
468
0
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
Abstraksi / Interface
Pemrograman Berorientasi Objek
muhamad.soleh@iti.ac.id
START

468 0

01

Abstraksi

02

Abstract method

03

time

04

Studi kasus

368 2

STORE

LUCK ROYALE

CHARACTER

VAULT

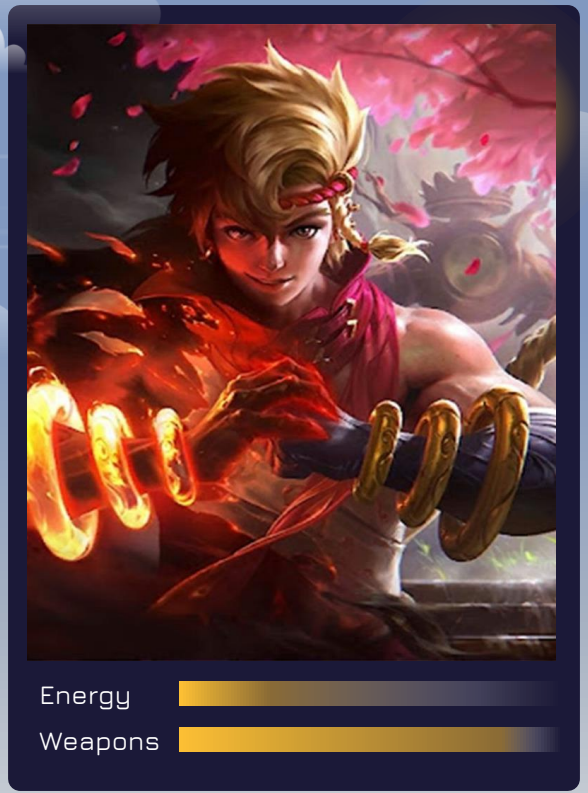
PET

COLLECTION

01 Abstraksi

Abstraksi

- Abstraksi biasa disebut juga sebagai Generalisasi.
- **Class abstrak tidak bisa menghasilkan objek**
- Class abstrak akan memaksa class objek yang menjadi subclass dari nya harus mengimplementasikan method yang terdapat di dalam class abstrak.



Implementasi abstract class pada kelas Hero.

Untuk mengimplementasikan kelas abstrak, kita harus meng-import module abc (abstract base class)

Pemrograman Berorientasi Objek

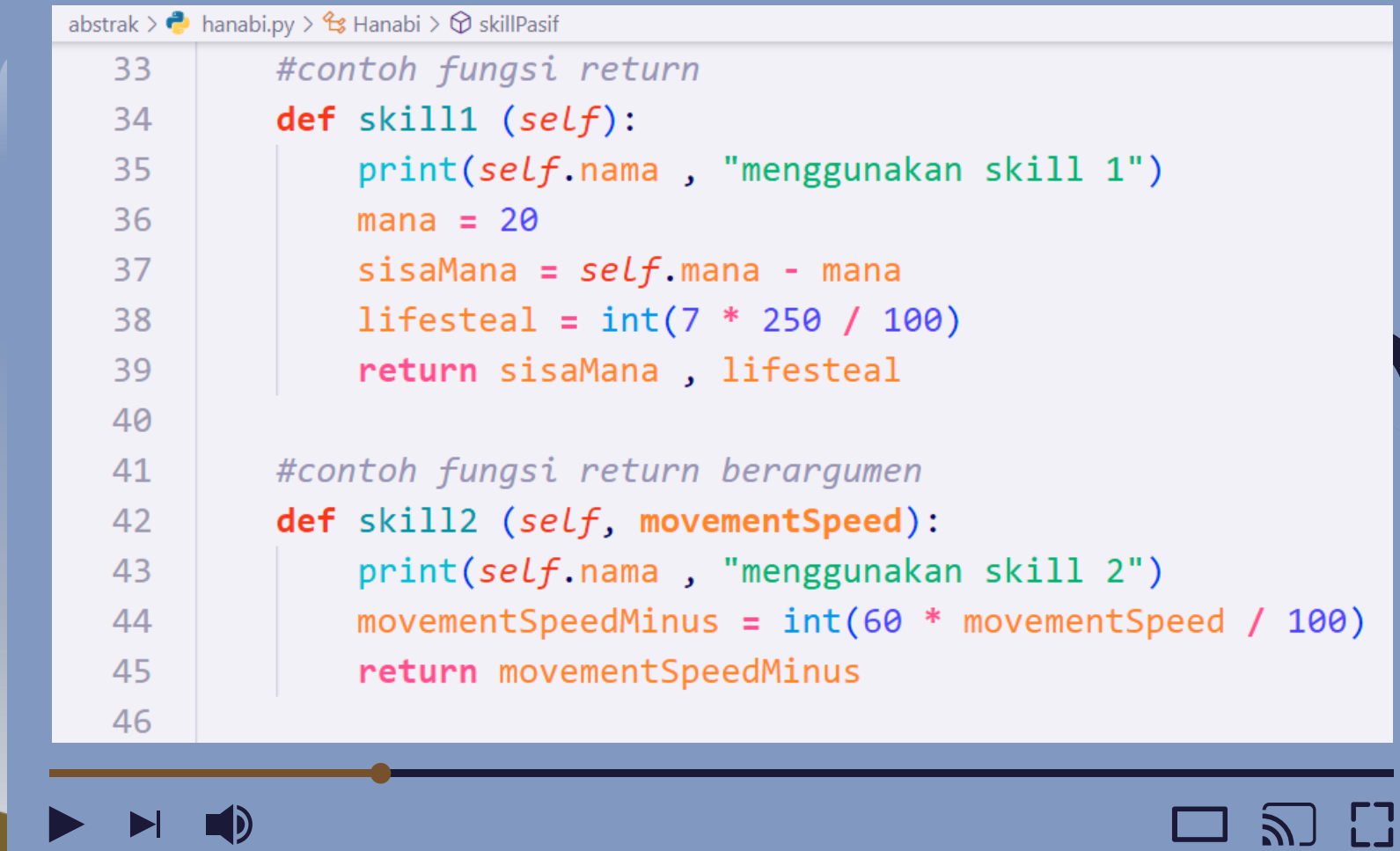
abstrak > hero.py > ...

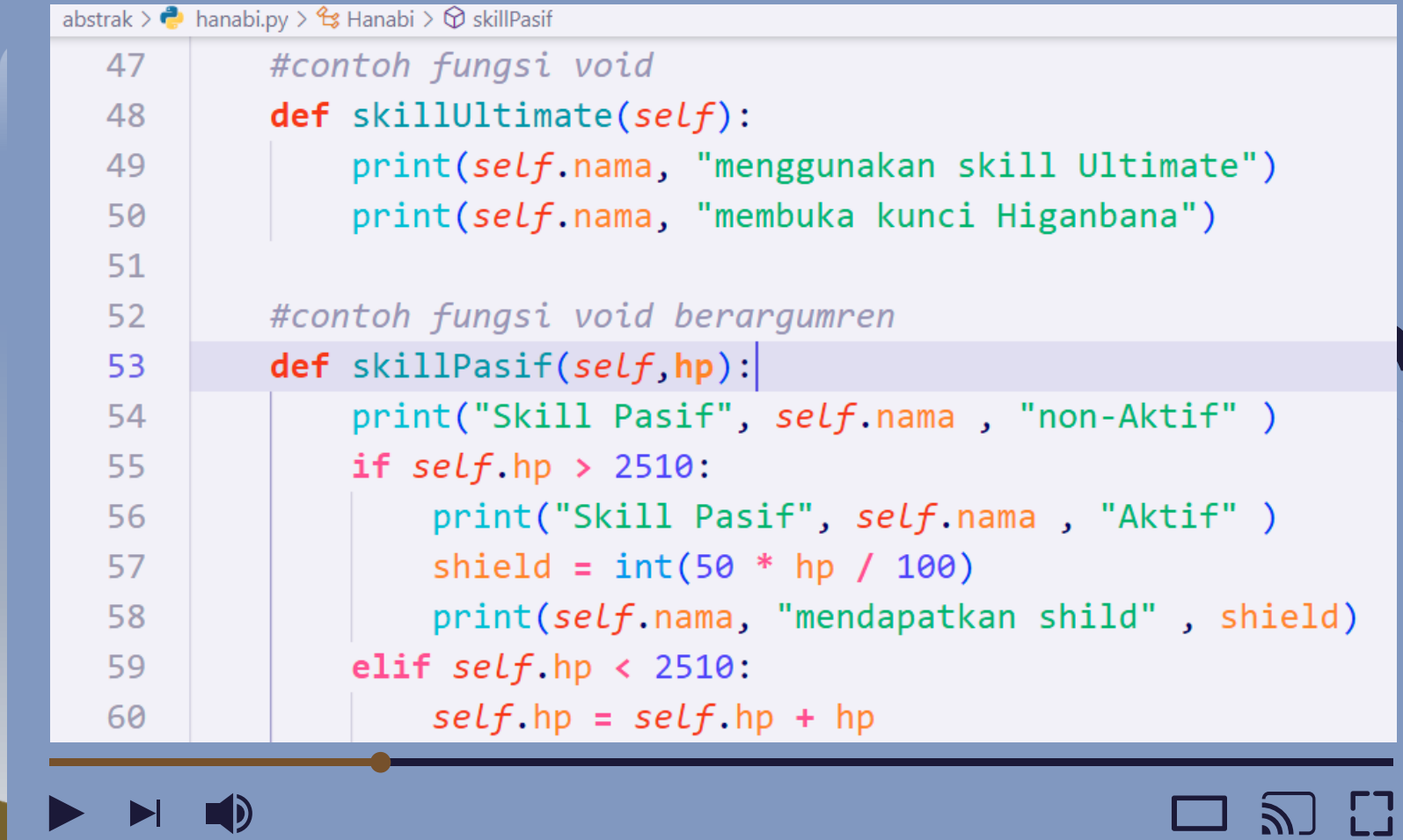
```python
from abc import ABC , abstractclassmethod

#membuat prototype Hero
class Hero(ABC):

    #atribut kelas
    jumlahHero = 0 #inisialisasi jumlah hero

    @abstractclassmethod
    def beliItem(self,item):
        print ("Fungsi membeli item",item.nama)

    @abstractclassmethod
    def farming(self,jungle):
        print("Fungsi farming")
```

