderWindow *win)
 94:     {
 95:         if (pac->pacLife < 0) { return; }
 96:         Sprite spr;
 97:         for (int i = 0; i < pac->pacLife; i+
 98:         {
 99:             spr = pac->getLifeSpr();
100:             spr.setPosition( Vector2f(30*i,
101:             (*win).draw( spr );
102:         }
103:     }
104:
105:     //---call from Pac class
106:     void setBlueGhost()
107:     {
108:         if (CntrGame::pacIsDead) { return; }
```

```
109:            cout << "Blue Status"<<endl;
110:            if (ghStatusThread.joinable()) { ghS
111:            sleep(milliseconds(20));
112:            ghostStatus = Blue;
113:            creatGhostThr();
114:            gmSound->play(GameSound::PlSound::Bl
115:            gmSound->stop(GameSound::PlSound::Si
116:        }
117:
118:        //---cal from Ghost class
119:        void collidToPac()
120:        {
121:            CntrGame::pacIsDead = true;
122:            pac->pacLife--;
123:            isCollid = true;
124:            stopAll();
125:            if (pac->pacLife < 0)
126:            {
127:                gameOver();
128:            }
129:        }
130:
131:        //--- STOP GAME
132:        void gameOver()
133:        {
134:            gameStatus = Demo;
135:            CntrGame::score = 0;
```

```
136:             CntrGame::level = 0;
137:             lifeWin = false;
138:             gameText->scoreTxt.setString("SCORE:
139:             pac->stop();
140:             Blinky->stop();
141:             Pinky->stop();
142:             Inky->stop();
143:             Clyde->stop();
144:             resetPacGhost();
145:             resetLevel();
146:         }
147:
148: private:
149:
150:         //---
151:         void loop( RenderWindow* win)
152:         {
153:             win->setActive(true);
154:             while (win->isOpen())
155:             {
156:                 if (CntrGame::score >= 10000 &&
157:                 {
158:                     lifeWin = true;
159:                     gmSound->play(GameSound::PlS
160:                     pac->pacLife++;
161:                 }
162:                 win->clear();
```

```
163:             if (gameStatus == Play)
164:             {
165:                 drawLife(win);
166:                 maze.drawWall(win);
167:                 win->draw(backSpr);
168:                 win->draw(gameText->gameOver
169:                 win->draw(gameText->scoreTxt
170:                 win->draw(gameText->levelTxt
171:                 if (ghostStatus == Blue)
172:                 {
173:                     string dif = to_string(c
174:                     if (stoi(dif) == 0) { di
175:                     gameText->countTxt.setSt
176:                     win->draw(gameText->coun
177:                 }
178:                 if (gameText->bonusTxt.getSt
179:                 {
180:                     win->draw(gameText->bonu
181:                 }
182:                 if(fruit->getVisible() ){ wi
183:             }
184:             else
185:             {
186:                 win->draw(gameText->enterTxt
187:                 win->draw( Blinky->getNameTx
188:                 win->draw(Pinky->getNameTxt(
189:                 win->draw(Inky->getNameTxt()
```

```
190:                 win->draw(Clyde->getNameTxt(
191:             }
192:
193:             win->draw(pac->getSprite());
194:             win->draw(Blinky->getSprite());
195:             win->draw(Pinky->getSprite());
196:             win->draw(Inky->getSprite());
197:             win->draw(Clyde->getSprite());
198:             win->display();
199:
200:         }
201:      }
202:
203:     //---
204:     void changeGhostState()
205:     {
206:                 cout << "Start thread for Gho
207:                 delay =  scaterInterval;
208:                 if (ghostStatus == Blue)
209:                 {
210:                     delay = blueInterval;
211:                     CntrGame::isBlueGhost = t
212:                 }
213:                 changeStatus();
214:                 //cout << "Status=" << ghostS
215:                 while ( ghStatusThread.joinab
216:                 {
```

```
217:                curTime = time(0);
218:                curTime += delay;
219:
220:                while (true && ghStatusTh
221:                {
222:                    if (curTime <= time(0
223:                }// wait for change ghost
224:
225:                sleep(milliseconds(10));
226:                if (ghostStatus == Blue)
227:                {
228:                  CntrGame::isBlueGhost =
229:                  CntrGame::ghostBonus =
230:                  gmSound->stop(GameSound
231:                  if(CntrGame::gameRun)gm
232:                }
233:                if (ghostStatus == Attack
234:                {
235:                    ghostStatus = Scater;
236:                    delay = scaterInterva
237:                    cout << "Scater " <<
238:
239:                }
240:                else
241:                {
242:                    ghostStatus =  Attack
243:                    delay = attackInterva
```

