```
namespace Re_NONA_.Elecment
{
    using System;
    using System.Threading;
    internal class And : Tools
    {
        public And()
        {
            stop2:
            Console.Write("输入A:");
            try
            {
                base.inputIndexA = Convert.ToUInt32(Console.ReadLine());
            }
            catch
            {
                Console.WriteLine("输入错误");
                goto stop2;
            }
            stop3:
            Console.Write("输入B:");
            try
            {
                base.inputIndexB = Convert.ToUInt32(Console.ReadLine());
            }
            catch
            {
                Console.WriteLine("输入错误");
                goto stop3;
            }
            stop4:
            Console.Write("输出:");
            try
            {
                base.outputIndex = Convert.ToUInt32(Console.ReadLine());
            }
            catch
            {
                Console.WriteLine("输入错误");
                goto stop4;
            }
            Console.Write("可延迟?(y/n)");
            switch (Console.ReadLine())
            {
                case "y":
                    base.ifWaitCtrl = true;
                    break;
            }
        }

        public And(uint iA, uint iB, uint o,bool iWC)
        {

```

```
54              base.ToolsNumber = Tools.Number;
55              Tools.Number++;
56              base.inputIndexA = iA;
57              base.inputIndexB = iB;
58              base.outputIndex = o;
59              base.waitForOutput = Port.ports[iA] && Port.ports[iB];
60              base.ifWaitCtrl = iWC;
61          }
62
63          public void PreRun()
64          {
65              base.waitForOutput = Port.ports[base.inputIndexA] && Port.ports
                  [base.inputIndexB];
66          }
67      }
68      internal class Or : Tools
69      {
70          public Or()
71          {
72              stop2:
73              Console.Write("输入A:");
74              try
75              {
76                  base.inputIndexA = Convert.ToUInt32(Console.ReadLine());
77              }
78              catch
79              {
80                  Console.WriteLine("输入错误");
81                  goto stop2;
82              }
83              stop3:
84              Console.Write("输入B:");
85              try
86              {
87                  base.inputIndexB = Convert.ToUInt32(Console.ReadLine());
88              }
89              catch
90              {
91                  Console.WriteLine("输入错误");
92                  goto stop3;
93              }
94              stop4:
95              Console.Write("输出:");
96              try
97              {
98                  base.outputIndex = Convert.ToUInt32(Console.ReadLine());
99              }
100             catch
101             {
102                 Console.WriteLine("输入错误");
103                 goto stop4;
104             }
105             Console.Write("可延迟?(y/n)");
```

```
106             switch (Console.ReadLine())
107             {
108                 case "y":
109                     base.ifWaitCtrl = true;
110                     break;
111             }
112         }
113         public Or(uint iA, uint iB, uint o,bool iWC)
114         {
115             base.ToolsNumber = Tools.Number;
116             Tools.Number++;
117             base.inputIndexA = iA;
118             base.inputIndexB = iB;
119             base.outputIndex = o;
120             base.waitForOutput = Port.ports[base.inputIndexA] || Port.ports
                  [base.inputIndexB];
121             base.ifWaitCtrl = iWC;
122         }
123 
124         public void PreRun()
125         {
126             base.waitForOutput = Port.ports[base.inputIndexA] || Port.ports
                  [base.inputIndexB];
127         }
128     }
129     internal class Nand : Tools
130     {
131         public Nand()
132         {
133             stop2:
134             Console.Write("输入A:");
135             try
136             {
137                 base.inputIndexA = Convert.ToUInt32(Console.ReadLine());
138             }
139             catch
140             {
141                 Console.WriteLine("输入错误");
142                 goto stop2;
143             }
144             stop3:
145             Console.Write("输入B:");
146             try
147             {
148                 base.inputIndexB = Convert.ToUInt32(Console.ReadLine());
149             }
150             catch
151             {
152                 Console.WriteLine("输入错误");
153                 goto stop3;
154             }
155             stop4:
156             Console.Write("输出:");
```

```
157             try
158             {
159                 base.outputIndex = Convert.ToUInt32(Console.ReadLine());
160             }
161             catch
162             {
163                 Console.WriteLine("输入错误");
164                 goto stop4;
165             }
166             Console.Write("可延迟?(y/n)");
167             switch (Console.ReadLine())
168             {
169                 case "y":
170                     base.ifWaitCtrl = true;
171                     break;
172             }
173         }
174         public Nand(uint iA, uint iB, uint o,bool iWC)
175         {
176             base.ToolsNumber = Tools.Number;
177             Tools.Number++;