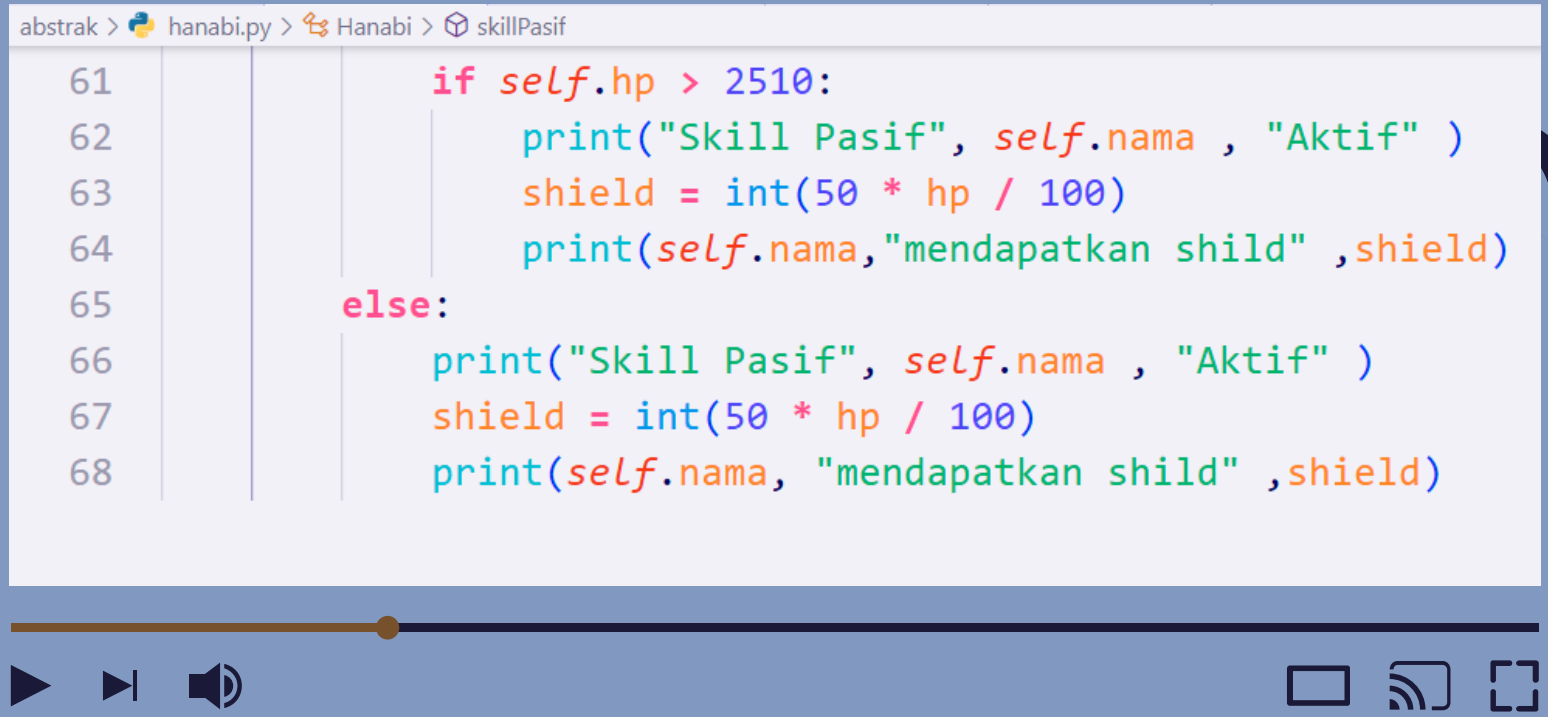
abstrak > hanabi.py > Hanabi

```python
from hero import Hero

class Hanabi(Hero):
    #magic function konstruktor
    def __init__ (self, n, r, h, m, ms,pa,ma):
        self.nama = n
        self.role = r
        self.hp = int(h)
        self.mana = int(m)
        self.movSpeed = int(ms)
        self.phyAtt = int(pa)
        self.maAtt = int(ma)
        self.item5 = []
        self.gold = 0
        Hero.jumlahHero += 1
```

```python
    def beliItem(self,item):
        print ("Fungsi membeli item",item.nama)
        if (self.gold >= item.harga):
            print(self.nama,"Membeli item",item.nama)
            self.item5.append(item)
            print(self.item5[0].nama,"sudah masuk slot")
        else:
            print(self.nama, "tidak memiliki cukup gold")

    def farming(self,jungle):
        print(self.nama, "farming")
        jungle.hp -= self.phyAtt
        if jungle.hp <= 0:
            jungle.mati()
            self.gold += jungle.goldReward

```

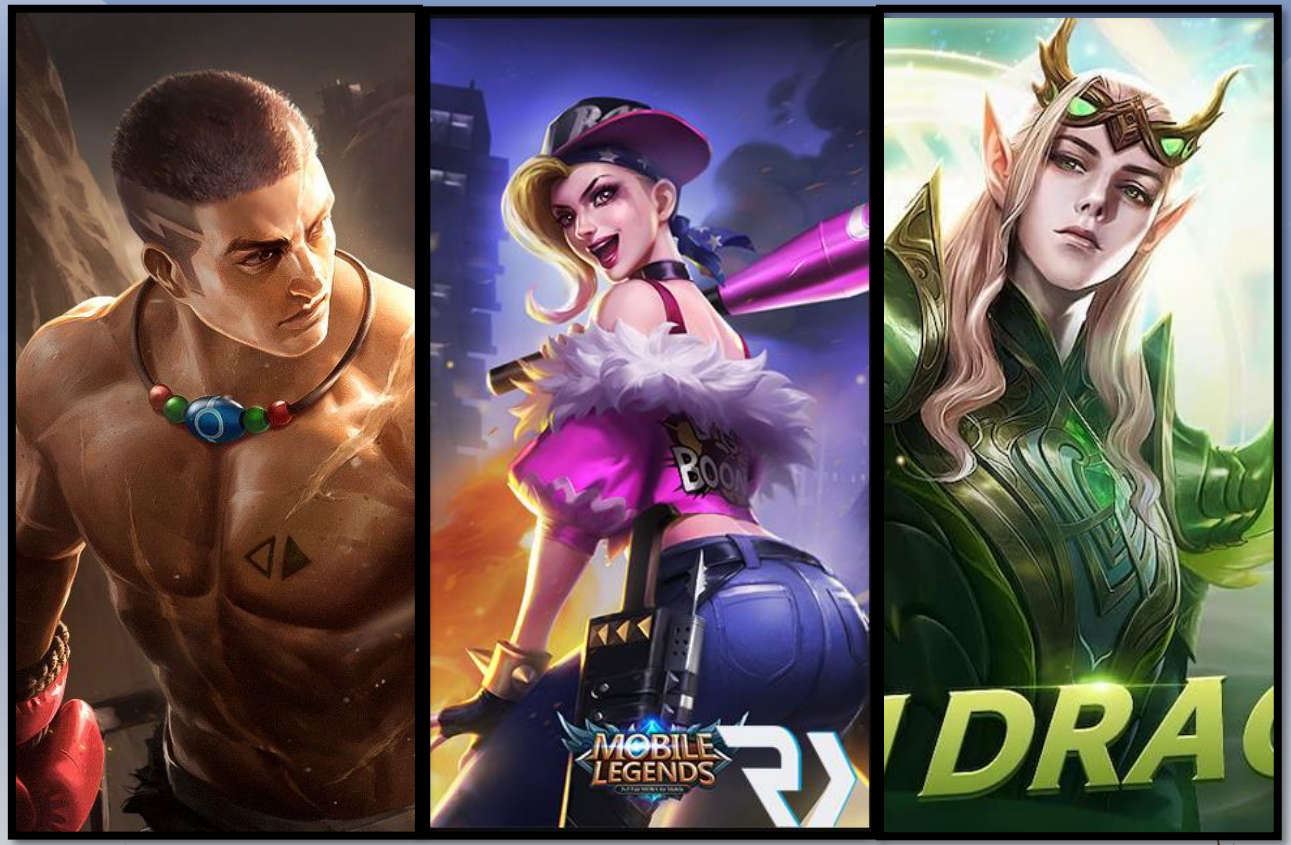

abstrak > hanabi.py > Hanabi > skillPasif
33 #contoh fungsi return
34 def skill1 (self):
35 print(self.nama , "menggunakan skill 1")
36 mana = 20
37 sisaMana = self.mana - mana
38 lifesteal = int(7 * 250 / 100)
39 return sisaMana , lifesteal
40
41 #contoh fungsi return berargumen
42 def skill2 (self, movementSpeed):
43 print(self.nama , "menggunakan skill 2")
44 movementSpeedMinus = int(60 * movementSpeed / 100)
45 return movementSpeedMinus
46

