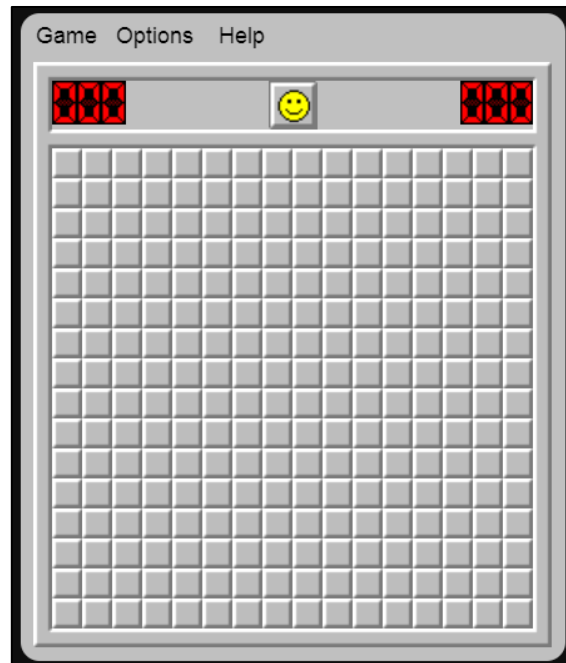


Nama : Baihaqi Rizki Nurfajri

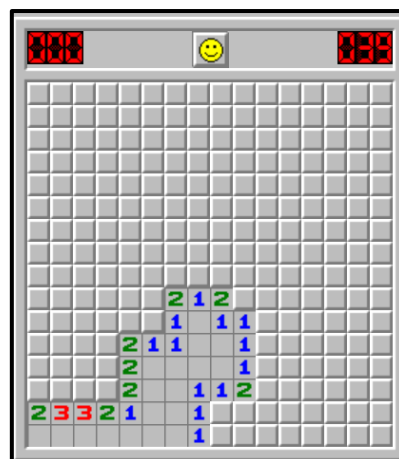
NRP : 5025211044

Github : <https://github.com/BeefRa/MineSweeper>

Analisis:



Pada game MineSweeper, saat kita menekan salah satu kotak, maka dia akan menunjukkan angka yang menandakan bahwa ada bom dengan jarak 3x3 dengan kotak yang kita tekan sebagai titik pusat. Namun, ada beberapa kejadian saat kita menekannya, maka akan banyak kotak yang terbuka seperti berikut:



Melalui hal ini, dapat kita terapkan suatu algoritma yang secara otomatis mencari neighbor dari kotak yang kita tekan secara otomatis, sehingga ketika pada neighbor tidak ditemukan bom secara berkala, maka kotak-kotak tersebut akan dibuka. Pada hal ini untuk mempercepat pencarian, kita dapat menerapkan BFS yang dapat mencari ke-8 tetangga dari

kotak yang kita tekan secara bersamaan hingga bom ditemukan sebagai neighbor dari kotak saat ini yang sedang dijelajahi oleh BFS.

Dalam penerapannya, pertama akan dilakukan peletakan bom secara ideal, yaitu 20.3% dari total kotak yang akan diterapkan. Pada kali ini, digunakan 8, 10, dan 16 grid, sehingga banyak bom secara berurutan adalah 13, 20, dan 52. Pada proses awal pemain akan diminta untuk memasukkan jumlah grid yang tertera.

```
def initiate_game():
    root = tk.Tk()
    root.withdraw()
    size = simpledialog.askinteger("Grid Size", "Enter grid size (8, 10, or 16):", minvalue=8, maxvalue=16)
    if size not in [8, 10, 16]:
        messagebox.showerror("Error", "Invalid grid size!")
        root.destroy()
        return
    if size == 8:
        mines_count = 12
    elif size == 10:
        mines_count = 20
    else:
        mines_count = 52
    game_window = tk.Toplevel(root)
    game = minesweeper(game_window, size, mines_count)
    game_window.mainloop()
```

```
def place_mines(self):
    while len(self.mine_locations) < self.mines_count:
        x, y = random.randint(0, self.size - 1), random.randint(0, self.size - 1)
        self.mine_locations.add((x, y))
```

Random.randint memungkinkan kita mendapatkan letak koordinat x dan y dengan ukuran grid(size-1) secara acak dan menyimpan lokasi dari setiap koordinat hingga banyak data pada mine_locations sudah mencapai jumlah bom yang sudah ditentukan (20.3%). Kemudian, akan dilakukan perhitungan pada setiap kotak untuk mengetahui berapa jumlah bom yang ada disekitarnya dengan luas 3x3 dengan kotak tersebut sebagai center-nya. Perhitungan tersebut akan menandai kotak ketika dia dipencet atau ketika dia terbuka melalui BFS, apakah kotak tersebut akan memunculkan angka penanda bom, kosong karena tidak ada bom disekitar, atau merupakan bom.

```
def calculate_adjacent_mines(self):
    self.adjacent_mine_counts = {}
    for x in range(self.size):
        for y in range(self.size):
            if (x, y) in self.mine_locations:
                self.adjacent_mine_counts[(x, y)] = -1
            else:
                self.adjacent_mine_counts[(x, y)] = sum(
                    (nx, ny) in self.mine_locations
                    for nx in range(x - 1, x + 2)
                    for ny in range(y - 1, y + 2)
                )
```