178             base.inputIndexA = iA;
179             base.inputIndexB = iB;
180             base.outputIndex = o;
181             base.waitForOutput = !Port.ports[base.inputIndexA] || !Port.ports
                  [base.inputIndexB];
182             base.ifWaitCtrl = iWC;
183         }
184 
185         public void PreRun()
186         {
187             base.waitForOutput = !Port.ports[base.inputIndexA] || !Port.ports
                  [base.inputIndexB];
188         }
189     }
190     internal class Nor : Tools
191     {
192         public Nor()
193         {
194             stop2:
195             Console.Write("输入A:");
196             try
197             {
198                 base.inputIndexA = Convert.ToUInt32(Console.ReadLine());
199             }
200             catch
201             {
202                 Console.WriteLine("输入错误");
203                 goto stop2;
204             }
205             stop3:
206             Console.Write("输入B:");
207             try
```

```
208             {
209                 base.inputIndexB = Convert.ToUInt32(Console.ReadLine());
210             }
211             catch
212             {
213                 Console.WriteLine("输入错误");
214                 goto stop3;
215             }
216             stop4:
217             Console.Write("输出:");
218             try
219             {
220                 base.outputIndex = Convert.ToUInt32(Console.ReadLine());
221             }
222             catch
223             {
224                 Console.WriteLine("输入错误");
225                 goto stop4;
226             }
227             Console.Write("可延迟?(y/n)");
228             switch (Console.ReadLine())
229             {
230                 case "y":
231                     base.ifWaitCtrl = true;
232                     break;
233             }
234         }
235         public Nor(uint iA, uint iB, uint o,bool iWC)
236         {
237             base.ToolsNumber = Tools.Number;
238             Tools.Number++;
239             base.inputIndexA = iA;
240             base.inputIndexB = iB;
241             base.outputIndex = o;
242             base.waitForOutput = !Port.ports[base.inputIndexA] && !Port.ports
                  [base.inputIndexB];
243             base.ifWaitCtrl = iWC;
244         }
245 
246         public void PreRun()
247         {
248             base.waitForOutput = !Port.ports[base.inputIndexA] && !Port.ports
                  [base.inputIndexB];
249         }
250     }
251     internal class Not : Tools
252     {
253         public Not()
254         {
255             stop2:
256             Console.Write("输入:");
257             try
258             {
```

```
259                     base.inputIndexA = Convert.ToUInt32(Console.ReadLine());
260                 }
261                 catch
262                 {
263                     Console.WriteLine("输入错误");
264                     goto stop2;
265                 }
266                 stop4:
267                 Console.Write("输出:");
268                 try
269                 {
270                     base.outputIndex = Convert.ToUInt32(Console.ReadLine());
271                 }
272                 catch
273                 {
274                     Console.WriteLine("输入错误");
275                     goto stop4;
276                 }
277                 Console.Write("可延迟?(y/n)");
278                 switch (Console.ReadLine())
279                 {
280                     case "y":
281                         base.ifWaitCtrl = true;
282                         break;
283                 }
284             }
285             public Not(uint iA, uint o,bool iWC)
286             {
287                 base.ToolsNumber = Tools.Number;
288                 Tools.Number++;
289                 base.inputIndexA = iA;
290                 base.outputIndex = o;
291                 base.waitForOutput = !Port.ports[base.inputIndexA];
292                 base.ifWaitCtrl = iWC;
293             }
294 
295             public void PreRun()
296             {
297                 base.waitForOutput = !Port.ports[base.inputIndexA];
298             }
299         }
300         internal class Is : Tools
301         {
302             public Is()
303             {
304                 stop2:
305                 Console.Write("输入:");
306                 try
307                 {
308                     base.inputIndexA = Convert.ToUInt32(Console.ReadLine());
309                 }
310                 catch
311                 {
```

```
312                 Console.WriteLine("输入错误");
313                 goto stop2;
314             }
315             stop4:
316             Console.Write("输出:");
317             try
318             {
319                 base.outputIndex = Convert.ToUInt32(Console.ReadLine());
320             }
321             catch
322             {
323                 Console.WriteLine("输入错误");
324                 goto stop4;
325             }
326             Console.Write("可延迟?(y/n)");
327             switch (Console.ReadLine())
328             {
329                 case "y":
330                     base.ifWaitCtrl = true;
331                     break;
332             }
333         }
334         public Is(uint iA, uint o,bool iWC)
335         {
336             base.ToolsNumber = Tools.Number;
