Aplikasi Himpunan dalam Pemilihan *Hero* pada *Game* DOTA 2

Tazkia Nizami - 13522032[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13522032@mahasiswa.itb.ac.id

Abstrak*—Dalam DOTA 2, pemilihan Hero merupakan aspek penting yang memerlukan pemahaman mendalam akan keunggulan dan kelemahan setiap Hero. Penelitian ini menggunakan informasi dari forum penggemar DOTA 2 yang diekstrak dengan Beautiful Soup 4, sebuah library Python, untuk mengekstrak data dari halaman web dan membentuk himpunan nama-nama Hero yang baik atau tidak untuk dimainkan berdasarkan Counter dan Countered oleh lawan. Langkah-langkah implementasi himpunan pada program diperinci bersama dengan eksperimen yang menunjukkan keberhasilan program dalam merekomendasikan Hero yang sesuai dengan Hero lawan yang telah dipilih. Penggunaan teori himpunan memberikan pendekatan yang berguna dalam memandu pemilihan Hero dalam DOTA 2 dengan mempertimbangkan Counter lawan.

Kata Kunci*—himpunan, DOTA 2, pemilihan hero, *Counter

I. PENDAHULUAN

Seiring dengan perkembangan teknologi, permintaan akan kebutuhan hiburan pun semakin banyak setiap tahunnya. Teknologi kemudian berperan dengan mempermudah para *developer* untuk menciptakan permainan dalam kategori yang beragam. Pengembang Permainan atau *Game* kemudian berlomba-lomba untuk menciptakan *game* yang bisa dinikmati oleh setiap orang. *Video game online* sendiri sudah menjadi minat dari jutaan orang di dunia.

DOTA 2 (Defence of The Ancients 2) adalah salah satu *game online* berkategori MOBA (*Multiplayer Online Battle Arena*) yang dikenal di seluruh dunia. Berbagai *server* telah diluncurkan di berbagai belahan dunia sehingga rata-rata pemain bulanan DOTA 2 sendiri mencapai 434.863 pemain (per Oktober 2023). [1]

Gambar 1. Logo DOTA 2
(Sumber: https://i.pinimg.com/originals/8a/8b/50/8a8b50da2bc4afa933718061fe291520.jpg)

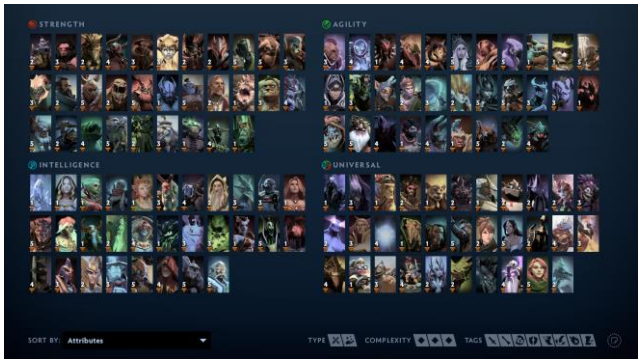
Pemain baru DOTA 2 membutuhkan waktu cukup lama hingga bisa terbiasa dengan mekanisme DOTA 2. Saat memainkannya, pemain harus mampu memperkirakan pergerakan lawan, memilih *item* yang cocok, bahkan mengingat berbagai *skill* yang dimiliki oleh *Hero* lawan. Dota 2 sendiri menganggap bahwa pemain baru membutuhkan waktu 100 jam *play time* untuk kemudian baru diizinkan untuk bermain dalam *Ranked Matchmaking*.

Salah satu bagian dari permainan DOTA 2 yang membutuhkan pemahaman yang luar biasa adalah saat pemilihan *Hero* yang akan dimainkan. Terdapat lebih dari 100 *Hero* yang dapat dipilih oleh pemain. Jumlah ini terlalu banyak sehingga cukup sulit untuk pemain biasa bisa langsung menyimpulkan *Hero* terbaik untuk memenangkan pertandingan saat itu.