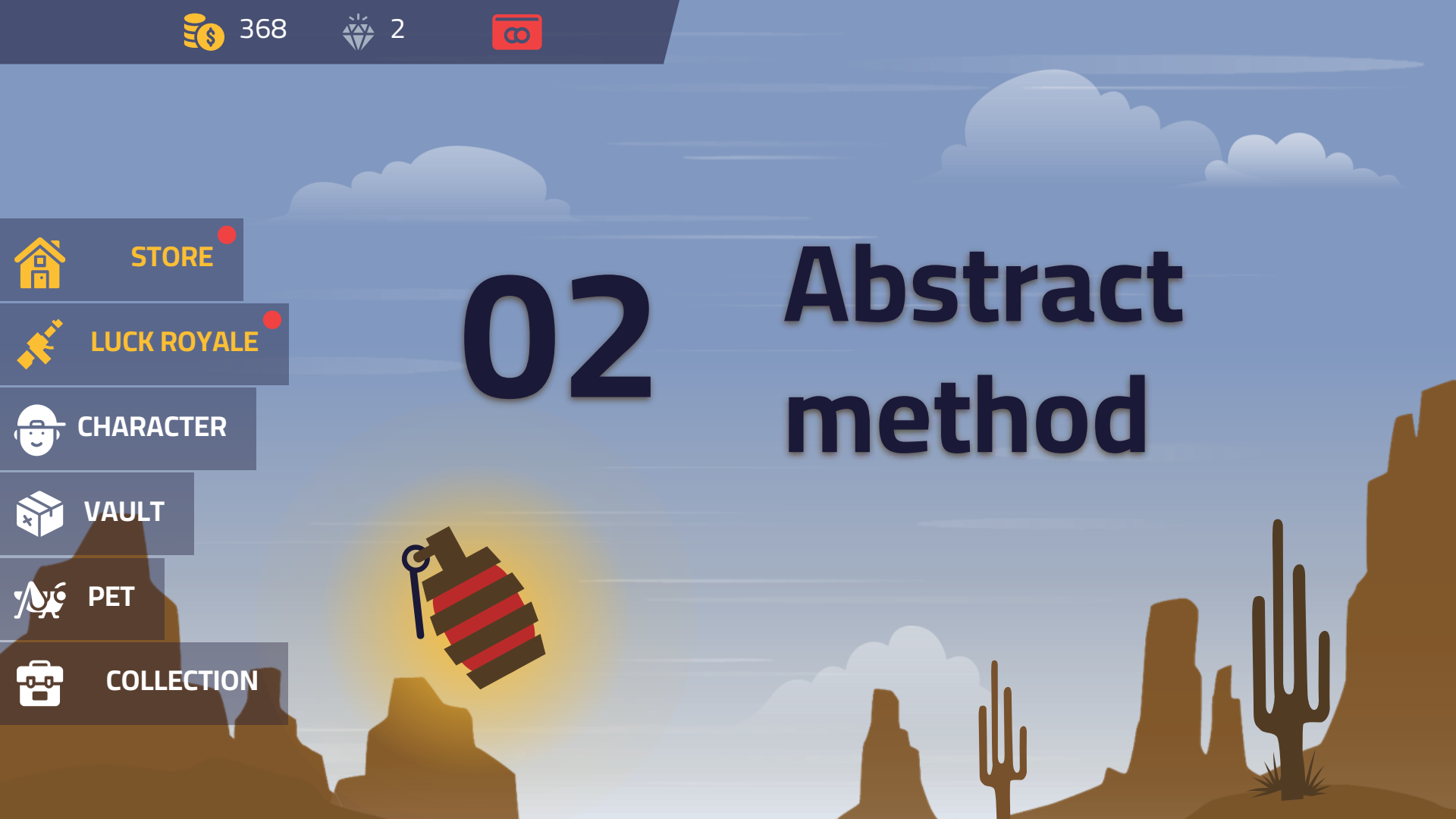

abstrak > hanabi.py > Hanabi > skillPasif
47 #contoh fungsi void
48 def skillUltimate(self):
49 print(self.nama, "menggunakan skill Ultimate")
50 print(self.nama, "membuka kunci Higanbana")
51
52 #contoh fungsi void berargumren
53 def skillPasif(self,hp):
54 print("Skill Pasif", self.nama , "non-Aktif")
55 if self.hp > 2510:
56 print("Skill Pasif", self.nama , "Aktif")
57 shield = int(50 * hp / 100)
58 print(self.nama, "mendapatkan shild" , shield)
59 elif self.hp < 2510:
60 self.hp = self.hp + hp


abstrak > hanabi.py > Hanabi > skillPasif
61 if self.hp > 2510:
62 print("Skill Pasif", self.nama , "Aktif")
63 shield = int(50 * hp / 100)
64 print(self.nama,"mendapatkan shild" ,shield)
65 else:
66 print("Skill Pasif", self.nama , "Aktif")
67 shield = int(50 * hp / 100)
68 print(self.nama, "mendapatkan shild" ,shield)

abstrak > main.py > ...

```python
from hanabi import Hanabi

hanabi = Hanabi ("Hanabi","MM",100,100,100,100,100)

print(hanabi)
print(hanabi.nama)
print(hanabi.skill1())
```