337             Tools.Number++;
338             base.inputIndexA = iA;
339             base.outputIndex = o;
340             base.waitForOutput = Port.ports[base.inputIndexA];
341             base.ifWaitCtrl = iWC;
342         }
343 
344         public void PreRun()
345         {
346             base.waitForOutput = Port.ports[base.inputIndexA];
347         }
348     }
349     internal class Port
350     {
351         public static bool[] ports = new bool[0x2710];
352     }
353     internal class Tools
354     {
355         public uint inputIndexA;
356         public uint inputIndexB;
357         public static uint Number = 0;
358         public uint outputIndex;
359         public uint ToolsNumber;
360         public bool waitForOutput;
361         public bool ifWaitCtrl = false;
362         public void Run()
363         {
364             if(this.ifWaitCtrl==true)
```

```
365                 Console.WriteLine("MKQ");
366
367                 Port.ports[this.outputIndex] = this.waitForOutput;
368
369             }
370         }
371     }
372     namespace Re_NONA_
373     {
374         using Re_NONA_.Elecment;
375         using System.Threading;
376         using System;
377         internal class MKQ
378         {
379
380         }
381         internal class Ctrl
382         {
383             public static void Help()
384             {//"本游戏由端口替代导线,允许玩家在端口之间创建逻辑门\n" +
385                 //"目前支持的逻辑门有:\n" +
386                 //"->And:仅当两个输入端同时有电时才输出电流\n" +
387                 //"->Or:只要有一个输入端有电便输出电流\n" +
388                 //"->Nand:仅当两个输入端口都有电时才不输出电流\n" +
389                 //"->Nor:仅当两个输入端口都没电时才输出电流\n" +
390                 //"->Not:输入端口与输出端口状态相反\n\n" +
391                 //"目前的版本不支持会造成矛盾的电路所带来的快速变化电流(端口除非改变,通电状态应该是一定的)\n" +
392                 //"如果输入错误会导致程序崩溃而非重新输入\n" +
393                 //"true:有电false:没电\n" +
394                 //"支持的指令如下:\n" +
395                 //"->new:创建新元件\n" +
396                 //"->check:检查元件状态\n" +
397                 //"->run:运行元件\n" +
398                 //"->ports:显示端口状态(待改进)\n" +
399                 Console.WriteLine();
400                 Console.WriteLine("******欢迎使用Re-NONA-0.3.0******");
401                 Console.WriteLine(
402                     "Re-NONA-是一款逻辑门电路创建游戏,\n" +
403                     "在游戏中,玩家可以了解到计算机的原理以及亲身体验发展过程");
404                 Console.WriteLine(
405                     "元件说明:元件是建立在端口之间的能对电路变化产生反应的器件\n" +
406                     "->And:仅当两个输入端同时有电时才输出电流\n" +
407                     "->Or:只要有一个输入端有电便输出电流\n" +
408                     "->Nand:仅当两个输入端口都有电时才不输出电流\n" +
409                     "->Nor:仅当两个输入端口都没电时才输出电流\n" +
410                     "->Not:输入端口与输出端口状态相反\n" +
411                     "->Is:输入端口与输出端口状态相同\n ");
412                 Console.WriteLine(
413                     "端口说明:端口是连接在元件之间传递电路变化的器件\n" +
414                     "可以认为一个端口就是一根导线");
415                 Console.WriteLine("功能介绍:");
416                 Console.WriteLine(
```

```
417                 "->New:在端口之间创建新的逻辑门元件\n" +
418                 "->News(消歧义):一次性创建多个元件\n" +
419                 "->Run:运行电路\n" +
420                 "->AutoRun:自动运行,暂无法退出\n" +
421                 "");
422         }
423         public static void Set()
424         {
425             bool ifOver = false;
426             Console.WriteLine("设置结束请输入stop");
427             while(!ifOver)
428             {
429                 byte portNum = 0;
430                 stop1:
431                 Console.Write("端口号:");
432                 try
433                 {
434                     portNum = Convert.ToByte(Console.ReadLine());
435                 }
436                 catch
437                 {
438                     Console.WriteLine("输入错误");
439                     goto stop1;
440                 }
441                 Console.WriteLine("目前状态:[{0}]{1}",portNum,Port.ports
                      [portNum]?"1":"0");
442                 Console.WriteLine("是否更改(y/n)");
443                 switch (Console.ReadLine())
444                 {
445                     case "y":
446                         Port.ports[portNum] = !Port.ports[portNum];
447                         break;
448                     default:
449                         break;
450                 }
451                 if (Console.ReadLine() == "stop")
452                 {
453                     ifOver = true;
454                 }
455 
456 
457             }
458         }
459         public static void New()
460         {
461             Console.WriteLine("创建类型待输入...(and/nand/or/nor/not/is)");
462             switch (Console.ReadLine())
463             {
464                 case "and":
465                     ToolsData.tools[Tools.Number] = new And();
466                     break;
467                 case "nand":
468                     ToolsData.tools[Tools.Number] = new Nand();
```

```
469                     break;
470                 case "or":
471                     ToolsData.tools[Tools.Number] = new Or();
472                     break;
473                 case "nor":
474                     ToolsData.tools[Tools.Number] = new Nor();
475                     break;
476                 case "not":
477                     ToolsData.tools[Tools.Number] = new Not();
478                     break;
479                 case "is":