Makalah ini akan menjelaskan mengenai penerapan teori himpunan untuk membantu pemain mempersempit pilihan *Hero* yang baik untuk dimainkan pada suatu pertandingan DOTA 2.

II. DASAR TEORI

A. *Hero DOTA 2*



Gambar 2. *Hero* di DOTA 2
(Sumber: *Game* DOTA 2)

Setiap *Hero* merupakan *Hero* yang unik dan memiliki keunggulan dan kekurangan. Karena itu, setiap *Hero* pasti memiliki *Hero* lain yang men-*Counter* dan di-*Counter* oleh dirinya. Sebagai contoh, Sniper memiliki keunggulan terhadap Undying karena Sniper memiliki daerah jangkauan serangan yang sangat jauh sehingga Undying akan kesulitan untuk mendekat tanpa kehilangan Sebagian besar darahnya. Di sisi lain, Sniper lemah melawan Anti-Mage karena salah satu

keahlian Anti-Mage adalah melakukan Blink atau berpindah tempat secara instan sehingga Anti-Mage dapat mendekati Sniper secara tiba-tiba tanpa masalah dan Sniper yang tidak memilliki darah dan armor yang tebal akan langsung habis oleh Anti-Mage.

Pemain tentunya ingin mencegah *Hero* yang dia pilih terancam oleh *Hero* pilihan lawan, oleh karena itu pemain perlu memperhatikan *Hero* apa saja yang sudah dipilih lawan saat pemilihan *Hero*.

B. Skema Pemilihan Hero

Setelah menemukan sebuah pertandingan, *Game* DOTA 2 akan mengarahkan pemain ke bagian Pra-pertandingan, yaitu tahap dimana para pemain di pertandingan tersebut secara bergantian memilih *Hero* yang akan dimainkan. (Mekanisme pemilihan *Hero* bergantung pada *Game Mode* yang dimainkan, pada makalah ini mengasumsikan *Game Mode All Pick Ranked Matchmaking* dimana pengguna di kedua tim memilih *Hero* secara bergantian)

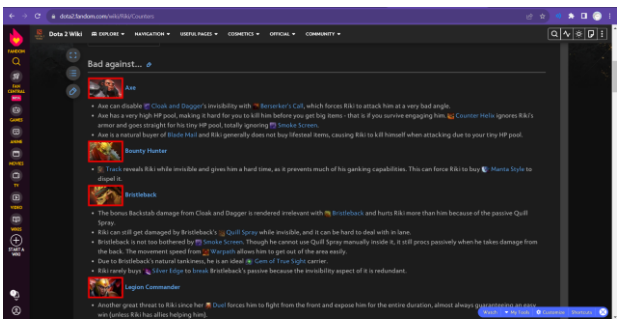
Normalnya, skema pemilihan *Hero* di pertandingan DOTA 2 adalah sebagai berikut

1. Setiap pemain memilih 1 *Hero* untuk di-Ban, beberapa *Hero* akan di-*ban* tergantung pada banyaknya pemain yang ingin suatu *Hero* tidak dimainkan
2. *First Pick* / Pemilih Pertama akan diacak antara kedua tim. Kemudian memilih 2 *Hero* pertama yang akan mereka mainkan.
3. Tim kedua kemudian memiliih 2 *Hero* yang akan dimainkan.
4. Langkah 2 dan 3 diulang hingga semua pemain memiliki *Hero* yang akan dimainkan. (Hanya 1 pemain yang memilih *Hero* terakhir, biasa disebut *Last Pick*).

Penerapan Himpunan dalam menentukan *Hero* yang akan dimainkan hanya dapat dilakukan jika pemain tidak melakukan *First Pick*, dan akan sangat efektif jika semua pemain lawan sudah memilih *Hero* yang akan dimainkan (*Last Pick*).

