

```

1 #Dr. Burkman
2 #Mod 10 Homework
3
4 #declare and fill the four suits for the deck
5 $spades = @(
6 'Ace of Spades', 'King of Spades',
7 'Queen of Spades', 'Jack of Spades',
8 '10 of Spades', '9 of Spades',
9 '8 of Spades', '7 of Spades',
10 '6 of Spades', '5 of Spades',
11 '4 of Spades', '3 of Spades',
12 '2 of Spades')
13
14 $diamonds = @(
15 'Ace of Diamonds', 'King of Diamonds',
16 'Queen of Diamonds', 'Jack of Diamonds',
17 '10 of Diamonds', '9 of Diamonds',
18 '8 of Diamonds', '7 of Diamonds',
19 '6 of Diamonds', '5 of Diamonds',
20 '4 of Diamonds', '3 of Diamonds',
21 '2 of Diamonds')
22
23 $clubs = @(
24 'Ace of Clubs', 'King of Clubs',
25 'Queen of Clubs', 'Jack of Clubs',
26 '10 of Clubs', '9 of Clubs',
27 '8 of Clubs', '7 of Clubs',
28 '6 of Clubs', '5 of Clubs',
29 '4 of Clubs', '3 of Clubs',
30 '2 of Clubs')
31
32 $hearts = @(
33 'Ace of Hearts', 'King of Hearts',
34 'Queen of Hearts', 'Jack of Hearts',
35 '10 of Hearts', '9 of Hearts',
36 '8 of Hearts', '7 of Hearts',
37 '6 of Hearts', '5 of Hearts',
38 '4 of Hearts', '3 of Hearts',
39 '2 of Hearts')
40
41 #declare the play suits
42 $play_spades = @()
43 $play_diamonds = @()
44 $play_clubs = @()
45 $play_hearts = @()
46
47 function new_deck ()
48 {
49     foreach ($i in $spades)
50     {
51         $Global:play_spades += $i

```

```

52     }
53     foreach ($i in $diamonds)
54     {
55         $Global:play_diamonds += $i
56     }
57     foreach ($i in $clubs)
58     {
59         $Global:play_clubs += $i
60     }
61     foreach ($i in $hearts)
62     {
63         $Global:play_hearts += $i
64     }
65     $Global:spades_gone=0
66     $Global:diamonds_gone=0
67     $Global:clubs_gone=0
68     $Global:hearts_gone=0
69 }
70
71 function remove_card ($array, $card_to_remove)
72 {
73     $temp_array = @()
74     for ($i=0; $i -lt $card_to_remove; $i++)
75     {
76         $temp_array += $array[$i]
77     }
78     for ($i=$card_to_remove+1; $i -lt $array.Count; $i++)
79     {
80         $temp_array += $array[$i]
81     }
82     return $temp_array
83 }
84
85 function get_card ()
86 {
87     clear
88     #check for valid input
89     $cards_requested = Read-Host "How many cards would you like to draw from
... this deck?"
90     if ($cards_requested -notmatch "^[+]?[0-9]" -or $cards_requested -ne
... [int]$cards_requested)
91     {
92         clear
93         Write-Host "Invalid Option. Press Enter to return to the main
... menu:"
94         Read-Host
95         return
96     }
97
98     #see if there are enough cards in the deck to meet the request
99     #write-host $play_spades.Count $play_diamonds.Count $play_clubs.count