480                     ToolsData.tools[Tools.Number] = new Is();
481                     break;
482                 default:
483                     Console.WriteLine("未定义之指令");
484                     break;
485             }
486         }
487         public static void News()
488         {
489             Console.WriteLine("批量创建类型待输入...(and/nand/or/nor/not/is)");
490             switch (Console.ReadLine())
491             {
492                 case "and":
493                     stop16:
494                     try
495                     {
496                         Console.Write("创建个数:");
497                         uint num = Convert.ToUInt32(Console.ReadLine());
498                         Console.Write("输入A起始:");
499                         uint iA = Convert.ToUInt32(Console.ReadLine());
500                         Console.Write("间隔:");
501                         uint num3 = Convert.ToUInt32(Console.ReadLine());
502                         Console.Write("输入B起始:");
503                         uint iB = Convert.ToUInt32(Console.ReadLine());
504                         Console.Write("间隔:");
505                         uint num5 = Convert.ToUInt32(Console.ReadLine());
506                         Console.Write("输出起始:");
507                         uint o = Convert.ToUInt32(Console.ReadLine());
508                         Console.Write("间隔:");
509                         uint num7 = Convert.ToUInt32(Console.ReadLine());
510                         for (uint i = 0; i < num; i++)
511                         {
512                             ToolsData.tools[Tools.Number] = new And(iA, iB,
                    o,false);
513                             iA += num3;
514                             iB += num5;
515                             o += num7;
516                         }
517                         break;
518                     }
519                     catch
520                     {
```

```
521                         Console.WriteLine("输入错误");
522                         goto stop16;
523                     }
524                 case "nand":
525                     stop17:
526                     try
527                     {
528                         Console.Write("创建个数:");
529                         uint num8 = Convert.ToUInt32(Console.ReadLine());
530                         Console.Write("输入A起始:");
531                         uint num9 = Convert.ToUInt32(Console.ReadLine());
532                         Console.Write("间隔");
533                         uint num10 = Convert.ToUInt32(Console.ReadLine());
534                         Console.Write("输入B起始:");
535                         uint num11 = Convert.ToUInt32(Console.ReadLine());
536                         Console.Write("间隔");
537                         uint num12 = Convert.ToUInt32(Console.ReadLine());
538                         Console.Write("输出起始:");
539                         uint num13 = Convert.ToUInt32(Console.ReadLine());
540                         Console.Write("间隔");
541                         uint num14 = Convert.ToUInt32(Console.ReadLine());
542                         for (int j = 0; j < num8; j++)
543                         {
544                             ToolsData.tools[Tools.Number] = new Nand(num9,
                    num11, num13,false);
545                             num9 += num10;
546                             num11 += num12;
547                             num13 += num14;
548                         }
549                         break;
550                     }
551                     catch
552                     {
553                         Console.WriteLine("输入错误");
554                         goto stop17;
555                     }
556                 case "or":
557                     stop18:
558                     try
559                     {
560                         Console.Write("创建个数:");
561                         uint num15 = Convert.ToUInt32(Console.ReadLine());
562                         Console.Write("输入A起始:");
563                         uint num16 = Convert.ToUInt32(Console.ReadLine());
564                         Console.Write("间隔");
565                         uint num17 = Convert.ToUInt32(Console.ReadLine());
566                         Console.Write("输入B起始:");
567                         uint num18 = Convert.ToUInt32(Console.ReadLine());
568                         Console.Write("间隔");
569                         uint num19 = Convert.ToUInt32(Console.ReadLine());
570                         Console.Write("输出起始:");
571                         uint num20 = Convert.ToUInt32(Console.ReadLine());
572                         Console.Write("间隔");
```

```
573                         uint num21 = Convert.ToUInt32(Console.ReadLine());
574                         for (int k = 0; k < num15; k++)
575                         {
576                             ToolsData.tools[Tools.Number] = new Or(num16,
                    num18, num20,false);
577                             num16 += num17;
578                             num18 += num19;
579                             num20 += num21;
580                         }
581                         break;
582                     }
583                     catch
584                     {