C. Forum Fandom DOTA 2

Fandom DOTA 2 adalah website forum terbuka untuk semua peminat *game* DOTA 2 untuk saling berbagai informasi dan belajar lebih dalam mengenai mekanisme DOTA 2. Salah satu informasi yang tersebar di sana adalah informasi mengenai daftar *Hero* yang baik dan buruk untuk digunakan ketika memainkan *Hero* tertentu. [2]



Gambar 3. Halaman *Counter*s dari *Hero* Riki
(Sumber: https://dota2.fandom.com/wiki/Riki/*Counter*s)

Dengan memanfaatkan informasi tersebut, proses pembuatan himpunan pilihan *Hero* yang mungkin dimainkan akan semakin akurat.

D. Himpunan

Himpunan (set) adalah kumpulan objek yang berbeda. Objek tersebut disebut sebagai elemen. Semua elemen pada sebuah himpunan harus berbeda dengan satu sama lain. Himpunan dapat dinyatakan dengan menggunakan tanda kurung kurawal "{}", diikuti dengan elemen-elemen yang ada di dalamnya. Contohnya, himpunan H = {1, 2, 3, 4, 5} adalah sebuah himpunan yang terdiri dari 5 elemen yaitu angka 1, 2, 3, 4, dan 5. [3]

Himpunan bisa dikenakan beberapa operasi dasar untuk mendapatkan himpunan baru sesuai kebutuhan, operasi yang dapat dilakukan diantaranya:

1. Irisan (*intersection)*
   Irisan berarti ketika dua buah himpunan diiriskan, maka akan terbentuk sebuah himpunan baru yang setiap elemennya terdapat pada himpunan pertama dan himpunan kedua.
2. Gabungan (*union)*
   Gabungan berarti ketika dua buah himpunan digabungkan, maka akan terbentuk himpunan baru yang elemennya terdapat pada himpunan pertama atau himpunan kedua.
3. Selisih (*difference*)
   Selisih berarti ketika himpunan pertama diselisihkan dengan himpunan kedua, maka akan terbentuk himpunan baru yang elemennya adalah elemen himpunan pertama yang tidak ada di himpunan kedua. [4]

E. Library Beautiful Soup 4

Beautiful Soup 4 merupakan sebuah library pada bahasa pemrograman Python yang berguna untuk mengolah data yang terdapat pada sebuah halaman web dalam bentuk html apapun yang dapat diakses melalui internet. [5] Library ini dapat membantu untuk mengestrak data dari website Forum Fandom DOTA 2 yang berisi nama-nama *Hero* yang baik untuk digunakan dan kemudian akan menjadi elemen-elemen dalam himpunan.

III. IMPLEMENTASI

A. Inisiasi Ekstraksi Data dengan Beautiful Soup 4

Sebelum melakukan operasi himpunan, data elemen himpunan yang akan digunakan perlu diekstraksi dari website Forum Fandom DOTA 2 dengan memanfaatkan *Library* Beautiful Soup 4 dalam bahasa Python. Pertama, pengguna perlu memasukkan nama *Hero*, yang dimainkan lawan.

```
# input nama hero
    nama_hero = (
        input("\033[94mMasukkan nama hero lawan: \033[0m")
    )
```

Gambar 4. *Skema menerima input nama Hero lawan*
(Sumber: Dokumentasi Pribadi)

Nama *Hero* lawan tersebut kemudian akan digunakan untuk mengakses halaman *Counter* dari *Hero* dengan memanfaatkan *Library* Beautiful Soup 4.

```
from bs4 import BeautifulSoup
import requests

    # Mengambil data dari website dota2.fandom.com sesuai dengan nama hero
    url = f"https://dota2.fandom.com/wiki/{nama_hero}/Counters"
    hero_yang_sudah_diinput.add(nama_hero)
    response = requests.get(url)
    with requests.get(url) as response:
        soup = BeautifulSoup(response.content, "html.parser")

```

Gambar 5. Penggunaan BS4 untuk ekstraksi halaman *Counter*
(Sumber: Dokumentasi Pribadi)

Setelah pembacaan halaman web selesai dilakukan, maka variabel 'soup' akan memuat teks html yang merupakan file html dari halaman *Counter* dari nama *Hero*. Langkah selanjutnya adalah melakukan seleksi data yang kemudian akan menjadi elemen dari himpunan.