```
244:                         cout << "Attack " <<
245:                     }
246:                     changeStatus();
247:
248:                     if (( maze.dotsCount - Cn
249:                     {
250:                         gmSound->setPich(Game
251:                     }
252:
253:                 }
254:                 cout << "Ended Thread GhostSta
255:                 cout << "**********************
256:     }
257:
258:     //---
259:     void changeStatus()
260:     {
261:         Blinky->changeGhostState();
262:         Pinky->changeGhostState();
263:         Inky->changeGhostState();
264:         Clyde->changeGhostState();
265:     }
266:
267:     //---
268:     void stopAll()
269:     {
270:         pac->stop();
```

```
271:            Blinky->stop();
272:            Pinky->stop();
273:            Inky->stop();
274:            Clyde->stop();
275:            CntrGame::gameRun = false;
276:            gameText->stopThread();
277:            fruit->stop();
278:            gmSound->stopAll();
279:            if (ghStatusThread.joinable() ){ ghS
280:            resetLevel();
281:            wait(2);
282:            if (pac->pacLife >= 0) { startLevel(
283:        }
284:
285:        //---
286:        void startGame()
287:        {
288:            maze.redrawDot();
289:            gameText->gameOverTxt.setString("");
290:            pac->pacLife = 2;
291:            gameStatus = Play;
292:            resetPacGhost();
293:            intro = true;
294:            CntrGame::gameRun = true;
295:            CntrGame::level=1;
296:            CntrGame::score=0;
297:            CntrGame::dotsEat = 0;
```

```
298:         blueInterval = 6;
299:         fruit->setLevel(CntrGame::level);
300:         gmSound->setPich(GameSound::Siren, 1
301:         startLevel();
302:     }
303:
304:     //---
305:     void startLevel()
306:     {
307:         if (intro)
308:         {
309:             intro = false;
310:             gmSound->play(GameSound::PlSound
311:         }
312:         isCollid = false;
313:         ghostStatus = Scater;
314:         gmSound->play(GameSound::PlSound::Si
315:         CntrGame::gameRun = true;
316:         resetPacGhost();
317:         pac->run();
318:         creatGhostThr();
319:         fruit->start();
320:         gameText->levelTxt.setString("LEVEL:
321:     }
322:
323:
324:     //---
```

```
325:     void nextLevel()
326:     {
327:         gmSound->setPich(GameSound::Siren, 1
328:         toNextLevel = true;
329:         CntrGame::dotsEat = 0;
330:         stopAll();
331:         maze.redrawDot();
332:         CntrGame::level++;
333:         gameText->levelTxt.setString("LEVEL:
334:         fruit->setLevel(CntrGame::level);
335:         if (CntrGame::level > 2 && CntrGame:
336:         else if (CntrGame::level >= 6 && Cnt
337:         else if (CntrGame::level >= 10) { bl
338:     }
339: 
340:     //---
341:     void creatGhostThr()
342:     {
343:         while (ghStatusThread.joinable()) {}
344:         ghStatusThread = thread(&Starter::ch
345:     }
346: 
347:     //---
348:     void resetLevel()
349:     {
350:         CntrGame::ghostBonus = 100;
351:         if(pac->pacLife<0)
```

```
352:         {
353:             gameText->gameOverTxt.setString(
354:         }
355:     }
356: 
357:     //---
358:     void resetPacGhost()
359:     {
360:         pac->reset();
361:         Blinky->reset();
362:         Pinky->reset();
363:         Inky->reset();
364:         Clyde->reset();
365:     }
366: 
367:     // wait for second
368:     void wait(int delayInt)
369:     {
370:         auto curTime = time(0);
371:         int counter=0;
372:         curTime += delayInt;
373:         while (true)
374:         {
375:             counter++;
376:             if (toNextLevel)
377:             {
378:                 if (counter % 30 == 0)
```

```
379:                     {
380:                      backSpr.setTexture(backFlas
381:                     }
382:                     else if (counter % 30 == 15)
383:                     {
384:                      backSpr.setTexture(backText
385:                     }
386:                 }
387:                 if (curTime < time(0)) { break;
388:                 sleep(milliseconds(10));
389:             }
390:             toNextLevel = false;
391:             backSpr.setTexture(backText);
392:         }
393:
394: };
395:
396:
```