MOBILE
LEGENDS

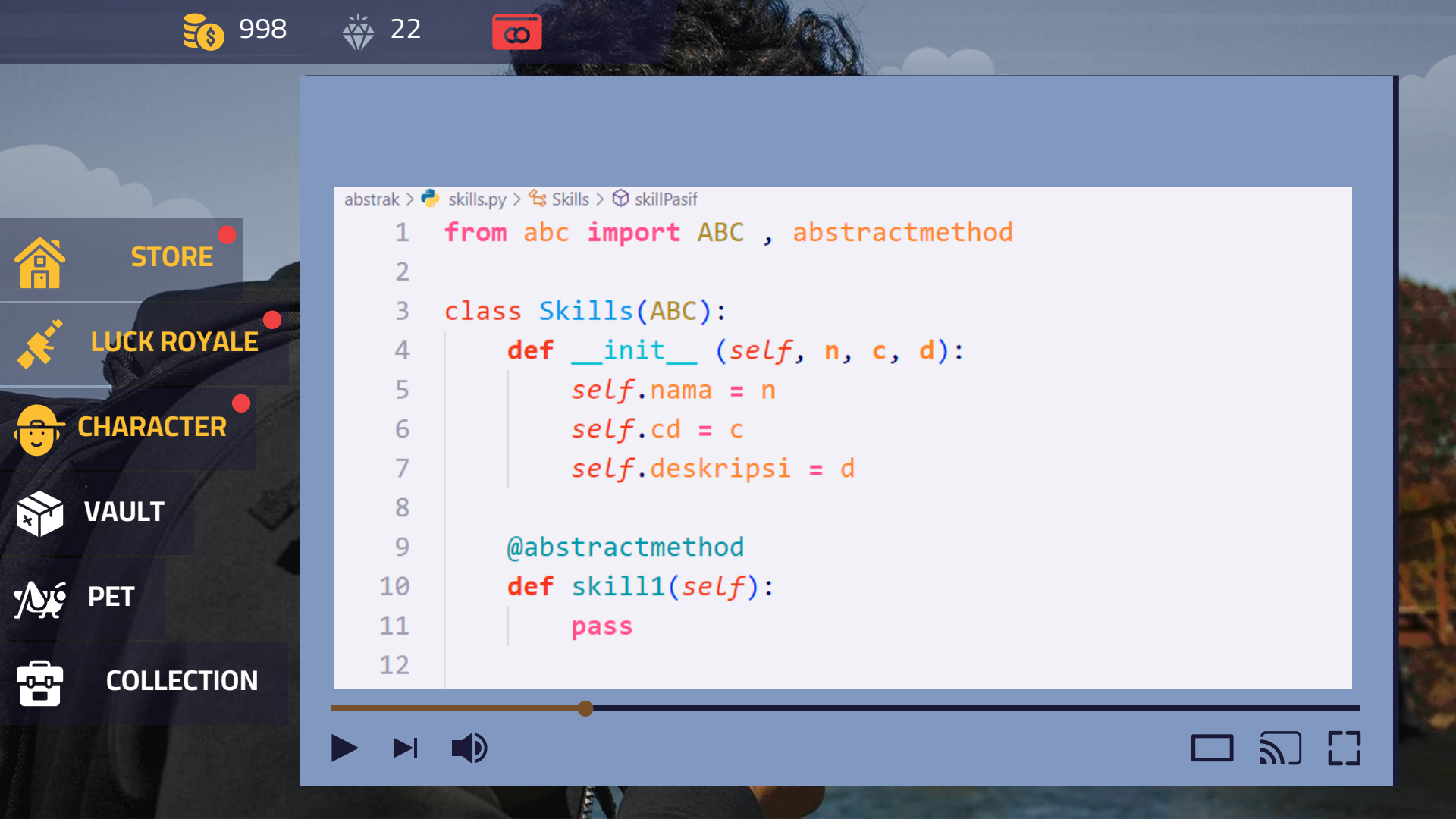


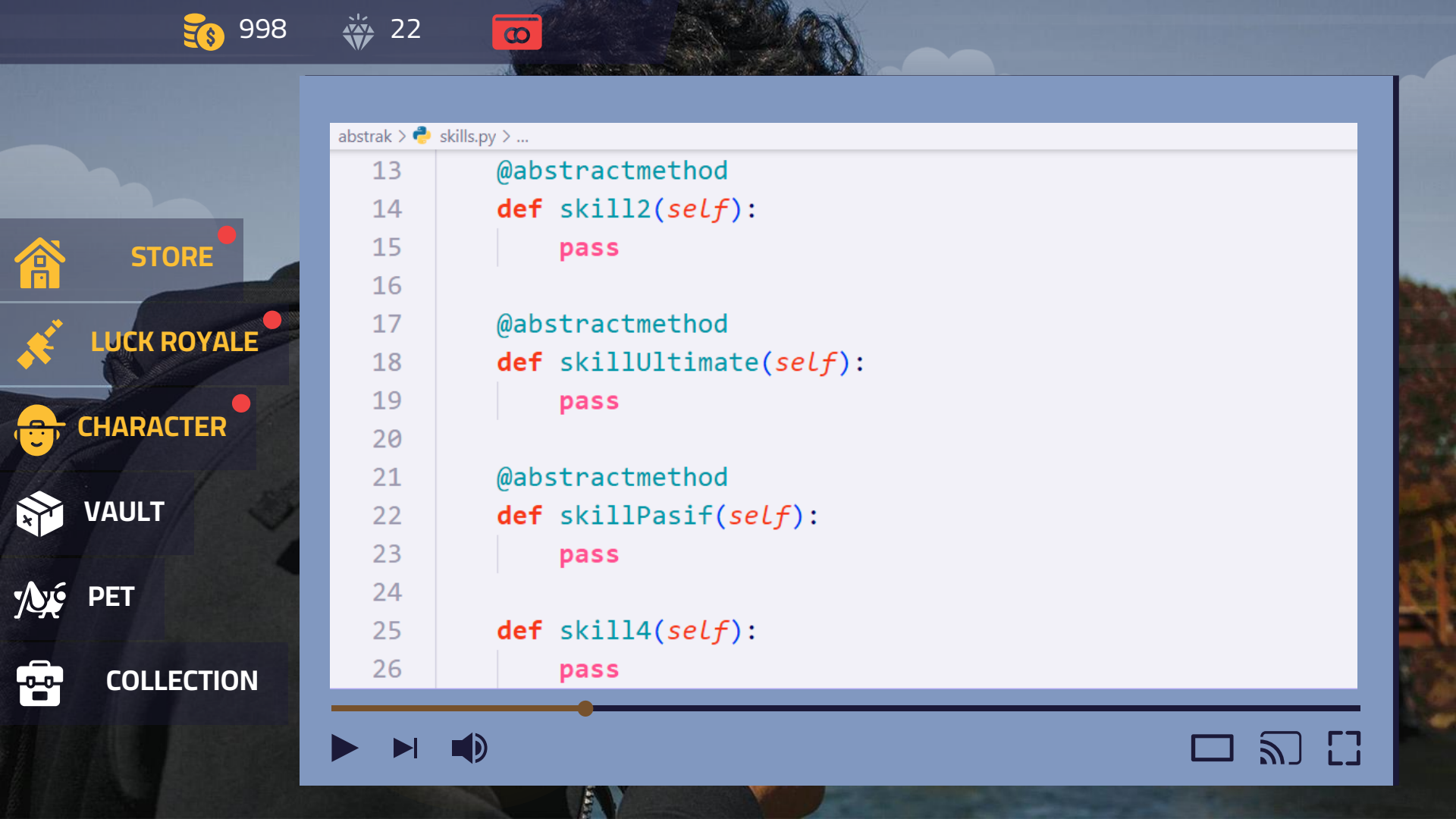
02 Abstract method

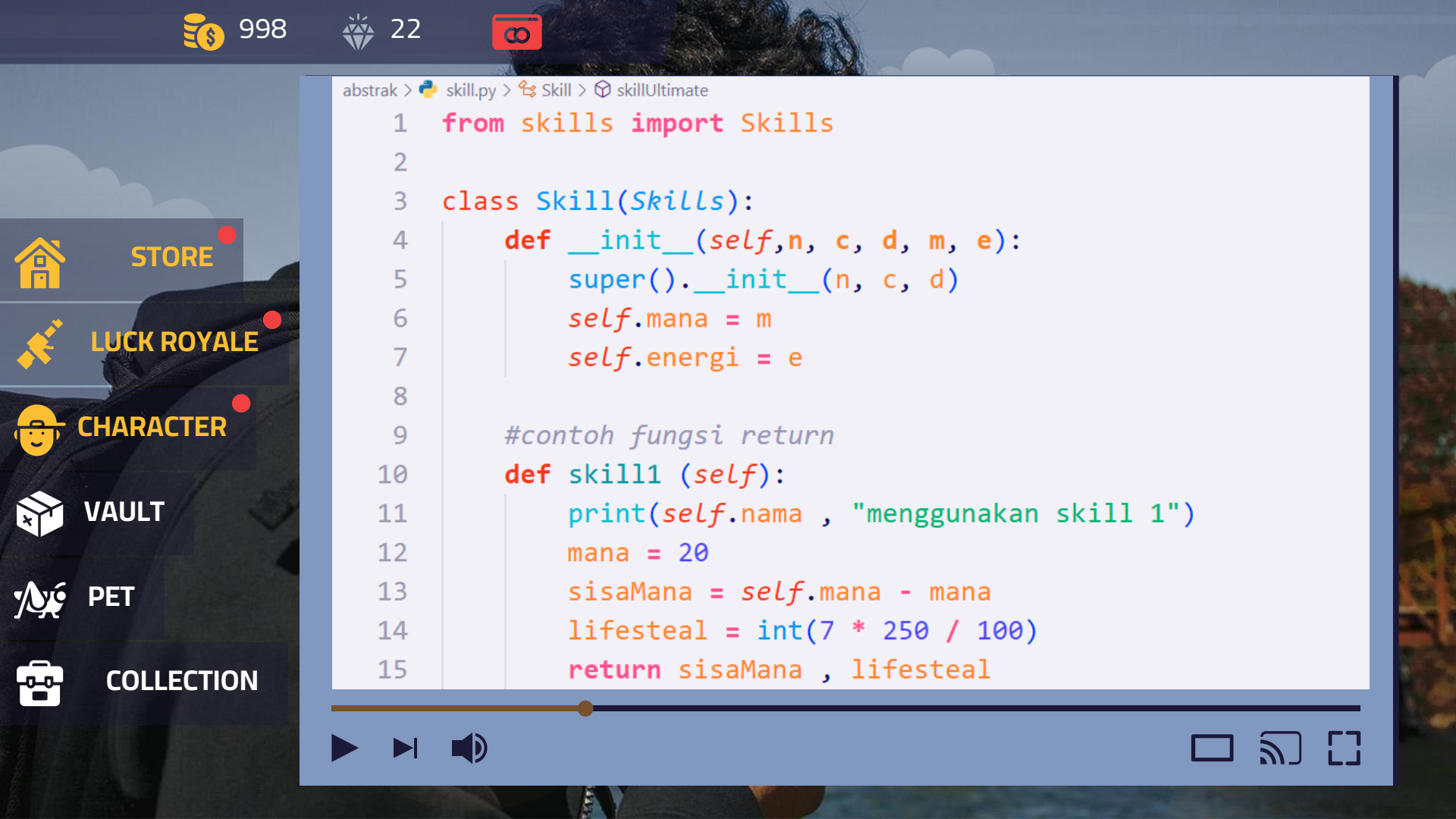

MOBILE
LEGENDS
BANG BANG

Abstract method

- Abstract Method merupakan sebuah method dalam abstract class yang wajib di miliki oleh sub class nya.
- Untuk fungsi yang harus di implementasikan oleh sub class nya juga harus menggunakan decorator @abstractclassmethod
- Syntaks decorator juga bisa menggunakan @abstractmethod

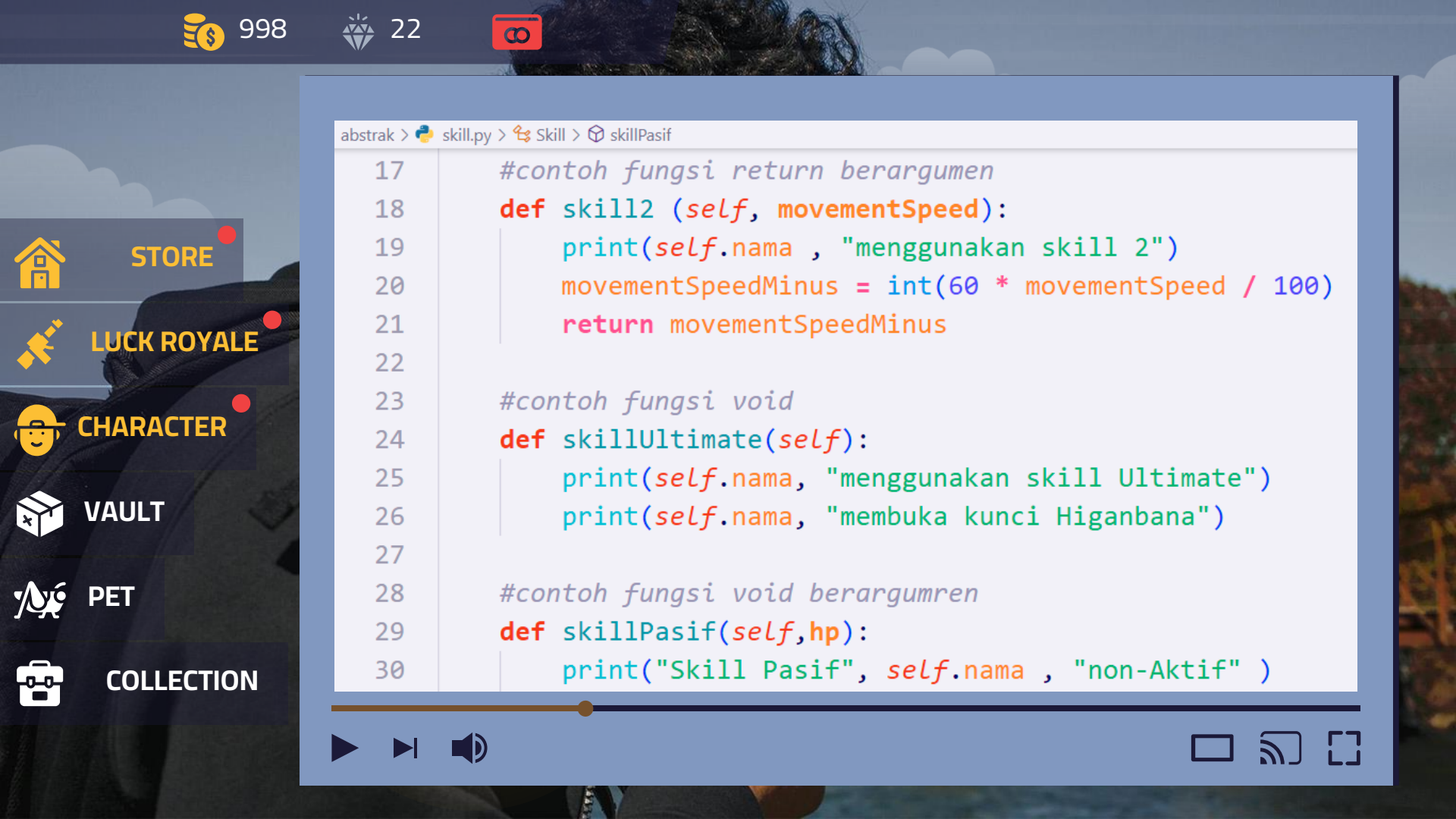

998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
abstrak > skills.py > Skills > skillPasif
1 from abc import ABC , abstractmethod
2
3 class Skills(ABC):
4 def __init__ (self, n, c, d):
5 self.nama = n
6 self.cd = c
7 self.deskripsi = d
8
9 @abstractmethod
10 def skill1(self):
11 pass
12


998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
abstrak > skills.py > ...
13 @abstractmethod
14 def skill2(self):
15 pass
16
17 @abstractmethod
18 def skillUltimate(self):
19 pass
20
21 @abstractmethod
22 def skillPasif(self):
23 pass
24
25 def skill4(self):
26 pass



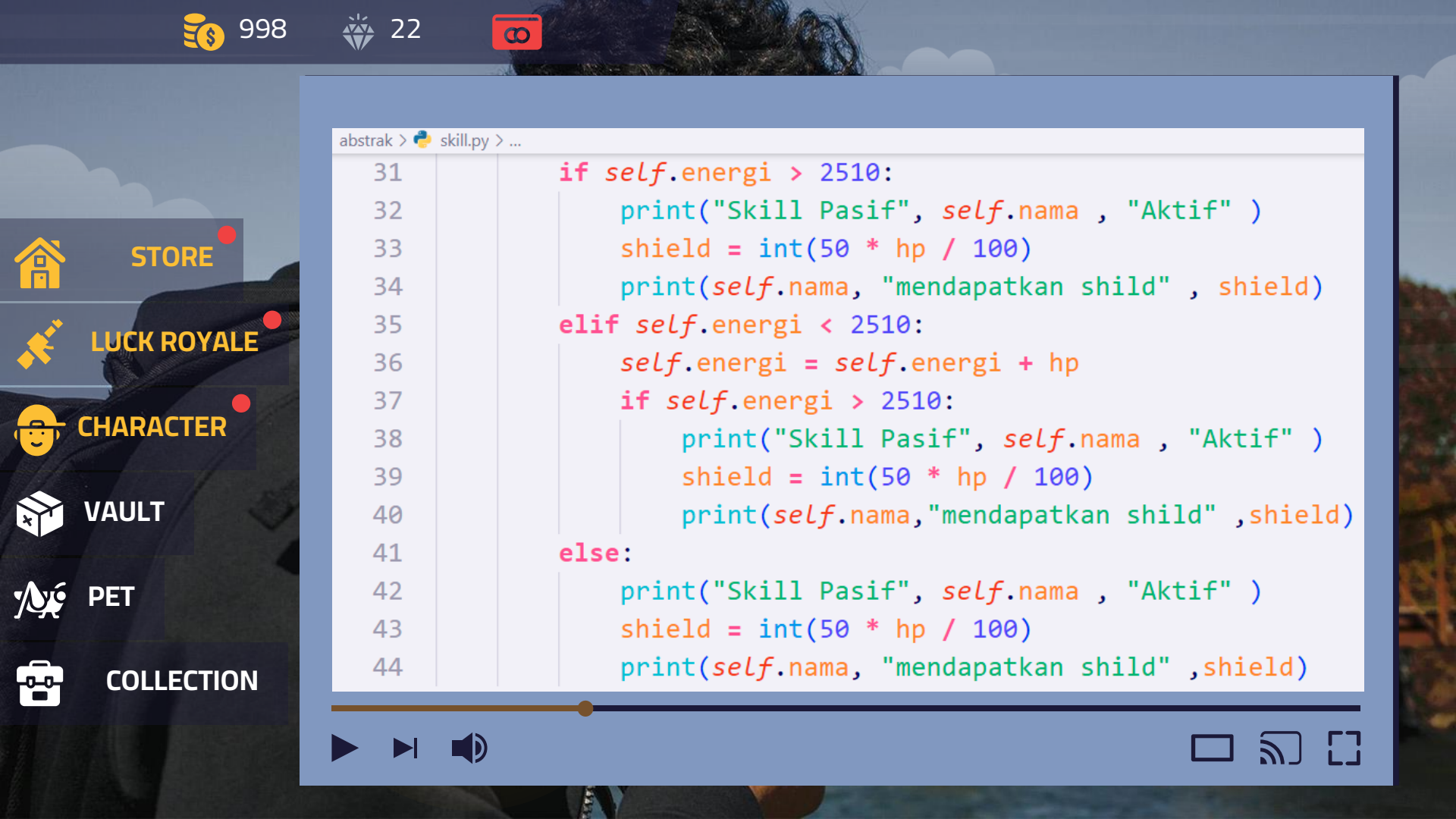
abstrak > skill.py > Skill > skillUltimate