Halaman *Counter* tersebut terbagi dalam beberapa bagian, namun hanya dua bagian yang perlu diperhatikan. Bagian pertama adalah bagian "Bad against…". Bagian ini memuat segala informasi terkait segala hal yang dapat mengancam *Hero* tersebut. Informasi ini termasuk *Hero* lain yang buruk jika dilawan, skill yang dapat merugikan *Hero* , hingga item yang dapat mengancam *Hero*. Data yang diekstraksi pada bagian ini hanyalah nama-nama *Hero* yang buruk jika dilawan oleh *Hero* tersebut.

Bagian kedua adalah bagian "Good against…". Bagian ini memuat segala informasi terkait segala hal yang menguntungkan untuk *Hero* tersebut. Informasi ini termasuk *Hero* lain yang dirugikan jika melawan *Hero* tersebut, skill yang dapat dilawan oleh *Hero* tersebut dengan baik, hingga item yang akan memperkuat *Hero* tersebut. Data yang diekstraksi dari bagian ini adalah nama-nama *Hero* yang baik jika dilawan oleh *Hero* tersebut.

Data tersebut kemudian dimasukkan ke dalam himpunan yang telah dibuat (dalam bahasa Python, digunakan istilah *set*) sebelumnya. Terdapat 3 himpunan yang dibutuhkan. Pertama, ada himpunan must_play yang elemennya adalah nama-nama *Hero* yang sebaiknya dimainkan oleh pengguna. Kedua, himpunan dont_play yang memuat nama *Hero* yang sebaiknya tidak dimainkan oleh pengguna. Selanjutnya, terdapat himpunan *Hero*_yang_sudah_diinput yang memuat semua nama *Hero* yang telah dimasukkan oleh pengguna sekaligus menunjukkan nama-nama *Hero* yang dimainkan oleh tim lawan. Terakhir, terdapat sebuah list yang memuat semua nama *Hero* di DOTA 2 yang tersimpan pada sebuah file .txt.

```
# inisiaisasi set / himpunan global
must_play = set()
dont_play = set()
hero_yang_sudah_diinput = set()

# baca file dota_heroes.txt
with open("dota_heroes.txt", "r") as f:
    dota_heroes = [line.rstrip().replace(" ", "_") for line in f]
```

Gambar 6. Inisasi *Set* / Himpunan
(Sumber: Dokumentasi Pribadi)

Setelah inisiasi himpunan selesai dilakukan, maka variabel soup sudah siap untuk mulai diekstrak datanya dan diolah dengan memanfaatkan teori himpunan.

B. Aplikasi Himpunan dalam Menentukan Hero yang Baik untuk Dimainkan

Secara garis besar, alur pengaplikasian teori himpunan untuk menentukan *Hero* yang baik untuk dimainkan adalah sebagai berikut:

1. Himpunan must_play, dont_play, dan *Hero*_yang_sudah_diinput yang bersifat *global*. Ketiga himpunan tersebut bersifat himpunan kosong ketika program baru dijalankan.
2. Data *Counter* dari halaman Forum Fandom DOTA 2 diekstrak dan dimasukkan ke dalam sebuah himpunan must_play_temp, dont_play_temp.
3. Lakukan operasi gabungan antara himpunan must_play dengan must_play_temp. Simpan hasilnya dalam himpunan must_play.

$$must\_play = (must\_play \cup must\_play\_temp) \quad (1)$$

4. Lakukan operasi gabungan antara himpunan dont_play dengan dont_play_temp. Simpan hasilnya dalam himpunan himpunan dont_play.

$$dont\_play = (dont\_play \cup dont\_play\_temp) \quad (2)$$

5. Lakukan operasi selisih untuk menghilangkan setiap *Hero* di must_play yang terdapat di himpunan dont_play.

$$must\_play = (must\_play - dont\_play) \quad (3)$$

6. Hilangkan *Hero* yang digunakan oleh tim lawan di himpunan must_play, dengan cara melakukan operasi selisih must_play dengan *Hero*_yang_sudah_diinput.

