```cpp
// Program1.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"
#pragma warning( disable : 4996 )
typedef unsigned char byte;
typedef unsigned short word;
unsigned char RAM[4096] = { 0 };
short Accu = 0;
int pc = 0;









// splits a word into two bytes than writes to ram[adr] and ram[adr+1] in little endian
void Store(word value, word adr)
{
    RAM[adr + 1] = (value >> 8) & 0xff;
    RAM[adr] = value & 0xFF;
}
// returns the 16bit hex stored at adr and adr+1 as a big endian
word Read_at(word adr)
{
    word temp = (RAM[adr + 1] << 8) + (RAM[adr] & 0xff);
    return temp;
}
// reads input from user
word Read_in(word mode)
{
    word A;
    if (mode == 0) {
        scanf("%x", &A);
    }
    else if (mode == 1)
    {
        scanf("%d", &A);
    }
    else
    {
        scanf("%c", &A);
    }
    return A;
}

// prints value to screen. modes: 0 = hex, 1 = dec, 2 = ASCII
void Write(word value, word mode)
```

```
49  {
50      if (mode == 0) {
51          printf("%x", value);
52      }
53      else if (mode == 1)
54      {
55          printf("%d", value);
56      }
57      else
58      {
59          printf("%c", value);
60      }
61  }
62
63  // function to fetch a instruction from Ram at pc and exceute it
64  int Step()
65  {
66      word  value = Read_at(pc); // full instruction code
67      byte opcode = (value >> 12) & 0xf; // opcode
68      word operand = value & 0xFFF; // operand
69
70      switch (opcode)
71      {
72      case 0x0: //hlt
73          pc = pc + 2;
74          return 0;
75
76
77      case 0x1: //not
78          Accu = ~Accu;
79          break;
80
81      case 0x2: //shift left shl
82          Accu = Accu << operand;
83          break;
84
85      case 0x3: //shift right shr
86          Accu = Accu >> operand;
87          break;
88
89      case 0x4: //inc
90          Accu++;
91          break;
92
93      case 0x5: //dec
94          Accu--;
95          break;
96
97      case 0x6: //jmp
```

```
 98          pc = operand;
 99          return 1;
100
101      case 0x7: //jaz
102          if (Accu == 0)
103              pc = operand;
104          break;
105
106      case 0x8: //lda
107          Accu = Read_at(operand);
108          break;
109
110      case 0x9: //sta
111          Store(Accu, operand);
112          break;
113
114      case 0xA: //add
115          Accu += Read_at(operand);
116          break;
117
118      case 0xB: //and
119          Accu = Accu & Read_at(operand);
120          break;
121
122      case 0xC: //orr
123          Accu = Accu | Read_at(operand);
124          break;
125
126      case 0xD: //xor
127          Accu = Accu ^ Read_at(operand);
128          break;
129
130      case 0xE: //out
131          Write(Accu, operand);
132          break;
133
134      case 0xF: //inp
135          Accu = Read_in(operand);
136
137
138      default:
139          break;
140      }
141      pc = pc + 2;
142
143  }
144  // function to call Step() until hlt is fetched
145  void Run()
146  {
```

```
147     int x = 1;
148     while (x)
149     {
150         x = Step();
151     }
152 }
153
154
155
156
157
158 // main program
159 int main()
160 {
161     unsigned char program[] = {
162         0x80,0x1c,
163         0xc0,0x1e,
164         0x90,0x06,
165         0x00,0x00,
166         0xb0,0x20,
167         0x70,0x16,
168         0xe0,0x02,
169         0x80,0x1c,
170         0x40,0x00,
171         0x90,0x1c,
172         0x60,0x00,
173         0x00,0x00,
174         0x42,0x41,
175         0x00,0x43,
176         0x00,0x18,
177         0x80,0x00,
178         0x00,0xff
179     };
180     int size = sizeof(program) / sizeof(program[1]);
181     for (int i = 0; i < size; i += 2)
182     {
183         word temp = program[i + 1] + (program[i] << 8);
184         Store(temp, i);
185     }
186
187     char command = '0';
188
189     while (command != 'q')
190     {
191
192         int j = 0;
193         int lb = 0x0;
194
195         printf("?");
```

```
196         scanf("%c", &command);
197         switch (command)
198         {
199         case 'd'://dump: usage d start length
200             int start, len;
201             scanf("%d %d", &start, &len);
202             printf("000   ");
203             for (int i = 0; i < (start + len); i++)
204             {
205                 if (j == 16) {
206                     lb += 16;
207                     printf("\n%x   [%x] ", lb, RAM[i]);
208                     j = 0;
209                 }
210                 else
211                     printf("[%x] ", RAM[i]);
212                 j++;
213             }
214             printf("\n");
215             break;
216         case 'a':   // shows the current value of pc and accumulator
217             printf("Accumulator: %d\n", Accu);
218             printf("PC: %d\n", pc);
219             break;
220         case'q': //exits the program
221             break;
222 
223         case 's':   // calls Step()
224             Step();
225             break;
226         case 'r':   // calls Run()
227             Run();
228             break;
229         case 'e':   //edits a value in ram. usage: e address value_to_input
230             byte addr, val;
231             scanf("%x %x", &addr, &val);
232 
233             if (val > 0xff || addr > 0xfff)
234                 printf("---invalid input---\n");
235             else
236             {
237                 RAM[addr] = val;
238             }
239             break;
240         case 'h': // display a help string
241             printf("\nCommands: Quit{ q },  Dump Ram{ d start length }, Print
                      pc and accum{ a }\n Run{ r }, Step{ s }, edit ram{ e address
                      value } \n");
242             break;
```

```
243
244             default:
245                 break;
246             }
247         }
248
249         return 0;
250     }
251
252
253
```