```
from abc import ABC, abstractmethod
class Employee(ABC):
    def __init__(self,name,emp_id,dept):
        self.name=name
        self.emp_id=emp_id
        self.dept=dept
    @abstractmethod
    def cal_salary(self):
        pass
    def disp_details(self):
        print("Employee Details:")
        print("Name:",self.name)
        print("Employee ID:",self.emp_id)
        print("Department:",self.dept)
class FullTimeEmployee(Employee):
    def __init__(self,name,emp_id,dept,monthly_salary):
        super().__init__(name,emp_id,dept)
        self.monthly_salary=monthly_salary
    def cal_salary(self):
        return self.monthly_salary
class PartTimeEmployee(Employee):
    def __init__(self,name,emp_id,dept,hrs_wage,hrs_worked):
        super().__init__(name,emp_id,dept)
        self.hrs_wage=hrs_wage
        self.hrs_worked=hrs_worked
    def cal_salary(self):
        return self.hrs_wage*self.hrs_worked
fte=FullTimeEmployee("Park","FT123","Hacker",50000)
fte.disp_details()
print(f"Employee Name:{fte.name} Pay:${fte.cal_salary():.2f}")
```

```
pte=PartTimeEmployee("Kim", "PT456", "IT", 20, 200)
pte.disp_details()
print(f"Employee Name:{pte.name} Pay:${pte.cal_salary():.2f}")
```