Pada kode di atas, adjacent_mine_count akan menghitung bom di areasekitar kotak, apabila kotak tersebut adalah bom maka akan diberikan nilai -1, kemudian akan dilakukan proses iteratif untuk menghitung jumlah bom pada area di sekitar kotak saat ini, berikut untuk visualisasinya:

(x-1, y+1)	(x, y+1)	(x+1, y+1)
(x-1, y)	(x, y)	(x+1, y)

(x-1, y-1) (x, y-1) (x+1, y-1)

Setelah proses tersebut, maka pada saat player menekan salah satu kotak, maka kotak akan terbuka sesuai dengan nilai dari fungsi sebelumnya. Untuk menentukan apa yang akan terjadi setelah nilai terungkap maka diterapkan fungsi sebagai berikut:

```
def reveal_cell(self, x, y):
    if (x, y) in self.revealed_cells:
        return
    elif (x, y) in self.mine_locations:
        self.end_game(False)
        return
    self.reveal_BFS(x, y)
    if len(self.revealed_cells) == self.size * self.size - self.mines_count:
        self.end_game(True)
```

```
def end_game(self, won):
    for (x, y) in self.mine_locations:
        self.buttons[(x, y)].config(text="M", state=tk.DISABLED, relief=tk.SUNKEN, bg='red')
    if won:
        messagebox.showinfo("MineSweeper", "You Won!!!")
    else:
        top = tk.Toplevel(self.master)
        top.title("Game Over")
        top.geometry("200x100")
        tk.Label(top, text="BOOM", font=("Helvetica", 24)).pack()
        tk.Label(top, text="you Loose", font=("Helvetica", 14)).pack()
        self.master.after(2000, self.master.destroy)
```

Pada kode tersebut, pertama akan dilakukan pemeriksaan, apabila kotak sudah dibuka, maka tidak akan mengembalikan apapun. Kemudian, apabila kotak merupakan lokasi dari bom, maka permainan berakhir game over yang ditandai dengan endgame(False). Apabila tidak keduanya, maka akan dilakukan pembukaan kotak secara iteratif dengan menggunakan BFS, apabila pada prosesnya banyak dari kotak yang terbuka berukuran dari pilihan grid(8x8 / 10x10 / 16x16) – total bom yang ada pada permainan, maka endgame(true) atau won.

```
def reveal_BFS(self, x, y):
    queue = deque([(x, y)])
    while queue:
        cx, cy = queue.popleft()
        if (cx, cy) in self.revealed_cells:
            continue
        self.revealed_cells.add((cx, cy))
        count = self.adjacent_mine_counts[(cx, cy)]
        if count == -1:
            self.buttons[(cx, cy)].config(text="M", state=tk.DISABLED, relief=tk.SUNKEN, bg='red')
        else:
            color = self.get_number_color(count)
            if count == 0:
                text = ""
            else:
                text = str(count)
            self.buttons[(cx, cy)].grid_forget()
            label = tk.Label(self.master, text=text, width=3, height=1, bg="light grey", fg=color, relief=tk.SUNKEN)
            label.grid(row=cy, column=cx)
            self.buttons[(cx, cy)] = label
        if count == 0:
            for nx in range(cx - 1, cx + 2):
                for ny in range(cy - 1, cy + 2):
                    if 0 <= nx < self.size and 0 <= ny < self.size and (nx, ny) not in self.revealed_cells:
                        queue.append((nx, ny))
```

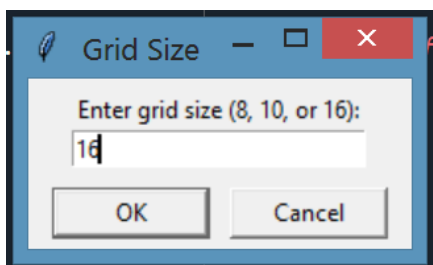
Pada metode ini, deque digunakan karena dapat melakukan append dan pop pada setiap ujungnya sehingga proses dapat lebih murah dan cepat. Setelah inisialisasi awal posisi x dan y pada queue, selama tidak kosong maka akan dilakukan pencarian. Pertama kotak saat