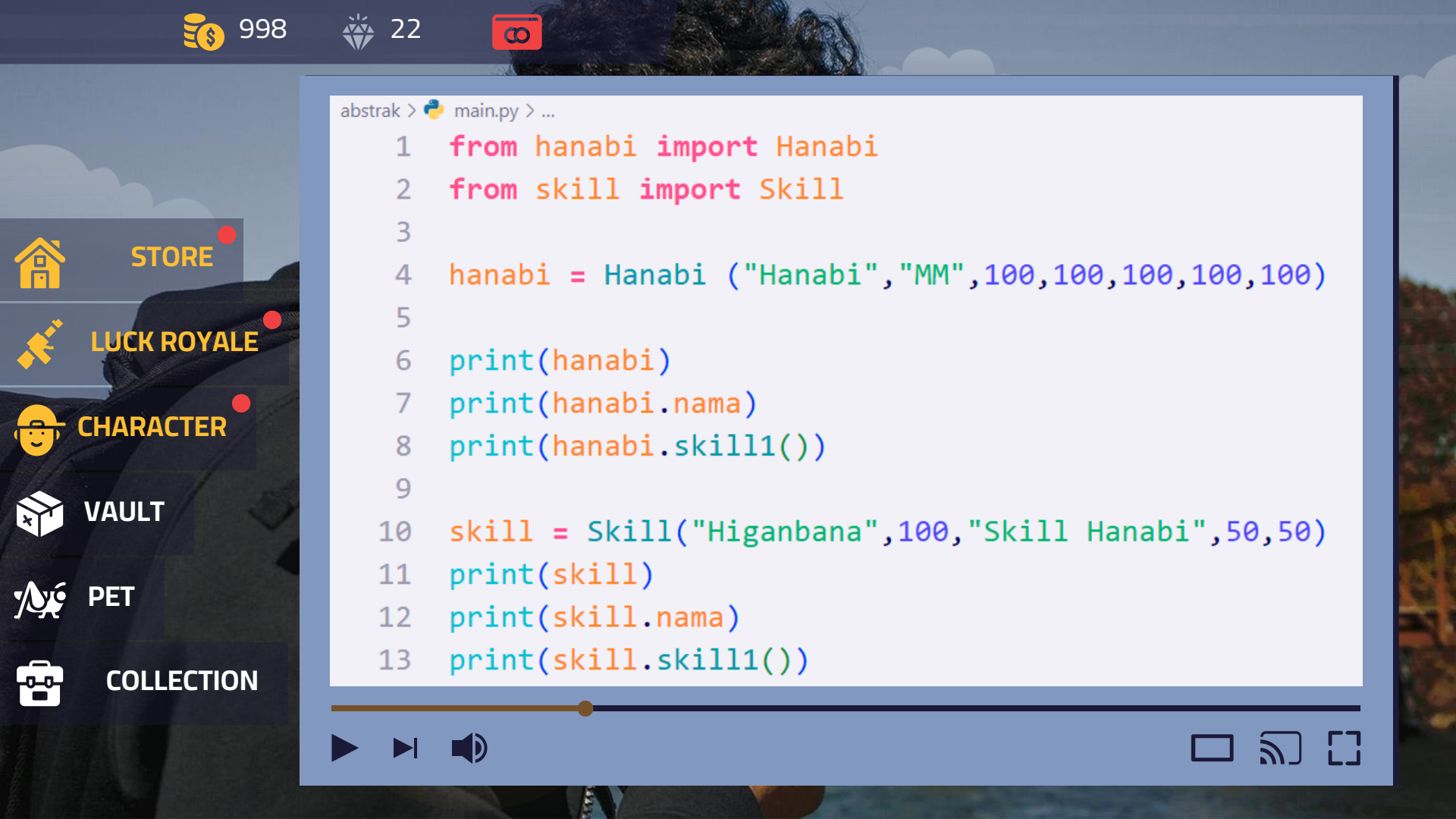
```python
from skills import Skills

class Skill(Skills):
    def __init__(self,n, c, d, m, e):
        super().__init__(n, c, d)
        self.mana = m
        self.energi = e

    #contoh fungsi return
    def skill1 (self):
        print(self.nama , "menggunakan skill 1")
        mana = 20
        sisaMana = self.mana - mana
        lifesteal = int(7 * 250 / 100)
        return sisaMana , lifesteal
```



abstrak > skill.py > Skill > skillPasif

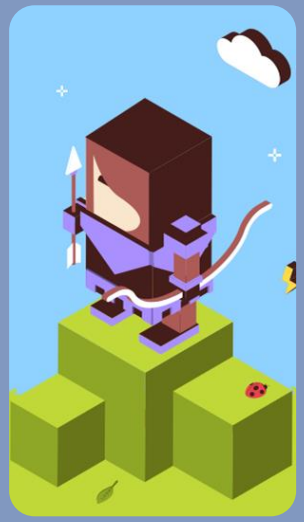
```python
    #contoh fungsi return berargumen
    def skill2 (self, movementSpeed):
        print(self.nama , "menggunakan skill 2")
        movementSpeedMinus = int(60 * movementSpeed / 100)
        return movementSpeedMinus

    #contoh fungsi void
    def skillUltimate(self):
        print(self.nama, "menggunakan skill Ultimate")
        print(self.nama, "membuka kunci Higanbana")

    #contoh fungsi void berargumren
    def skillPasif(self,hp):
        print("Skill Pasif", self.nama , "non-Aktif" )
```


998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
abstrak > skill.py > ...
31 if self.energi > 2510:
32 print("Skill Pasif", self.nama , "Aktif")
33 shield = int(50 * hp / 100)
34 print(self.nama, "mendapatkan shild" , shield)
35 elif self.energi < 2510:
36 self.energi = self.energi + hp
37 if self.energi > 2510:
38 print("Skill Pasif", self.nama , "Aktif")
39 shield = int(50 * hp / 100)
40 print(self.nama,"mendapatkan shild" ,shield)
41 else:
42 print("Skill Pasif", self.nama , "Aktif")
43 shield = int(50 * hp / 100)
44 print(self.nama, "mendapatkan shild" ,shield)



abstrak > main.py > ...

```python
from hanabi import Hanabi
from skill import Skill

hanabi = Hanabi ("Hanabi","MM",100,100,100,100,100)

print(hanabi)
print(hanabi.nama)
print(hanabi.skill1())

skill = Skill("Higanbana",100,"Skill Hanabi",50,50)
print(skill)
print(skill.nama)
print(skill.skill1())
```



Ingat!

Fungsi Super digunakan untuk memanggil fungsi konstruktor pada kelas induk dari sub class nya.


MOBILE LEGENDS
BANG BANG
DRAGON TAMER
ESTES
"RATTAN DRAGON"