$$must\_play = (must\_play - hero\_yang\_sudah\_diinput) \quad (4)$$

7. Himpunan must_*Hero* sudah memuat *Hero* yang baik untuk dimainkan pengguna dan siap ditampilkan di layar.
8. Pengguna dapat memasukkan nama *Hero* lawan lainnya dan langkah 2, 3, 4, 5, 6, 7 akan diulang namun dengan himpunan must_play, dont_play, dan

*Hero*_yang_sudah_diinput yang sudah memiliki elemen dari hasil sebelumnya.

Perlu diingat bahwa himpunan must_play_temp dan dont_play_temp adalah himpunan yang memuat data *Counter* dari *Hero* yang baru dimasukkan pengguna.

Setelah operasi himpunan tersebut selesai dilakukan, himpunan must_play akan berisi nama-nama *Hero* yang baik untuk dimainkan oleh pengguna dan nama-nama tersebut dapat dipastikan bukan merupakan elemen dari himpunan dont_play. Sehingga pengguna tidak perlu memikirkan tentang *Hero* apa yang tidak di-*Counter* oleh lawan, namun pengguna cukup memilih salah satu *Hero* yang terdapat pada himpunan must_play dan terjamin bahwa *Hero* yang dipilih tidak di-*Counter* oleh lawan dan *Hero* tersebut cenderung merupakan *Counter* dari *Hero* lawan.

C. Membuat Program yang Sangkil dan Mangkus

Setelah penulis membuat program sesuai dengan alur garis besar sesuai dengan langkah-langkah yang dijelaskan pada poin B. Penulis melakukan beberapa perubahan untuk meningkatkan kualitas dari kode program tersebut. Sehingga dihasilkan kode program yang lebih sangkil dan mangkus.

```python
# Cari tag <a> dan <span> yang memiliki atribut id
for tag in soup.find_all(["a", "span"]):
    if "id" in tag.attrs:
        if tag["id"] == "Bad_against...":
            flag = "bad"
        elif tag["id"] == "Good_against...":
            flag = "good"
        elif tag["id"] == "Works_well_with...":
            # menghentikan pencarian jika sudah
            # menemukan tag dengan id "Works_well_with..."
            break
    elif (
        flag
        and tag.name == "a"
        and "title" in tag.attrs
        and tag["title"] in dota_heroes
    ):
        if flag == "bad":
            # jika hero tersebut sudah ada di must_play,
            # tempatkan di urutan pertama
            if tag["title"] in must_play:
                must_play.remove(tag["title"])
            # Secara tidak langsung, akan melakukan operasi gabungan antara
            # himpunan must_play sebelumnya dengan must_play hero yang baru
            # atau must_play (sebelumnya) ∪ must_play (hero yang baru)
            must_play.add(tag["title"])
        elif flag == "good":
            # Secara tidak langsung, akan melakukan operasi gabungan antara
            # himpunan dont_play sebelumnya dengan dont_play hero yang baru
            # atau dont_play (sebelumnya) ∪ dont_play (hero yang baru)
            dont_play.add(tag["title"])
```

Gambar 7. *Source code* bagian ekstraksi data
(Sumber: Dokumentasi Pribadi)

Penulis melakukan beberapa peningkatan pada bagian ekstraksi data dari halaman Forum Fandom DOTA 2 yang tidak merubah hasil akhir dan garis besar dari program pada poin B.

Peningkatan pertama, penulis mengatur program untuk hanya memeriksa setiap tag ‘a’ dan ‘span’ pada kode HTML. Hal ini dilakukan setelah melihat bahwa data nama *Hero* yang dibutuhkan sudah cukup hanya dengan melihat bagian isi dari tag ‘a’ dan ‘span’ saja. Jumlah iterasi yang dilakukan oleh program juga akan berkurang drastis karena hanya memeriksa isi dari kedua tag tersebut.