585                         Console.WriteLine("输入错误");
586                         goto stop18;
587                     }
588                 case "nor":
589                     stop19:
590                     try
591                     {
592                         Console.Write("创建个数:");
593                         uint num22 = Convert.ToUInt32(Console.ReadLine());
594                         Console.Write("输入A起始:");
595                         uint num23 = Convert.ToUInt32(Console.ReadLine());
596                         Console.Write("间隔");
597                         uint num24 = Convert.ToUInt32(Console.ReadLine());
598                         Console.Write("输入B起始:");
599                         uint num25 = Convert.ToUInt32(Console.ReadLine());
600                         Console.Write("间隔");
601                         uint num26 = Convert.ToUInt32(Console.ReadLine());
602                         Console.Write("输出起始:");
603                         uint num27 = Convert.ToUInt32(Console.ReadLine());
604                         Console.Write("间隔");
605                         uint num28 = Convert.ToUInt32(Console.ReadLine());
606                         for (int m = 0; m < num22; m++)
607                         {
608                             ToolsData.tools[Tools.Number] = new Nor(num23,
                    num25, num27,false);
609                             num23 += num24;
610                             num25 += num26;
611                             num27 += num28;
612                         }
613                         break;
614                     }
615                     catch
616                     {
617                         Console.WriteLine("输入错误");
618                         goto stop19;
619                     }
620                 case "not":
621                     stop20:
622                     try
623                     {
```

```
624                         Console.Write("创建个数:");
625                         uint num29 = Convert.ToUInt32(Console.ReadLine());
626                         Console.Write("输入起始:");
627                         uint num30 = Convert.ToUInt32(Console.ReadLine());
628                         Console.Write("间隔");
629                         uint num31 = Convert.ToUInt32(Console.ReadLine());
630                         Console.Write("输出起始:");
631                         uint num32 = Convert.ToUInt32(Console.ReadLine());
632                         Console.Write("间隔");
633                         uint num33 = Convert.ToUInt32(Console.ReadLine());
634                         for (int n = 0; n < num29; n++)
635                         {
636                             ToolsData.tools[Tools.Number] = new Not(num30,
                        num32,false);
637                             num30 += num31;
638                             num32 += num33;
639                         }
640                         break;
641                     }
642                     catch
643                     {
644                         Console.WriteLine("输入错误");
645                         goto stop20;
646                     }
647                 case "is":
648                     stop33:
649                     try
650                     {
651                         Console.Write("创建个数:");
652                         uint num29 = Convert.ToUInt32(Console.ReadLine());
653                         Console.Write("输入起始:");
654                         uint num30 = Convert.ToUInt32(Console.ReadLine());
655                         Console.Write("间隔");
656                         uint num31 = Convert.ToUInt32(Console.ReadLine());
657                         Console.Write("输出起始:");
658                         uint num32 = Convert.ToUInt32(Console.ReadLine());
659                         Console.Write("间隔");
660                         uint num33 = Convert.ToUInt32(Console.ReadLine());
661                         for (int n = 0; n < num29; n++)
662                         {
663                             ToolsData.tools[Tools.Number] = new Is(num30,
                        num32,false);
664                             num30 += num31;
665                             num32 += num33;
666                         }
667                         break;
668                     }
669                     catch
670                     {
671                         Console.WriteLine("输入错误");
672                         goto stop33;
673                     }
674                 default:
```

```
675                     Console.WriteLine("未定义之指令");
676                     break;
677             }
678         }
679         public static void AutoRun(uint startPort, uint endPort)
680         {
681             bool ifOver = false;
682             int slpTime = 1;
683             stop21:
684             Console.WriteLine("主时钟刷新间隔待输入...(/ms)");
685             try
686             {
687                 slpTime = Convert.ToInt32(Console.ReadLine());
688             }
689             catch
690             {
691                 Console.WriteLine("输入错误");
692                 goto stop21;
693             }
694             Console.WriteLine("任意键启动...");
695             Console.ReadLine();
696             for (uint j = startPort; j <= endPort; j++)
697             {
