```python
import numpy as np

def call_option_payoff(S, K):
  # Option payoff function (European call option)
  return np.maximum(S - K, 0)

def put_option_payoff(S, K):
  # Option payoff function (European put option)
  return np.maximum(K - S, 0)

def monte_carlo_sims(S0, T, r, sigma, K, n_t, n_sims, option_type):
  # This function generates monte carlo paths for the risky asset
using
  # geometric Brownian motion and uses arithmetic average to
  # estimate the option (call or put) price.

  # Simulate the risky asset's price path using geometric Brownian
motion
  dt = T / n_t
  risky_returns = np.exp((r - 0.5 * sigma**2) * dt + sigma *
np.sqrt(dt) * np.random.randn(n_sims, n_t))
  risky_prices = S0 * np.cumprod(risky_returns, axis=1)

  # Compute the option's payoff at expiration for each simulation
  if option_type == 'call':
    option_payoffs = call_option_payoff(risky_prices[:, -1], K)
  else:
    option_payoffs = put_option_payoff(risky_prices[:, -1], K)

  # Calculate the expected option payoff and discount it to present
value
  expected_payoff = np.mean(option_payoffs) # arithmetic mean

  # Evaluate the present value of the option
  option_price = np.exp(-r * T) * expected_payoff

  return option_price

np.random.seed(0)

# Define option parameters
stock_price = 100  # Current price of the underlying asset
strike_price = 105   # Strike price of the option
expiry_time = 1.0   # Time to expiration (in years)
risk_free_rate = 0.05  # Risk-free interest rate
volatility = 0.25  # Volatility of the underlying asset

n_simulations = 50000 # number of monte carlo simulations (i.e. risky
asset paths)
n_steps = 252  # Number of time steps (daily steps for one year)
```

```python
# Price of a call option
call_option_price = monte_carlo_sims(stock_price, expiry_time,
risk_free_rate, volatility, strike_price, n_steps, n_simulations,
'call')
print(f"European call option price: {call_option_price:.2f}")

# Price of a put option
put_option_price = monte_carlo_sims(stock_price, expiry_time,
risk_free_rate, volatility, strike_price, n_steps, n_simulations,
'put')
print(f"European put option price: {put_option_price:.2f}")
```

```
European call option price: 10.09
European put option price: 9.87
```