Peningkatan kedua terletak pada penghilangan himpunan must_play_temp dan dont_play_temp. Hal ini dilakukan untuk mengurangi memori yang digunakan oleh program. Pengurangan memori ini akan berdampak positif terhadap program saat dijalankan. Sebagai gantinya, penulis mengatur agar data yang baru diekstraksi akan langsung dimasukkan ke dalam himpunan yang sudah ada.

Peningkatan terakhir terletak pada saat memasukkan nama *Hero* ke dalam himpunan must_play. Jika nama *Hero* tersebut sudah menjadi elemen dari himpunan must_play, maka nama tersebut akan dimasukkan ulang sehingga terletak paling depan. Hal ini akan meningkatkan keakuratan program karena urutan awal elemen nama *Hero* yang terdapat pada himpunan must_play kemungkinan besar merupakan *Counter* dari *Hero* lawan yang telah dimasukkan pengguna sebelumnya dan saat ini.

Setelah kode pada bagian ekstraksi selesai dieksekusi, maka elemen dari himpunan must_play dan dont_play sudah siap digunakan dan diolah. Maka langkah selanjutnya adalah melakukan operasi selisih. Penulis melakukan sedikit perubahan skema dimana himpunan must_play dan dont_play diselisihkan dengan himpunan *Hero*_yang_sudah_diinput. Kemudian himpunan must_play diselisihkan dengan himpunan dont_play.

```python
# hapus nama hero dan semua nama hero yang sudah diinput dari set must_play dan dont_play
# must_play - hero_yang_sudah_diinput
must_play.difference_update(hero_yang_sudah_diinput)
# dont_play - hero_yang_sudah_diinput
dont_play.difference_update(hero_yang_sudah_diinput)

# Melakukan operasi selisih antara himpunan must_play dan dont_play
# must_play - dont_play
must_play.difference_update(dont_play)
```

Gambar 8. *Source code* bagian operasi selisih
(Sumber: Dokumentasi Pribadi)

Setelah proses tersebut selesai dieksekusi, maka secara garis besar program telah sukses melakukan operasi untuk mengisi himpunan must_play dengan nama-nama *Hero* sesuai yang diharapkan. Operasi yang dilakukan oleh program tersebut secara keseluruhan adalah sebagai berikut:

$$\begin{aligned} must\_play \; = (&must\_play \cup must\_play\_temp) \\ &- (dont\_play \cup dont\_play\_temp) \\ &- hero\_yang\_sudah\_diinput \end{aligned} \qquad (5)$$

Jika dibandingkan dengan garis besar pengaplikasian himpunan untuk menentukan *Hero* yang baik untuk dimainkan pada poin B, operasi yang dilakukan sudah ekivalen. Setelah proses ini, maka elemen dari himpunan must_play sudah siap untuk ditampilkan. Maka, langkah selanjutnya adalah menampilkan nama *Hero* apa saja yang dimainkan oleh lawan dan menampilkan nama-nama *Hero* pada himpunan must_play.

```python
# Menampilkan nama hero lawan di hero_yang_sudah_diinput
# dengan warna hijau
print("Hero yang dimainkan lawan:", end="")
for hero in hero_yang_sudah_diinput:
    print(f"\033[92m {hero.replace('_', ' ')}\033[0m,", end="")
# Menghilangkan koma di akhir
print("\b \n")
# Menampilkan nama hero yang harus dimainkan
print("Hero yang baik untuk dimainkan:")
for i, hero in enumerate(top_heroes, start=1):
    print(f"{i}. {hero}")
print()
```

Gambar 9. *Source code* bagian menampilkan hasil
(Sumber: Dokumentasi Pribadi)

Penulis melakukan proses menampilkan ke layar dengan menambahkan warna pada teks untuk meningkatkan estetika dan mempermudah pengguna untuk membaca informasi.

```
Hero yang dimainkan lawan: Riki, Ancient Apparition,
Hero yang baik untuk dimainkan:
1. Slardar
2. Oracle
3. Phoenix
4. Terrorblade
5. Lycan
6. Tidehunter
7. Axe
8. Razor
9. Viper
10. Juggernaut
```