698                 Console.Write("[{0}]{1}", j, Port.ports[j] ? "1" : "0");
699                 Console.Write("\n");
700                 Console.ReadLine();
701             }
702             Thread.Sleep(slpTime);
703             while (!ifOver)
704             {
705                 TestRun();
706                 for (uint j = startPort; j <= endPort; j++)
707                 {
708                     Console.Write("[{0}]{1}", j, Port.ports[j] ? "1" : "0");
709                     Console.Write("\n");
710                 }
711                 Thread.Sleep(slpTime);
712             }
713         }
714         public static void Run(uint startPort, uint endPort)
715         {
716             bool isOver = false;
717             string input = null;
718             Console.WriteLine("stop 指令用于退出...");
719             for (uint j = startPort; j <= endPort; j++)
720             {
721                 Console.Write("[{0}]{1}", j, Port.ports[j] ? "1" : "0");
722
723
724             }Console.Write("\n");
725             while (!isOver)
726             {
727                 TestRun();
```

```
728
729                 for (uint j = startPort; j <= endPort; j++)
730                     {
731                         Console.Write("[{0}]{1}", j, Port.ports[j]?"1":"0");
732                         Console.Write("\n");
733                     }
734                 input = Console.ReadLine();
735                 if (input == "stop")
736                 {
737                     isOver = true;
738                 }
739
740             }
741         }
742         public static void TestRun()
743         {
744            for (int ii = 0; ii < Tools.Number + 1; ii++)
745             {
746                 if (ToolsData.tools[ii].ifWaitCtrl == true)
747                 {
748                     Console.WriteLine("MQW");
749                     PreRun(ii);
750                     ToolsData.tools[ii].Run();
751                 }
752
753             }
754             for (int j = 0; j < 101; j++)
755             {
756
757                 for (int i = 0; i <= Tools.Number; i++)
758                 {
759
760                     if (ToolsData.tools[i].ifWaitCtrl == false)
761                     {
762                         PreRun(i);
763                         ToolsData.tools[i].Run();
764
765                     }
766
767
768                 }
769
770             }
771
772
773         }
774
775         public static void PreRun(int i)
776         {
777             if (ToolsData.tools[i] is And)
778             {
779                 ((And)ToolsData.tools[i]).PreRun();
780             }
```

```
781             else if (ToolsData.tools[i] is Nand)
782             {
783                 ((Nand)ToolsData.tools[i]).PreRun();
784             }
785             else if (ToolsData.tools[i] is Or)
786             {
787                 ((Or)ToolsData.tools[i]).PreRun();
788             }
789             else if (ToolsData.tools[i] is Nor)
790             {
791                 ((Nor)ToolsData.tools[i]).PreRun();
792             }
793             else if (ToolsData.tools[i] is Not)
794             {
795                 ((Not)ToolsData.tools[i]).PreRun();
796             }
797             else if (ToolsData.tools[i] is Is)
798             {
799                 ((Is)ToolsData.tools[i]).PreRun();
800             }
801         }
802     }
803     internal class ToolsData
804     {
805         public static Tools[] tools = new Tools[byte.MaxValue];
806     }
807     internal class Program
808     {
809         private static void Main(string[] args)
810         {
811             Console.WriteLine("**************NONA***************");
812             Console.WriteLine(
813                                 "11111111111111111111111111111111111111111111111111 ↵
                        \n" +
814                                 "11111111111111111111111111111111111111111111111111 ↵
                        \n" +
815                                 "11111111000011001000000111000110111000111111111 ↵
                        \n" +
816                                 "11111110001011011001100011010110111010011111111 ↵
                        \n" +
817                                 "11111110011000011011110010010110110011011111111 ↵
                        \n" +
818                                 "11111100111100011001110010111001100111101111111 ↵
                        \n" +
819                                 "11111100111100011100000110111001001111100111111 ↵
                        \n" +
820                                 "11111101111111011111111110111011001111100111111 ↵
                        \n" +
821                                 "11111111111111111111111111111111111111111111111 ↵
                        \n" +
822                                 "11111111111111111111111111111111111111111111111 ↵
                        \n");
823             while (true)
```

```
824             {
825                 Console.WriteLine("指令待输入...(new/news/run/autorun/help/set/
                      author/logo)");
826                 switch (Console.ReadLine())
827                 {
828                     case "new":
829                         Ctrl.New();
830                         break;
831 
832                     case "news":
833                         Ctrl.News();