```

```

99... $play_hearts.Count
100     $cards_remaining = $Global:play_spades.Count +
...   $Global:play_diamonds.Count + $Global:play_clubs.Count +
...   $Global:play_hearts.Count
101     if ($cards_requested/1 -gt $cards_remaining)
102     {
103         clear
104         write-host "There are only $cards_remaining cards left in the deck
... but you have requested $cards_requested."
105         write-host "`r`nPress the Enter key to return to the main menu: "
106         Read-Host
107         return
108     }
109     Write-Host "Your cards are:`r`n"
110
111     #loop and get the number of requested cards
112     while ($cards_requested -gt 0)
113     {
114         if ($spades_gone -eq 1 -and $diamonds_gone -eq 1 -and $clubs_gone
... -eq 1 -and $hearts_gone -eq 1)
115         {
116             Write-Host "All the cards have been drawn from this deck"
117             break
118         }
119
120         #get a random suit
121         $suit = Get-Random -Minimum 0 -Maximum 4
122         if ($suit -eq 0)
123         {
124             $suit_count = $play_spades.Count
125             if ($suit_count -eq 0)
126             {
127                 $spades_gone = 1
128                 continue
129             }
130             else
131             {
132                 $card = Get-Random -Minimum 0 -Maximum ($suit_count)
133                 if ($play_spades.count -eq 1){
134                     Write-Host $play_spades
135                     $Global:play_spades = @()
136                     $cards_requested = $cards_requested - 1
137                     continue
138                 }
139                 Write-Host $play_spades[$card]
140                 if ($play_spades.count -gt 1)
141                 {
142                     $Global:play_spades = remove_card -array $play_spades
... -card_to_remove $card}
143                     $cards_requested = $cards_requested - 1
144                 }

```

```

145     }
146     if ($suit -eq 1)
147     {
148         $suit_count = $play_diamonds.Count
149         if ($suit_count -eq 0)
150         {
151             $diamonds_gone = 1
152             continue
153         }
154         else
155         {
156             $card = Get-Random -Minimum 0 -Maximum ($suit_count)
157             if ($play_diamonds.count -eq 1){
158                 Write-Host $play_diamonds
159                 $Global:play_diamonds = @()
160                 $cards_requested = $cards_requested - 1
161                 continue
162             }
163             Write-Host $play_diamonds[$card]
164             if ($play_diamonds.count -gt 1)
165             {
166                 $Global:play_diamonds = remove_card -array $play_diamonds
167 ... -card_to_remove $card}
168                 $cards_requested = $cards_requested - 1
169             }
170         }
171         if ($suit -eq 2)
172         {
173             $suit_count = $play_clubs.Count
174             if ($suit_count -eq 0)
175             {
176                 $clubs_gone = 1
177                 continue
178             }
179             else
180             {
181                 $card = Get-Random -Minimum 0 -Maximum ($suit_count)
182                 if ($play_clubs.count -eq 1){
183                     Write-Host $play_clubs
184                     $Global:play_clubs = @()
185                     $cards_requested = $cards_requested - 1
186                     continue
187                 }
188                 Write-Host $play_clubs[$card]
189                 if ($play_clubs.count -gt 1)
190                 {
191                     $Global:play_clubs = remove_card -array $play_clubs
192 ... -card_to_remove $card}
193                     $cards_requested = $cards_requested - 1
194                 }
195             }
196         }

```

```

194         if ($suit -eq 3)
195         {
196             $suit_count = $play_hearts.Count
197             if ($suit_count -eq 0)
198             {
199                 $hearts_gone = 1
200                 continue
201             }
202             else
203             {
204                 $card = Get-Random -Minimum 0 -Maximum ($suit_count)
205                 if ($play_hearts.count -eq 1){
206                     Write-Host $play_hearts
207                     $Global:play_hearts = @()
208                     $cards_requested = $cards_requested - 1
209                     continue
210                 }
211                 Write-Host $play_hearts[$card]
212                 if ($play_hearts.count -gt 1)
213                 {
214                     $Global:play_hearts = remove_card -array $play_hearts
215                     ... -card_to_remove $card}
216                     $cards_requested = $cards_requested - 1
217                 }
218             }
219
220
221     }
222
223
224 Read-Host
225
226
227 }
228
229 #get the new deck for the first time
230 new_deck
231
232
233 while ($true)
234 {
235     clear
236     Write-Host ("
237 Welcome to the card deck simulator.
238
239 Please select from the following options:
240
241     1. Draw a selected number of cards from the current deck
242     2. Get a new deck of cards
243     3. Exit

```

```
244 ")
245
246 $user_menu_choice = read-host "Option#"
247
248 if ($user_menu_choice -eq 1)
249 {
250     get_card
251 }
252 elseif ($user_menu_choice -eq 2)
253 {
254     new_deck
255 }
256 elseif ($user_menu_choice -eq 3)
257 {
258     clear
259     break
260 }
261 else
262 {
263     clear
264     read-host "That is not a valid selection. Press Enter to continue"
265 }
266
267
268
269 }
```