

```
1 // Glenn Hewey
2 // Cecs 424
3 // Lab 3
4 // March 23 2018
5 #include <stdio.h>
6 #include <stdlib.h>
7
8 struct Employee {
9     void** vtable;
10    int age;
11 };
12
13 struct HourlyEmployee {
14     void** vtable;
15     int age;
16     double hourly_rate;
17     double hours;
18 };
19
20 struct CommissionEmployee {
21     void** vtable;
22     int age;
23     double sales_amount;
24 };
25
26 struct SeniorSalesman {
27     void** vtable;
28     int age;
29     double sales_amount;
30 };
31
32 // Function declaration
33 void Speak_Hourly(struct Employee* employee);
34 void Speak_Commission(struct Employee* employee);
35 void Speak_SeniorSalesman(struct Employee* employee);
36
37 double GetPay_Hourly(struct Employee* employee);
38 double GetPay_Commission(struct Employee* employee);
39 double GetPay_SeniorSalesman(struct Employee* employee);
40
41 void Construct_Hourly(struct HourlyEmployee* hEmployee);
42 void Construct_Commission(struct CommissionEmployee* cEmployee);
43 void Construct_SeniorSalesman(struct SeniorSalesman* sEmployee);
44
45 // vtables for hourly, commission, and senior salesman
46 void* Vtable_Hourly[2] = {Speak_Hourly, GetPay_Hourly};
47 void* Vtable_Commission[2] = {Speak_Commission, GetPay_Commission};
48 void* Vtable_SeniorSalesman[2] = {Speak_Commission, GetPay_SeniorSalesman};
49
50 // Outputs the speak method for hourly employees
51 void Speak_Hourly(struct Employee* employee){
52     printf("I work for %.2lf dollars per hour.\n", ((struct HourlyEmployee *)
employee)->hourly_rate);
53 }
54
55 // Outputs the speak method for commission and senior salesman employees
56 void Speak_Commission(struct Employee* employee){
57     printf("I make commission on %.2lf dollars in sales!\n", ((struct
CommissionEmployee *) employee)->sales_amount);
58 }
```

```
59
60 // Returns a double of how much the hourly employee makes
61 double GetPay_Hourly(struct Employee* employee){
62     return ((struct HourlyEmployee *) employee)->hourly_rate * ((struct
HourlyEmployee *) employee)->hours;
63 }
64
65 // Returns a double of how much the commission employee makes
66 double GetPay_Commission(struct Employee* employee){
67     return (0.1*((struct CommissionEmployee *) employee)->sales_amount) +
40000.0;
68 }
69
70 // Returns a double of how much the senior salesman makes
71 double GetPay_SeniorSalesman(struct Employee* employee){
72     double percent = 0.2;
73     if( ((struct SeniorSalesman *) employee)->age >= 40 ){
74         percent += 0.05;
75     }
76     return (percent*((struct SeniorSalesman *) employee)->sales_amount) +
50000.0;
77 }
78
79 // Constructs an hourly employee initializes values to 0 and points to the
current vtable
80 void Construct_Hourly(struct HourlyEmployee *hEmployee){
81     hEmployee->vtable = Vtable_Hourly;
82     hEmployee->age = 0;
83     hEmployee->hourly_rate = 0;
84     hEmployee->hours = 0;
85 }
86
87 // Constructs a commission employee initializes values to 0 and points to the
current vtable
88 void Construct_Commission(struct CommissionEmployee *cEmployee){
89     cEmployee->vtable = Vtable_Commission;
90     cEmployee->age = 0;
91     cEmployee->sales_amount = 0;
92 }
93
94 // Constructs a senior salesman initializes values to 0 and points to the
current vtable
95 void Construct_SeniorSalesman(struct SeniorSalesman *sEmployee){
96     sEmployee->vtable = Vtable_SeniorSalesman;
97     sEmployee->age = 0;
98     sEmployee->sales_amount = 0;
99 }
100
101 int main()
102 {
103     // declare employee pointer variable
104     struct Employee* e;
105
106     // user input to choose what type of employee
107     int i;
108     printf("input employee type\n1) Hourly \n2) Commission \n3) Senior Salesman
\n");
109     scanf("%d", &i);
110
111     switch(i) {
```

```
112 case 1:
113     printf("You chose Hourly\n");
114
115     // allocate using malloc
116     e = malloc(sizeof(struct HourlyEmployee));
117
118     // Use constructor
119     Construct_Hourly((struct HourlyEmployee *)e);
120
121     // Ask the user how old the employee is
122     printf("input age: ");
123     scanf("%d", &(e)->age);
124
125     // Ask user for rate
126     printf("input rate: ");
127     scanf("%lf", &((struct HourlyEmployee *)e)->hourly_rate);
128
129     // Ask user for hours
130     printf("input hours: ");
131     scanf("%lf", &((struct HourlyEmployee *)e)->hours);
132
133     break;
134 case 2:
135     printf("You chose Commission\n");
136
137     // allocate using malloc
138     e = malloc(sizeof(struct CommissionEmployee));
139
140     // Use constructor
141     Construct_Commission((struct CommissionEmployee *)e);
142
143     // Ask the user how old the employee is
144     printf("input age: ");
145     scanf("%d", &(e)->age);
146
147     // Ask user for sales
148     printf("input sales: ");
149     scanf("%lf", &((struct CommissionEmployee *)e)->sales_amount);
150
151     break;
152 case 3:
153     printf("You chose Senior Salesman\n");
154
155     // allocate using malloc
156     e = malloc(sizeof(struct SeniorSalesman));
157
158     // Use constructor
159     Construct_SeniorSalesman((struct SeniorSalesman *)e);
160
161     // Ask the user how old the employee is
162     printf("input age: ");
163     scanf("%d", &(e)->age);
164
165     // Ask user for sales
166     printf("input sales: ");
167     scanf("%lf", &((struct SeniorSalesman *)e)->sales_amount);
168
169     break;
170 default:
171     break;
```

```
172 }
173
174 // Invoke speak method using employee vtable
175 ((void (*)(struct Employee*))e->vtable[0])((struct Employee *)e);
176
177 // Invoke getPay method using employee vtable
178 printf("I make: %.2lf\n", ((double (*)(struct Employee*))e->vtable[1])
((struct Employee *)e));
179
180 free(e);
181
182 return 0;
183 }
```