834                         break;
835 
836                     case "run":
837                         stop22:
838                         try
839                         {
840                             Console.WriteLine("运行时显示端口待输入...");
841                             Console.Write("起始端口:");
842                             uint startPort = Convert.ToUInt32(Console.ReadLine
                        ());
843                             Console.Write("终止端口:");
844                             uint endPort = Convert.ToUInt32(Console.ReadLine
                        ());
845                             Ctrl.Run(startPort, endPort);
846                             break;
847                         }
848                         catch
849                         {
850                             Console.WriteLine("输入错误");
851                             goto stop22;
852                         }
853                     case "autorun":
854                         stop23:
855                         try
856                         {
857                             Console.WriteLine("运行时显示端口待输入...");
858                             Console.Write("起始端口:");
859                             uint startPort = Convert.ToUInt32(Console.ReadLine
                        ());
860                             Console.Write("终止端口:");
861                             uint endPort = Convert.ToUInt32(Console.ReadLine
                        ());
862                             Ctrl.AutoRun(startPort, endPort);
863                             break;
864                         }
865                         catch
866                         {
867                             Console.WriteLine("输入错误");
868                             goto stop23;
869                         }
870                     case "help":
871                         Ctrl.Help();
```

```
872                         break;
873                     case "set":
874                         Ctrl.Set();
875                         break;
876                     case"author":
877                         Console.WriteLine(
878                             "11111111111011111111111111111111011111111111111 ↵
                    \n11111111111000001111111111100000111111111111111 ↵
                    \n11111111111100000000000000000000111111111111111 ↵
                    \n11111111111100000000000000000000001111111111111 ↵
                    \n11111111111000000000000000000000000000111111111 ↵
                    \n11111100000000000000000000000000000000000000111 ↵
                    \n11000000000000000000000000000000000000000000011 ↵
                    \n00000000000000000000000000000000000000000000001 ↵
                    \n00000000000000000000000000000000000000000000011 ↵
                    \n11101110000111111110000001000000000000000000111 ↵
                    \n11111000000110111100000000000000000000000001111 ↵
                    \n11111100001001111100000000000000000000000111111 ↵
                    \n11111000011111111000000000000000000000000111111 ↵
                    \n11100000111111111000000000000000000011000001111 ↵
                    \n10000001111111110000000000000011111111100000111 ↵
                    \n10000011111111100000000000000011111111111101111 ↵
                    \n11001111111111100000000000000011111111111111111 ↵
                    \n11111111111111110000000000000011111111111111111 ↵
                    \n11111111111111111000000000000111111111111111111 ↵
                    \n11111111111111111010000011111111111111111111111 ↵
                    \n11111111111111111110100111111111111111111111111 ↵
                    \n11111111111111111111111111111111111111111111111 ↵
                    \n11111111111111111111111111111111111111111111111\n");
879                         break;
880                     case "logo":
881                         Console.WriteLine(
882                             "11111111111111111111111111111111111111111111111 ↵
                    \n" +
883                             "11111111111111111111111111111111111111111111111 ↵
                    \n" +
884                             "11111111000011001000000111000110111000111111111 ↵
                    \n" +
885                             "11111110001011011001100011010110111010011111111 ↵
                    \n" +
886                             "11111110011000011011110010010110110011011111111 ↵
                    \n" +
887                             "11111100111100011001110010111001100111101111111 ↵
                    \n" +
888                             "11111100111100011100000110111001001111100111111 ↵
                    \n" +
889                             "11111101111111011111111110111011001111100111111 ↵
                    \n" +
890                             "11111111111111111111111111111111111111111111111 ↵
                    \n" +
891                             "11111111111111111111111111111111111111111111111 ↵
                    \n");
892                         break;
```

```
893                         default:
894                             Console.WriteLine("未定义之指令");
895                             break;
896 
897 
898                     }
899                 }
900             }
901         }
902 
903 }
904 
905 
```