```cpp
#include "PetStructure.h"


void Init(TAMA* Pet)
{
    // this sets up all the initial values apart from the name so no issues occur when the code runs
    Pet->Levels[0] = 50;
    Pet->Levels[1] = 50;
    Pet->Levels[3] = 50;
    Pet->HAP = 0;
    Pet->isDead = false;
    Pet->isAsleep = false;
    Pet->Start, Pet->duration = 0;


}
void NamingPet(TAMA* Pet)
{
    //this will allow the user to name the pet if they want to and if not it will be called Mayberry by default.
    char Key;
    cout << "Do you want to name your pet?\n\n Y = Yes     N = No\n\n";

    Key = _getch();
    if (Key == 'y' || Key == 'Y')
    {
        system("CLS");
        cout << "What do you want to call it?\n\n";
        cin >> Pet->Name;
    }
    else
    {
        Pet->Name = "Mayberry";
    }
    system("CLS");
    Pet->Start = 0; // this is the only thing that isn't initialized in 'init' as it puts the timer start to before the pet actually exists
}

void DisplayImage(TAMA* Pet)// this will display art of the pet of whether it is sleeping/passed out or not
{
    switch (Pet->isAsleep | Pet->PassedOut)
    {
    case true:
        cout << "       |\\    _,,,---,,_" << endl;
        cout << "ZZZzz /,`.-'`'    -.  ;-;;,_" << endl;
        cout << "     |,4-  ) )-,_. ,\\ ( `'-'" << endl;
        cout << "    '---''(_/--'  `-'\\_)" << endl;
        break;
    case false:
        cout << " _._     _,-\'\"\"\"`-._" << endl;
        cout << "(,-.`._,'(      |\\`-/|" << endl;
        cout << "    `-.-' \\ )-`( , o o)" << endl;
        cout << "          `-    \\`_`\"'-\"" << endl;
```

```
53          break;
54      }
55  }
56
57  void DisplayStats(TAMA* Pet,HANDLE Cons)// This will be used to display the current values for the pets stats
58  {
59  #pragma region string array and HANDEL
60
61
62      SetConsoleTextAttribute(Cons, 5);//sets the name to pink/purple
63      cout << "Pet Name: " << Pet->Name << endl; // shows the name of the pet
64
65      DisplayImage(Pet);
66      string Displays[4]; // this will be used to display the state of the pet
67  #pragma endregion
68
69
70  #pragma region Hunger String
71      /*
72
73          this just sets the value of the string for the Hunger string and the relevant colour for the status
74
75      */
76
77      if (Pet->Levels[0] >= 80)
78      {
79          SetConsoleTextAttribute(Cons, 2);//green
80          Displays[0] = "Well Fed";
81      }
82      else if (Pet->Levels[0] < 80 && Pet->Levels[0] >= 60)
83      {
84          SetConsoleTextAttribute(Cons, 3);//aqua
85
86          Displays[0] = "Slightly Peckish";
87      }
88      else if (Pet->Levels[0] < 60 && Pet->Levels[0] >= 40)
89      {
90          SetConsoleTextAttribute(Cons, 7);//white
91
92          Displays[0] = "Hungry";
93      }
94      else if (Pet->Levels[0] < 40 && Pet->Levels[0] >= 20)
95      {
96          SetConsoleTextAttribute(Cons, 6);//yellow
97
98          Displays[0] = "Rather Hungry";
99      }
100     else
101     {
102         SetConsoleTextAttribute(Cons, 4);//red
103
104         Displays[0] = "Starving";
105     }
106
```

```
107     cout << "\n\nHunger: " << Displays[0];
108 #pragma endregion
109
110 #pragma region Hydration String
111     /*
112
113         this just sets the value of the string for the Hydration string and
              the relevant colour for the status
114
115     */
116
117     if (Pet->Levels[1] >= 80)
118     {
119         SetConsoleTextAttribute(Cons, 2);//green
120
121         Displays[1] = "Hydrated";
122     }
123     else if (Pet->Levels[1] < 80 && Pet->Levels[1] >= 60)
124     {
125         SetConsoleTextAttribute(Cons, 3);//Aqua
126
127         Displays[1] = "Quenched"; // fairly certain this means pretty much the
              exact same as hydrated
128     }
129     else if (Pet->Levels[1] < 60 && Pet->Levels[1] >= 40)
130     {
131         SetConsoleTextAttribute(Cons, 7);//White
132
133         Displays[1] = "Slightly Parched";
134     }
135     else if (Pet->Levels[1] < 40 && Pet->Levels[1] >= 20)
136     {
