```python
import abc
from numpy import mean, maximum

class Payoff(object, metaclass=abc.ABCMeta):
    @property
    @abc.abstractmethod
    def expiry(self):
        """Get the expiry date."""
        pass

    @expiry.setter
    @abc.abstractmethod
    def expiry(self, newExpiry):
        """Set the expiry date."""
        pass

    @abc.abstractmethod
    def payoff(self):
        pass


class VanillaPayoff(Payoff):
    def __init__(self, expiry, strike, payoff):
        self.__expiry = expiry
        self.__strike = strike
        self.__payoff = payoff

    @property
    def expiry(self):
        return self.__expiry

    @expiry.setter
    def expiry(self, new_expiry):
        self.__expiry = new_expiry

    @property
    def strike(self):
        return self.__strike

    @strike.setter
    def strike(self, new_strike):
        self.__strike = new_strike

    def payoff(self, spot):
        return self.__payoff(self, spot)


def call_payoff(option, spot):
    return maximum(spot - option.strike, 0.0)

def put_payoff(option, spot):
    return maximum(option.strike - spot, 0.0)



class ExoticPayoff(Payoff):
    def __init__(self, expiry, strike, payoff):
        self.__expiry = expiry
        self.__strike = strike
        self.__payoff = payoff

    @property
    def expiry(self):
        return self.__expiry

    @expiry.setter
    def expiry(self, new_expiry):
        self.__expiry = new_expiry

    @property
    def strike(self):
        return self.__strike

    @strike.setter
    def strike(self, new_strike):
        self.__strike = new_strike

    def payoff(self, spot):
        return self.__payoff(self, spot)

def arithmeticAsianCallPayoff(option, spot):
   ## Assume that spot is a NumPy ndarray
   ## Call the `mean` method to get the arithmetic average
   average = spot.mean()
   return maximum(average - option.strike, 0.0)

def arithmeticAsianPutPayoff(option, spot):
    average = spot.mean()
    return maximum(option.strike - average, 0.0)
```