368 2

STORE

LUCK ROYALE

CHARACTER

VAULT

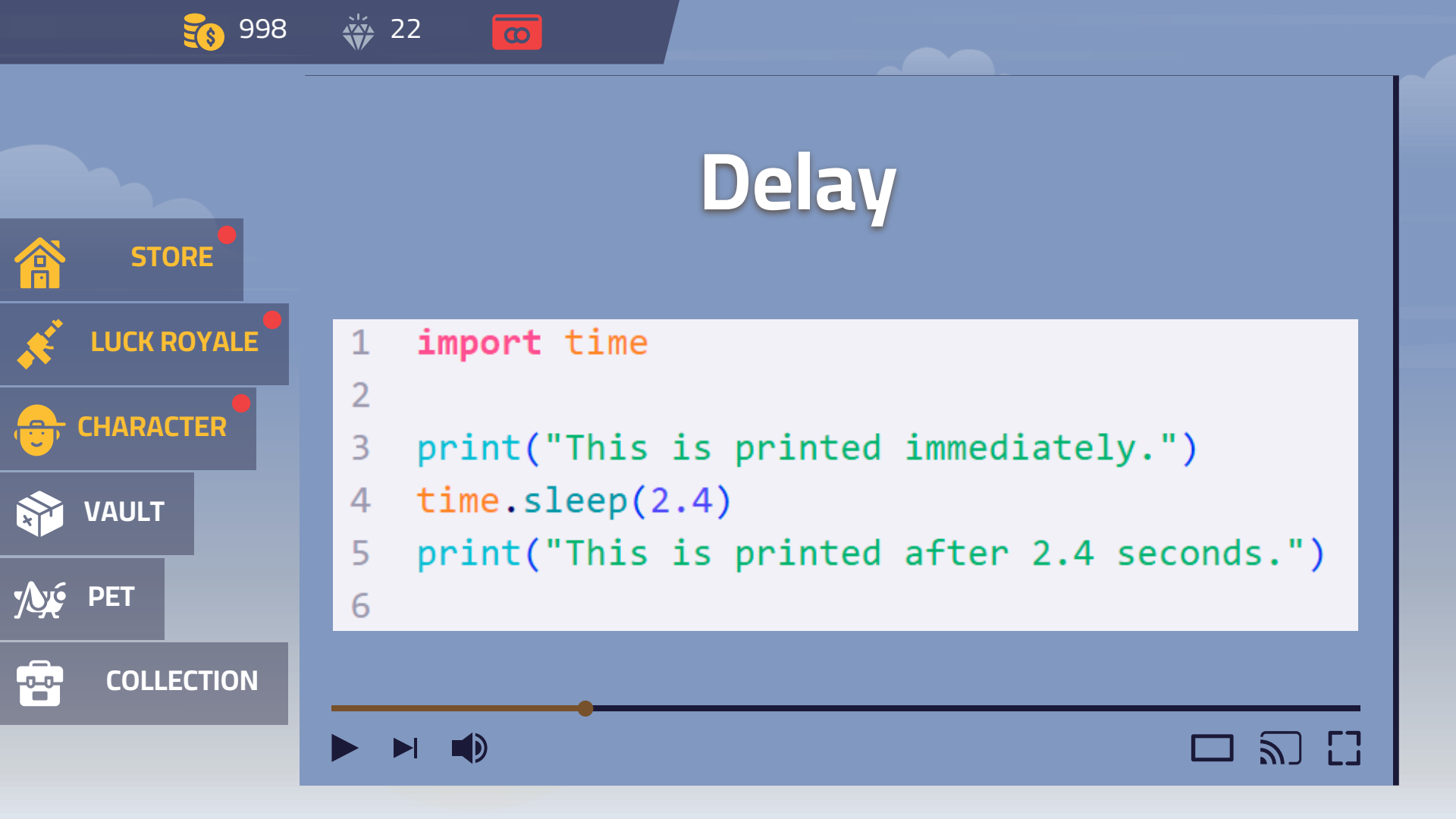
PET

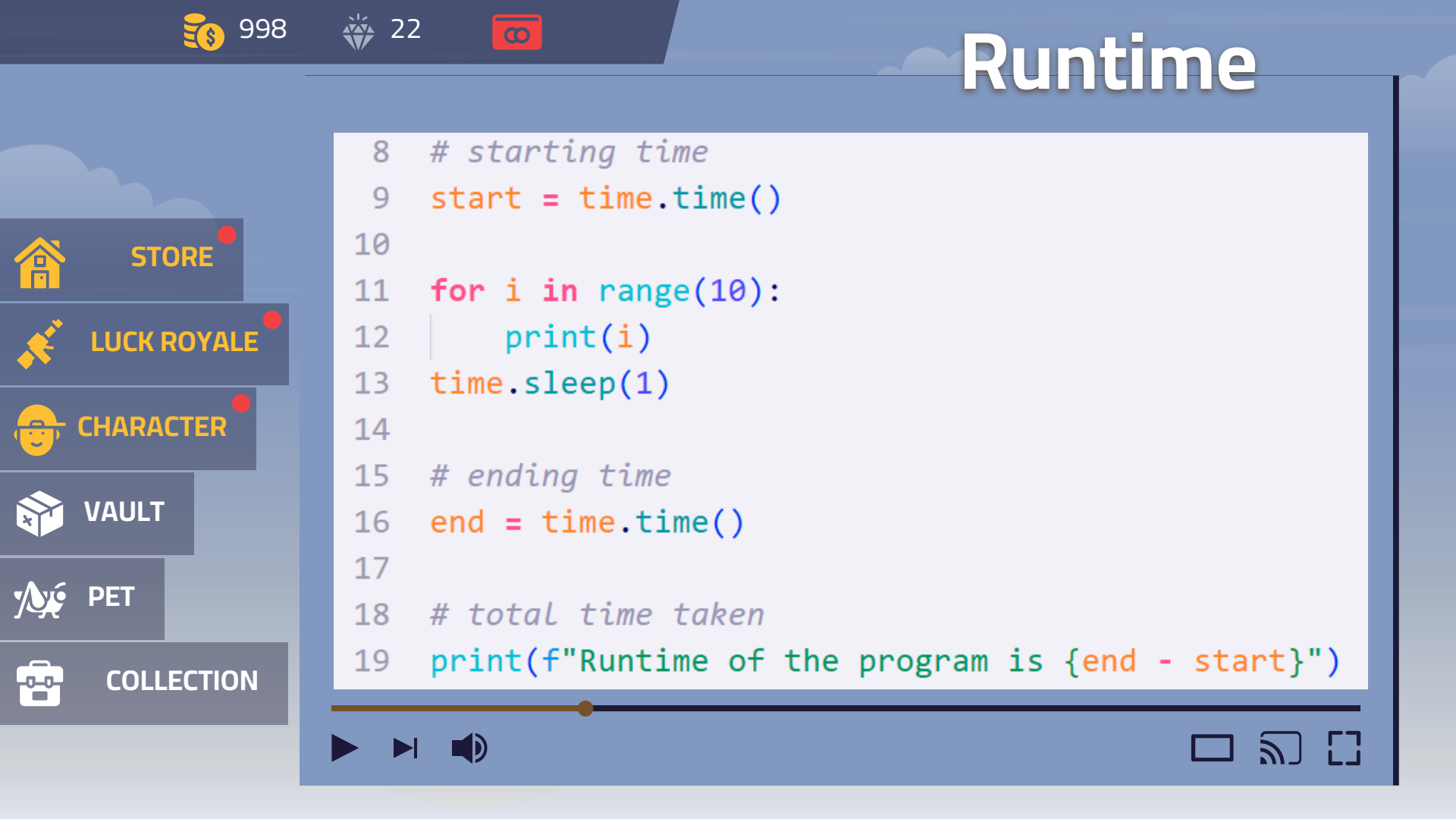
COLLECTION

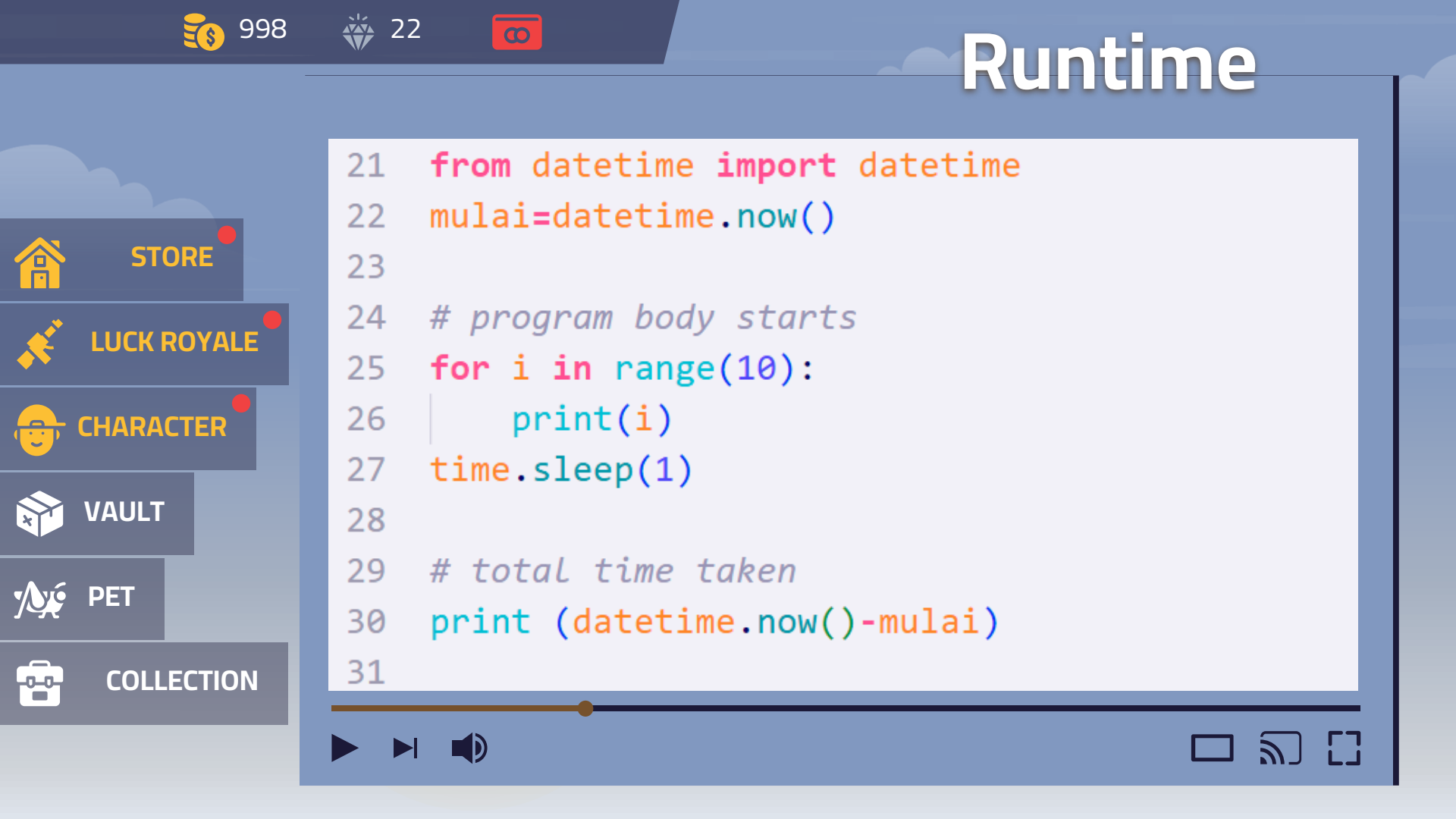
03 Time

Time

- Module Time digunakan untuk program yang memiliki fungsi waktu.
- Beberapa fungsi seperti runtime, delay, countdown, dan fungsi lainnya dapat diimplementasikan.
- Module time ini juga berkaitan erat dengan modul datetime.


998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
Delay
1 import time
2
3 print("This is printed immediately.")
4 time.sleep(2.4)
5 print("This is printed after 2.4 seconds.")
6


998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
Runtime
# starting time
start = time.time()
for i in range(10):
print(i)
time.sleep(1)
# ending time
end = time.time()
# total time taken
print(f"Runtime of the program is {end - start}")


998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
Runtime
21 from datetime import datetime
22 mulai=datetime.now()
23
24 # program body starts
25 for i in range(10):
26 print(i)
27 time.sleep(1)
28
29 # total time taken
30 print (datetime.now()-mulai)
31


998
22
Date-time
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
1 import datetime
2
3 x = datetime.datetime(2018, 6, 1)
4 print(x)
5 print(x.strftime("%B"))
6
7 y = datetime.datetime.now()
8 print(y)
9 print(y.year)
10 print(y.strftime("%A"))
11

strftime

```
1  import time
2
3  named_tuple = time.localtime()
4  time_string = time.strftime("%m/%d/%Y, %H:%M:%S", named_tuple)
5  get_sec = time.strftime("%S", named_tuple)
6
7  print(time_string)
8  print(get_sec)
9
```

STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION

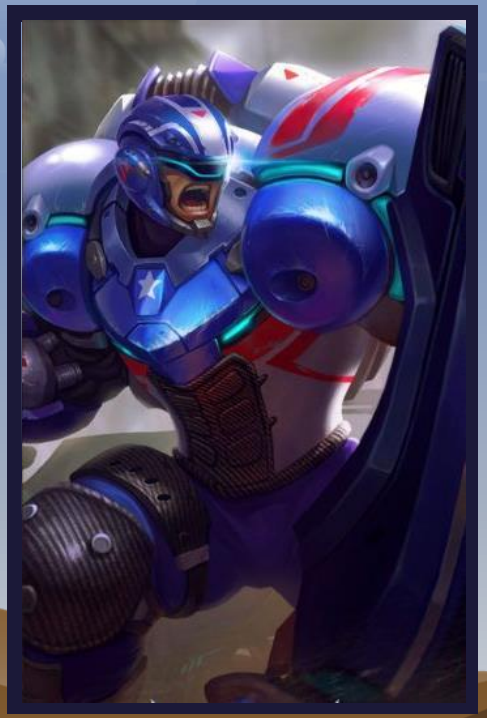
Directive	Description	Example
%a	Weekday, short version	Wed
%A	Weekday, full version	Wednesday
%w	Weekday as a number 0-6, 0 is Sunday	3
%d	Day of month 01-31	31
%b	Month name, short version	Dec
%B	Month name, full version	December
%m	Month as a number 01-12	12
%y	Year, short version, without century	18
%Y	Year, full version	2018