137         SetConsoleTextAttribute(Cons, 6);//Yellow
138
139         Displays[1] = "Parched";
140     }
141     else
142     {
143         SetConsoleTextAttribute(Cons, 4);//Red
144
145         Displays[1] = "De-hydrated";
146     }
147
148     cout << "\nHydration: " << Displays[1];
149 #pragma endregion
150
151 #pragma region Happiness String
152
153     /*
154
155         this just sets the value of the string for the Happiness String and
              the relevant colour to the status
156
157     */
158
159     if (Pet->Levels[2] >= 80)
```

```
160     {
161         SetConsoleTextAttribute(Cons, 2);//Green
162 
163         Displays[2] = "Exstatic";
164     }
165     else if (Pet->Levels[2] < 80 && Pet->Levels[2] >= 60)
166     {
167         SetConsoleTextAttribute(Cons, 3);//Aqua
168 
169         Displays[2] = "Happy";
170     }
171     else if (Pet->Levels[2] < 60 && Pet->Levels[2] >= 40)
172     {
173         SetConsoleTextAttribute(Cons, 7);//White
174 
175         Displays[2] = "Neutral";
176     }
177     else if (Pet->Levels[2] < 40 && Pet->Levels[2] >= 20)
178     {
179         SetConsoleTextAttribute(Cons, 6);//Yellow
180 
181         Displays[2] = "Sad";
182     }
183     else
184     {
185         SetConsoleTextAttribute(Cons, 4);//Red
186 
187         Displays[2] = "Depressed";
188     }
189 
190     cout << "\nHappiness: " << Displays[2];
191 
192     cout << "\nPet meter: " << Pet->HAP << "/10";
193 #pragma endregion
194 
195 #pragma region Tiredness String
196 
197     /*
198 
199     this just sets the value of the string for the Tiredness string and the ↵
        relevant colour for the status
200 
201     */
202 
203     if (Pet->Levels[3] >= 80)
204     {
205         SetConsoleTextAttribute(Cons, 2);//green
206 
207         Displays[3] = "Wide Awake";
208     }
209     else if (Pet->Levels[3] < 80 && Pet->Levels[3] >= 60)
210     {
211         SetConsoleTextAttribute(Cons, 3);//green
212 
213         Displays[3] = "Awake";
214     }
```

```
215     else if (Pet->Levels[3] < 60 && Pet->Levels[3] >= 40)
216     {
217         SetConsoleTextAttribute(Cons, 7);//green
218 
219         Displays[3] = "Drowzy";
220     }
221     else if (Pet->Levels[3] < 40 && Pet->Levels[3] >= 20)
222     {
223         SetConsoleTextAttribute(Cons, 6);//green
224 
225         Displays[3] = "Tired";
226     }
227     else
228     {
229         SetConsoleTextAttribute(Cons, 4);//green
230 
231         Displays[3] = "Falling Asleep";
232     }
233 
234     cout << "\nTiredness: " << Displays[3] << "\n\n";
235 #pragma endregion
236 
237 #pragma region Controls
238 
239     //sets the colour for the controls and prints them
240 
241     SetConsoleTextAttribute(Cons, 1);//Blue
242 
243     cout << "Press f to feed the pet\nPress h to hydrate the pet\nPress p to
        pet the pet\nPress s to put the pet to sleep\nPress x to Euthanise the
        pet\n\n";
244 #pragma endregion
245 }
246 
247 void Decrease(TAMA* Pet, int n)// this will pick either the hunger or
      hydration stat
248 {
249     int Selection = rand() % n - 1;
250 
251     switch (Selection)
252     {
253     case 0:
254         Pet->Levels[0] -= rand() % 2 + 0b1; //0b1 is used to get it to stop
            putting those annoying green lines under it as "doesn't like cast a
            4 bit number to an 8 bit cast"
255 
256 
257         break;
258     case 1:
259         Pet->Levels[1] -= rand() % 2 + 0b1;
260 
261         break;
262     }
263 
264 }
265 
```

```
266 void StatCap(TAMA* Pet, int n)//will check to see that none of them has
      exceeded the max values of 100 and the min values of 0;
267 {
268     for (int i = 0; i < n; i++)
269     {
270         if (Pet->Levels[i] > 100)
271         {
272             Pet->Levels[i] = 100;
273         }
274         if (Pet->Levels[i] < 0)
275         {
276             Pet->Levels[i] = 0;
277         }
278     }
279     if (Pet->HAP > 10) // caps the HAP stat to 10
280     {
281         Pet->HAP = 10;
282     }
283     if (Pet->HAP < 0)
284     {
285         Pet->HAP = 0;
286     }
287 }
```