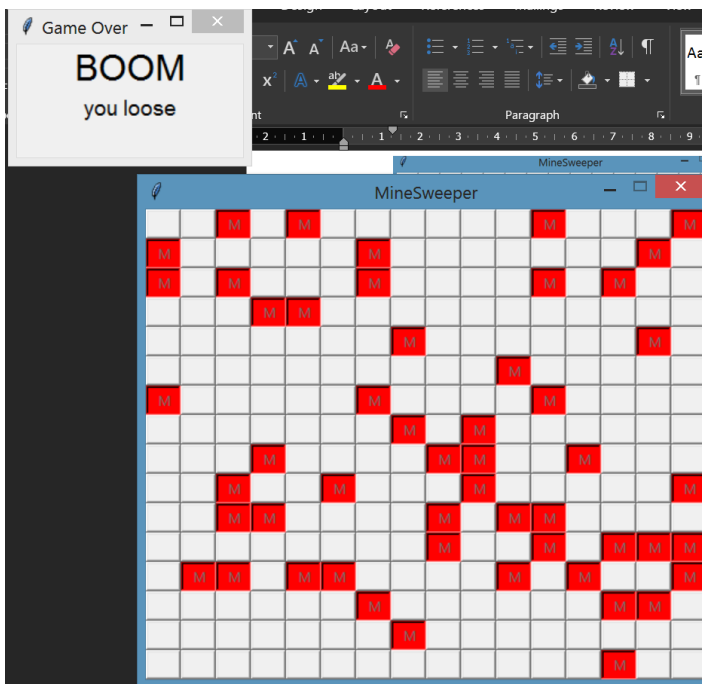
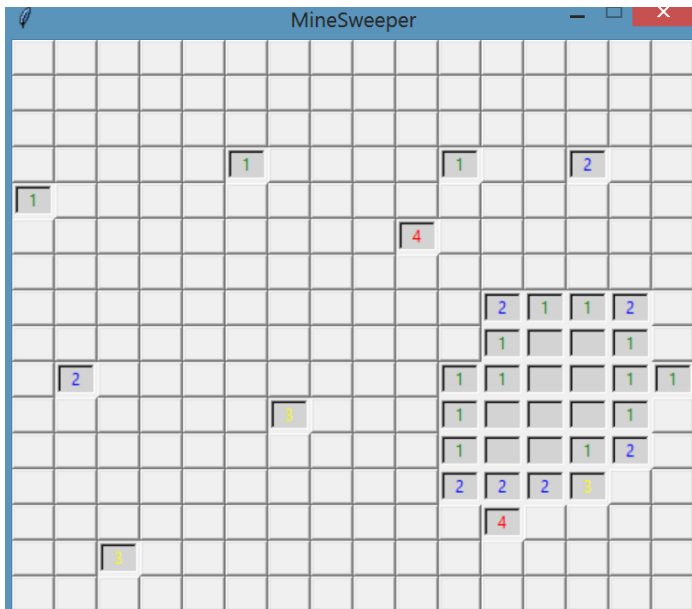
ini akan diperiksa, apakah dia sudah dibuka atau belum, jika sudah maka akan melewati proses iterasi, jika belum, maka akan ditambahkan pada kotak yang sudah dibuka dan dilakukan pencarian berapa banyak bom di area sekitar dengan fungsi `adjacent_mine_counts` yang sudah dilakukan pada awal pembeantukan game. Jika nilai kotak saat ini pada `adjacent_mine_counts` adalah -1, sama seperti sebelumnya, maka kotak tersebut adalah bom yang akan menampilkan huruf M dan kota berubah warna menjadi merah, karena kode ini dipanggil pada fungsi `reveal_cell`, maka `end_game(False)` atau pesan kekalahan akan dipanggil.

Pada tahap ini, jika kotak saat ini bukanlah bom, maka akan diambil warna untuk text dari fungsi `get_bumber_color`. Jika tidak ada bom sama sekali maka text kosong, jika ada maka text adalah jumlah dari bom yang sudah dicari dengan fungsi `adjacent_mine_counts`. Untuk menampilkan angka, dibutuhkan label, karena untuk menerapkan warna pada text tidak bisa langsung pada kotak, sehingga label digunakan agar teks angka dapat berwarna. Selanjutnya, ketika pada kotak saat ini tidak ditemukan ranjau, maka `neighbor` dari kotak saat ini akan ditambahkan kedalam antrian untuk diproses dengan menggunakan `append`.

Game Minesweeper juga memungkinkan pengguna untuk mendai kotak sebagai flag untuk mempermudah pemain dalam membuat pilihan, berikut merupakan implementasi dan beberapa tangkapan layar untuk output dari game:

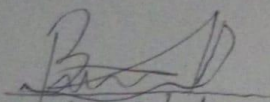
```
def mark_cell(self, x, y):
    btn = self.buttons[(x, y)]
    if btn["text"] == "F":
        btn.config(text="")
    elif btn["state"] == tk.NORMAL:
        btn.config(text="F")
```





By the name of Allah (God) Almighty, herewith I pledge and truly declare that I have solved
 Quiz 2 by myself, did not do any cheating by any means, did not do any plagiarism, and did not accept
 anybody's help by any means. I am going to accept all of the consequences by my name if it has
 proven that I have done any cheating and/or plagiarism".

Sundara, May 24th 2024


 Barhaes Rizki Nurprati
 5025211044