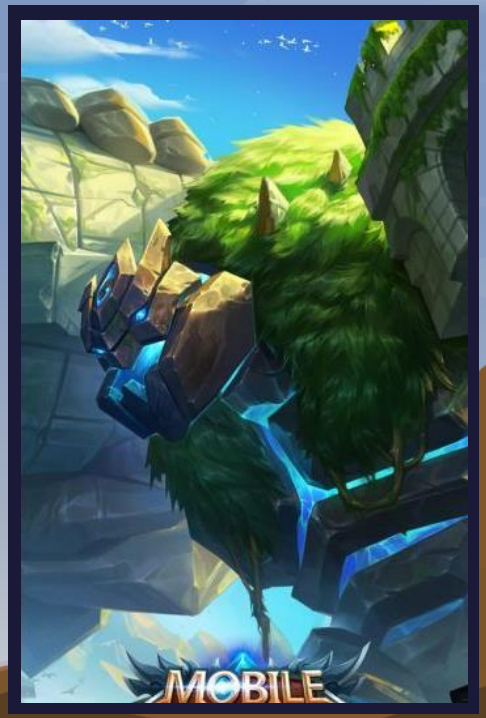
%H	Hour 00-23	17
%I	Hour 00-12	05
%p	AM/PM	PM
%M	Minute 00-59	41
%S	Second 00-59	08
%f	Microsecond 000000-999999	548513
%z	UTC offset	+0100
%Z	Timezone	CST
%j	Day number of year 001-366	365

STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION

%U	Week number of year, Sunday as the first day of week, 00-53	52
%W	Week number of year, Monday as the first day of week, 00-53	52
%c	Local version of date and time	Mon Dec 31 17:41:00 2018
%C	Century	20
%x	Local version of date	12/31/18
%X	Local version of time	17:41:00
%%	A % character	%
%G	ISO 8601 year	2018
%u	ISO 8601 weekday (1-7)	1
%V	ISO 8601 weeknumber (01-53)	01

Timer





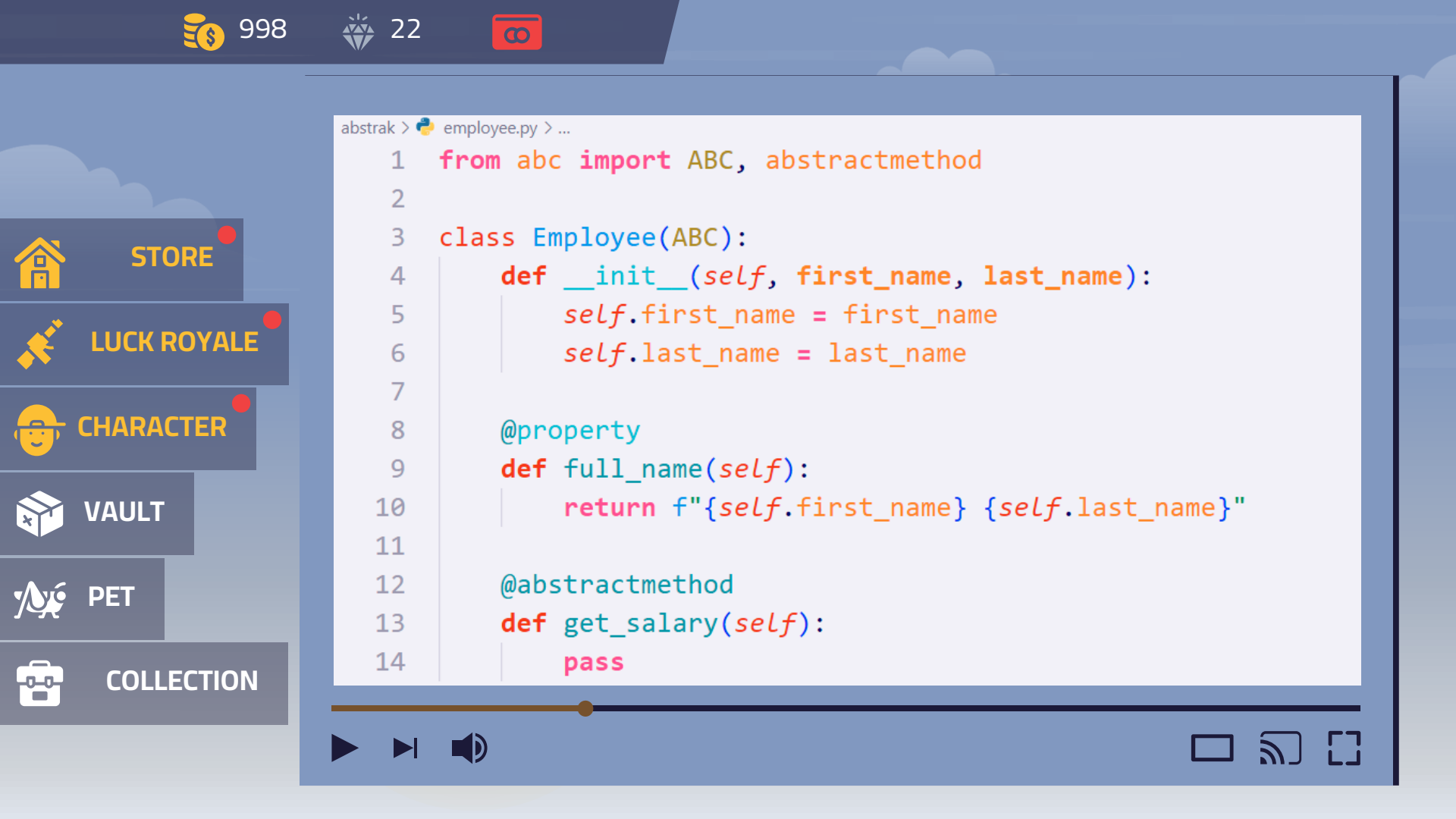

998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
time_cd.py > ...
countdown
1 import time
2
3 def countdown(t):
4 while t:
5 mins, secs = divmod(t, 60)
6 timer = '{:02d}:{:02d}'.format(mins, secs)
7 print(timer, end="\r")
8 time.sleep(1)
9 t -= 1
10 print('Game Selesai!!')
11
12 t = input("Masukkan waktu mundur dalam detik: ")
13
14 countdown(int(t))

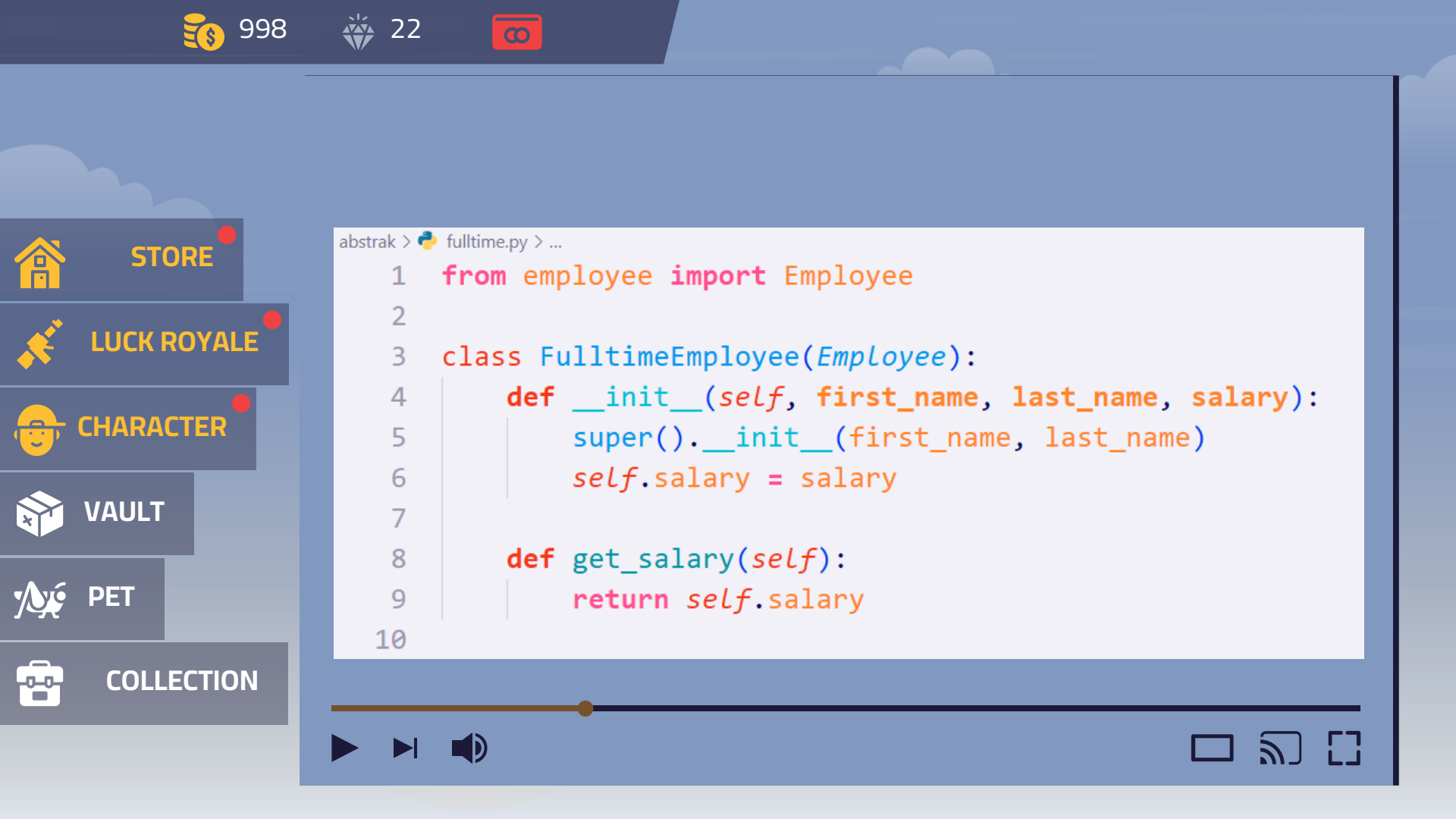

998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
time_cu.py > ...
countup
1 import time
2
3 def countup(t):
4 t_start = 0
5 while t_start < t:
6 mins, secs = divmod(t_start, 60)
7 timer = '{:02d}:{:02d}'.format(mins, secs)
8 print(timer, end="\r")
9 time.sleep(1)
10 t_start += 1
11 print('Game Selesai!!')
12
13 t = input("Masukkan waktu mundur dalam detik: ")
14
15 countup(int(t))