Gambar 10. Contoh menampilkan hasil ke layar
(Sumber: Dokumentasi Pribadi)

Langkah terakhir dari program adalah memberikan pilihan kepada pengguna untuk memasukkan *Hero* lawan yang baru. Pengguna juga perlu diberikan pilihan untuk mengosongkan kembali himpunan dan pilihan untuk keluar dari program.

```python
print(
        "ketik \033[31mreset\033[0m untuk mengosongkan set atau \033[31mexit\033[0m untuk keluar dari program"
    )
    main()
```

Gambar 11. *Source code* setelah menampilkan hasil
(Sumber: Dokumentasi Pribadi)

```python
# input nama hero
    nama_hero = (
        input("\033[94mMasukkan nama hero lawan: \033[0m")
        .title()
        .strip()
        .replace(" ", "_")
    )
    if nama_hero == "Exit":
        exit()
    elif nama_hero == "Reset":
        must_play = set()
        dont_play = set()
        hero_yang_sudah_diinput = set()
        print("Set telah direset")
        main()
    elif nama_hero not in dota_heroes:
        print("\033[91mHero tidak ditemukan\033[0m")
        main()
    elif nama_hero in hero_yang_sudah_diinput:
        print("\033[91mHero sudah diinput\033[0m")
        main()
```

Gambar 12. *Source code handle input* pengguna
(Sumber: Dokumentasi Pribadi)

Perlu diingat bahwa fungsi main adalah fungsi utama keseluruhan program. Sehingga setiap pemanggilan fungsi main berarti mengeksekusi program dari awal.

Setelah program selesai diprogram, maka program sudah selesai dan siap digunakan oleh pengguna.

IV. EKSPERIMEN

Kasus yang paling ideal dan paling optimal untuk menunjukkan kinerja dari program adalah kasus ketika pengguna melakukan *Last Pick*, sehingga pengguna memiliki 5 nama *Hero* lawan yang kemudian dimasukkan pada program untuk menemukan *Hero* terbaik untuk digunakan. Nama *Hero* lawan yang akan digunakan pada eksperimen ini berasal dari *Game Mode All Pick Ranked Matchmaking* penulis yang terbaru, dimana tim lawan memainkan *Hero* Rubick, Drow Ranger, Lina, Warlock, dan Spirit Breaker.

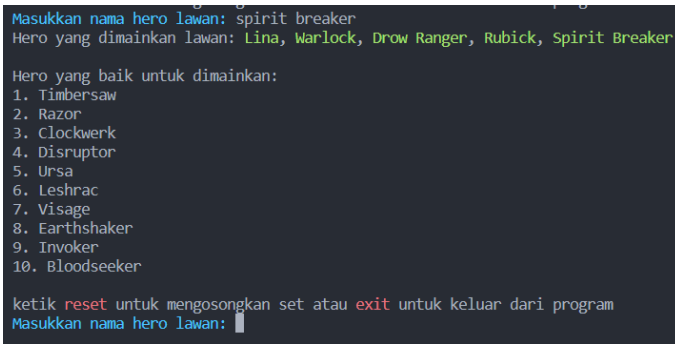
Pengujian pertama yaitu ketika pengguna memasukkan satu nama *Hero*, misalkan Rubick. Maka hasilnya dapat dilihat pada Gambar 13.

```
Masukkan nama hero lawan: Rubick
Hero yang dimainkan lawan: Rubick

Hero yang baik untuk dimainkan:
1. Ursa
2. Anti-Mage
3. Riki
4. Clinkz
5. Bristleback
6. Necrophos
```

Gambar 13. Hasil program dengan masukkan Rubick
(Sumber: Dokumentasi Pribadi)

Jika diperhatikan, hanya terdapat 6 *Hero* yang direkomendasikan oleh program untuk melawan Rubick. Hal ini terjadi karena jika melihat halaman *Counter* dari Rubick , maka dapat dilihat bahwa hanya 6 *Hero* tersebut yang merupakan *Counter* dari Rubick. [6] Selanjutnya penulis akan memasukkan sisa dari nama *Hero* yang belum dimasukkan, yaitu Drow Ranger, Lina, Warlock, dan Spirit Breaker.