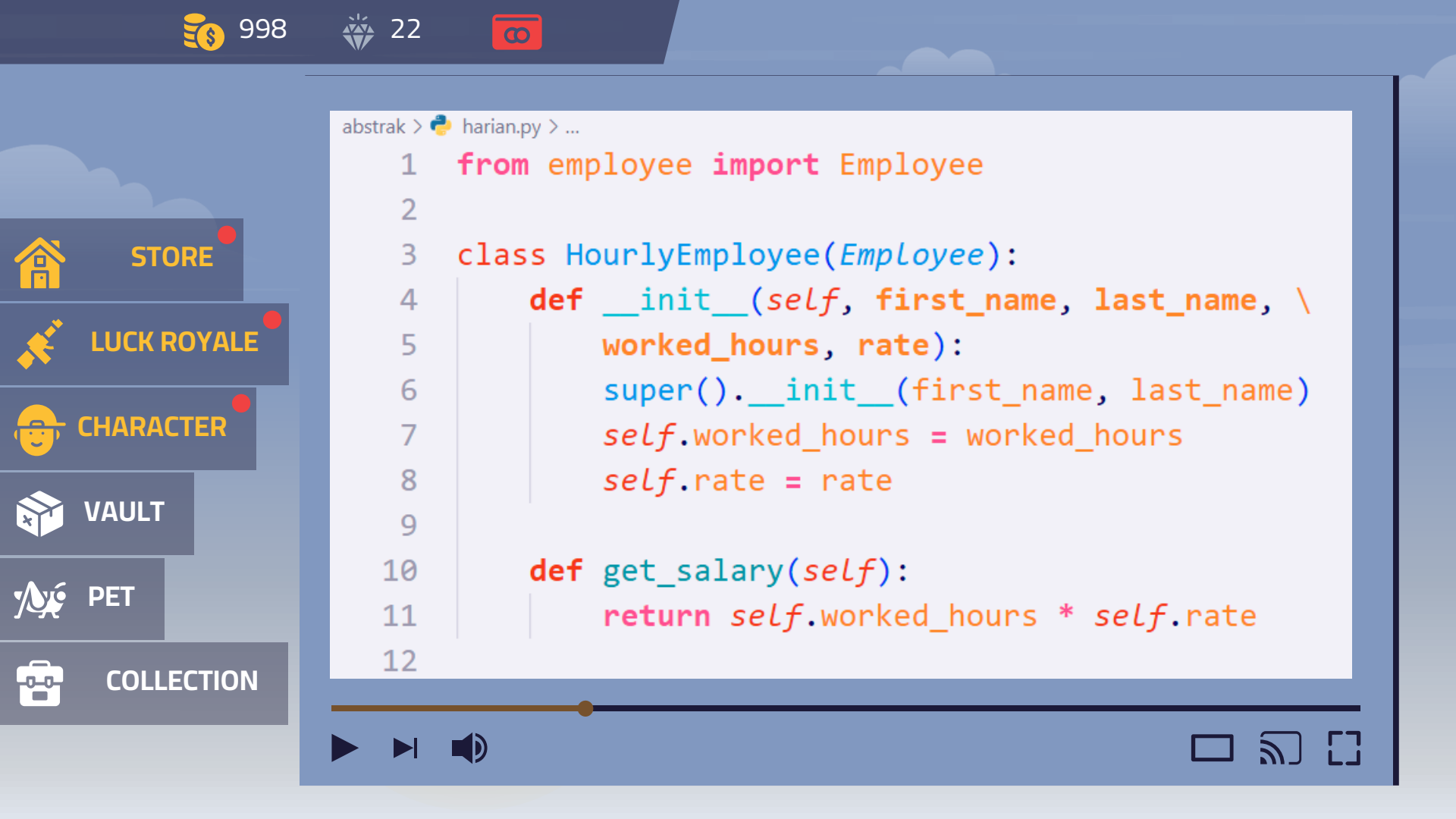

368
2
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
04 Studi Kasus

Payroll Karyawan

- Abstrak kelas akan sangat berfungsi pada kasus yang masing-masing kelas-nya memiliki keterkaitan yang erat.
- Misalkan kita ingin membuat aplikasi payroll yang mengatur honor karyawan, baik karyawan tetap maupun karyawan harian.

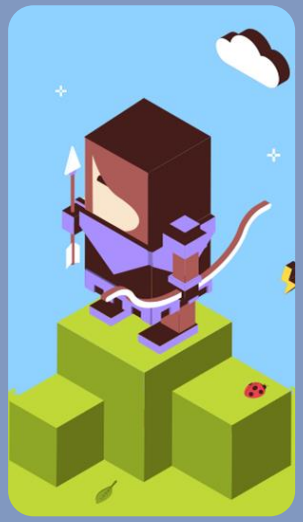

998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
abstrak > employee.py > ...
1 from abc import ABC, abstractmethod
2
3 class Employee(ABC):
4 def __init__(self, first_name, last_name):
5 self.first_name = first_name
6 self.last_name = last_name
7
8 @property
9 def full_name(self):
10 return f"{self.first_name} {self.last_name}"
11
12 @abstractmethod
13 def get_salary(self):
14 pass


998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
abstrak > fulltime.py > ...
1 from employee import Employee
2
3 class FulltimeEmployee(Employee):
4 def __init__(self, first_name, last_name, salary):
5 super().__init__(first_name, last_name)
6 self.salary = salary
7
8 def get_salary(self):
9 return self.salary
10

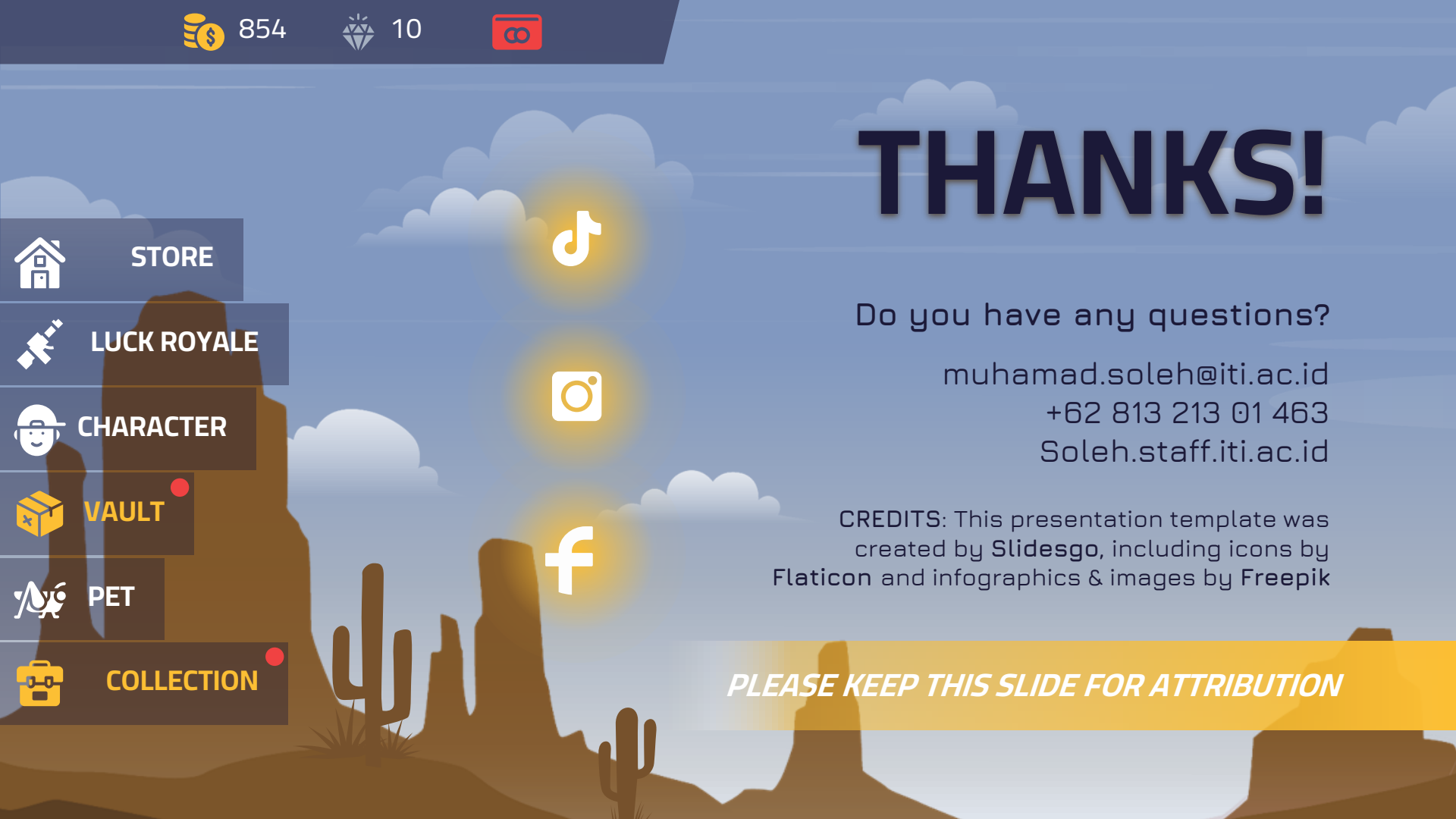

998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
abstrak > harian.py > ...
1 from employee import Employee
2
3 class HourlyEmployee(Employee):
4 def __init__(self, first_name, last_name, \
5 worked_hours, rate):
6 super().__init__(first_name, last_name)
7 self.worked_hours = worked_hours
8 self.rate = rate
9
10 def get_salary(self):
11 return self.worked_hours * self.rate
12


998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
abstrak > payroll.py > ...
1 class Payroll:
2 def __init__(self):
3 self.employee_list = []
4
5 def add(self, employee):
6 self.employee_list.append(employee)
7
8 def print(self):
9 for e in self.employee_list:
10 print(f"{e.full_name} \t ${e.get_salary()}")
11


998
22
STORE
LUCK ROYALE
CHARACTER
VAULT
PET
COLLECTION
abstrak > main_payroll.py > ...
1 from fulltime import FulltimeEmployee
2 from harian import HourlyEmployee
3 from payroll import Payroll
4
5 payroll = Payroll()
6
7 payroll.add(FulltimeEmployee('John', 'Doe', 6000))
8 payroll.add(FulltimeEmployee('Jane', 'Doe', 6500))
9 payroll.add(HourlyEmployee('Jenifer', 'Smith', 200, 50))
10 payroll.add(HourlyEmployee('David', 'Wilson', 150, 100))
11 payroll.add(HourlyEmployee('Kevin', 'Miller', 100, 150))
12
13 payroll.print()
14




Website
Home > Content > Post
PAYROLL
search



THANKS!

Do you have any questions?

muhamad.soleh@iti.ac.id
+62 813 213 01 463
Soleh.staff.iti.ac.id


slidesgo