Gambar 14. Hasil program setelah lima *Hero* dimasukkan
(Sumber: Dokumentasi Pribadi)

Pada Gambar 14, terlihat bahwa program sudah menampilkan 10 nama *Hero* dengan benar dan jika halaman *Counter* dari setiap *Hero* diperiksa di halaman Forum Fandom DOTA 2, maka dapat dipastikan bahwa tidak ada *Hero* yang ditampilkan di layar di-*Counter* oleh *Hero*-*Hero* yang digunakan oleh lawan. Maka, dengan eksperimen ini sebagai buktinya, penulis menyatakan bahwa program sudah bisa berjalan dengan semestinya.

V. KESIMPULAN

Pemilihan *Hero* pada Game DOTA 2 adalah salah satu dari banyak situasi yang membutuhkan ilmu mendalam terkait Game DOTA 2 secara keseluruhan. Terdapat banyak metode yang bisa memberikan saran kepada pengguna untuk memilih *Hero* dengan mempertimbangkan banyak faktor di dalam Game. Hasil dari penelitian ini dapat dikembangkan atau diintegerasikan dengan teori lainnya untuk meningkatkan kualitas dari pilihan *Hero* yang dapat dipilih oleh pengguna. Akan tetapi, untuk sekedar menghindari *Hero* yang dipilih pengguna di-*Counter* oleh musuh, maka penggunaan Teori Himpunan untuk mempersempit pilihan *Hero* sudah cukup.

VI. PENUTUP

Terima kasih kepada Allah SWT karena atas nikmat dan rahmat-Nya, penulis dapat menyelesaikan makalah berjudul "Aplikasi Himpunan dalam Pemilihan *Hero* pada Game DOTA 2" dengan baik. Selain itu, tidak lupa juga ucapan terima kasih kepada dosen mata kuliah Matematika Diskrit, Dr. Nur Ulfa Maulidevi, S. T, M. Sc., Dr. Ir. Rinaldi Munir, M. T., dan Dr. Fariska Zakhralativa Ruskanda, S.T. yang telah membimbing penulis selama berkuliah di mata kuliah ini. Penulis juga mengucapkan terima kasih kepada seluruh sumber yang dijadikan referensi pada makalah ini.

REFERENSI

[1] Steam, "Steam Charts DOTA 2," [Online]. Available: https://steamcharts.com/app/570. [Accessed 7 Desember 2023].

[2] F. G. Community, "DOTA 2 Wiki," [Online]. Available: https://dota2.fandom.com/wiki/Dota_2_Wiki. [Accessed 7 Desember 2023].

[3] M. Dr. Ir. Rinaldi, "Teori Himpunan (Bagian 1)," 2023. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/01-Himpunan(2023)-1.pdf. [Diakses 7 Desember 2023].

[4] M. Dr. Ir. Rinaldi, "Teori Himpunan (Bagian 2)," 2023. [Online]. Available: https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/02-Himpunan(2023)-2.pdf. [Accessed 7 Desember 2023].

[5] L. Richardson, "Beautiful Soup 4.12.0 documentation," 2023. [Online]. Available: https://www.crummy.com/software/BeautifulSoup/bs4/doc/. [Accessed 7 12 2023].

[6] F. G. Community, "Rubick/*Counter*s," [Online]. Available: https://dota2.fandom.com/wiki/Rubick/*Counter*s. [Accessed 8 Desember 2023].

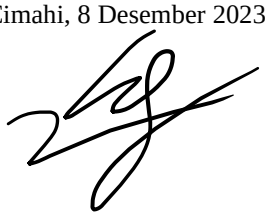
TAUTAN KODE PROGRAM

https://github.com/TazakiN/DOTA2_Counter_Picker_Helper

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Cimahi, 8 Desember 2023



Tazkia Nizami 13522032