DSL Homework

محمدصالح پژند – 400521171

در ابتدا فایل گرامر را تغییر میدهیم. hint را اضافه میکنیم.

و به program اضافه میکنیم:

```
start: program EOF;
program: output? hint? initiate_game bomb_placements;
```

در مرحله بعد وارد فایل listener میشویم و hint را به override_rules اضافه میکنیم. سپس تابع exiteHint را هم تعریف میکنیم:

```
def exitHint(self, ctx):

make_ast_subtree(self.ast, ctx, node_value: "hint", keep_node=True)

33
```

در مرحله بعد وارد code generator میشویم و hint را به non_operands اضافه میکنیم تا در قسمت های بعد نوع ان معلوم باشد. در تابع generate_code_based_on_non_operands یک elseif هم مانند قسمت های دیگر برای hint در نظر میگیریم.

در تابع generate_program اینگونه میکنیم:

```
def generate_program(self):
    placements_code = self.code_stack.pop()
    initiate_code = self.code_stack.pop()

    hint_type = 'False'

if len(self.code_stack) > 0:
    temp_code = self.code_stack.pop()
    if temp_code.startswith('##COMPILER_PARAM:::boolean:::'):
        hint_type = temp_code.replace('##COMPILER_PARAM:::boolean:::', '')
        print(hint_type)
    else:
        self.code_stack.append(temp_code)

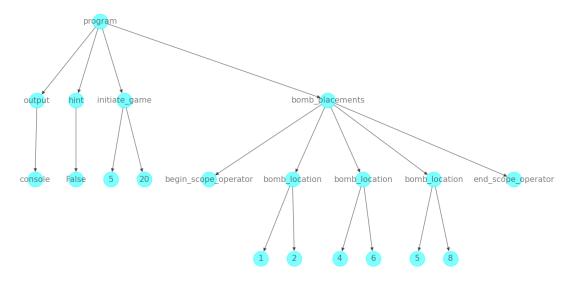
output_type = 'console'
```

این قسمت دقیقا مانند output_type میباشد ولی باید قبل ان قرار بگیرد چون در استک بالاتر است پس زودتر باید پاپ شود. نوع دیفالت ان را هم false قرار میدهیم چون ممکن است کاربر اصلا hint را وارد نکند. بعد از این قسمت ما hint_type را داریم و بر اساس ان دو نوع خروجی تولید میکنیم:

در صورت false بودن همان کد قبل تولید میشود ولی در صورت true بودن یک تابع تعریف میکنیم تا تعداد همسایه های true را بشمار د و خروجی مناسب را قرار

Tests:

```
1    output: console
2    hint: False
3    game: 5 X 20
4    bomb: 1 , 2
5    bomb: 4 , 6
6    bomb: 5 , 8
```



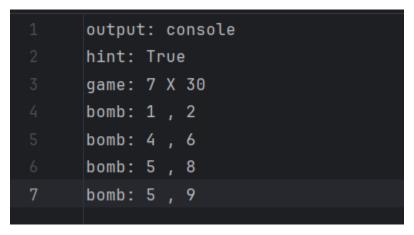
```
bombs = [[False for y in range(20)] for x in range(5)]
bombs[0][1] = True

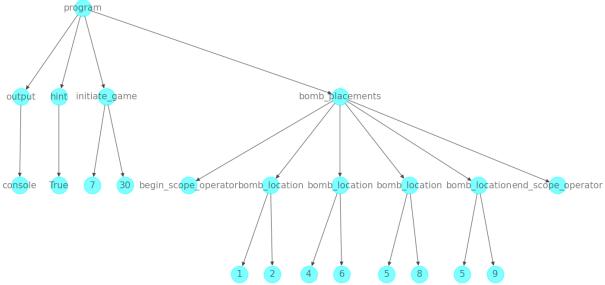
bombs[3][5] = True

bombs[4][7] = True

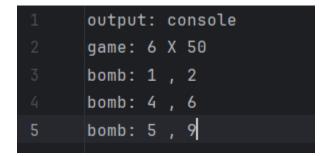
for row in bombs:
    for column in row:
        if not column:
             print('#', end_='')
        else:
        print()
```

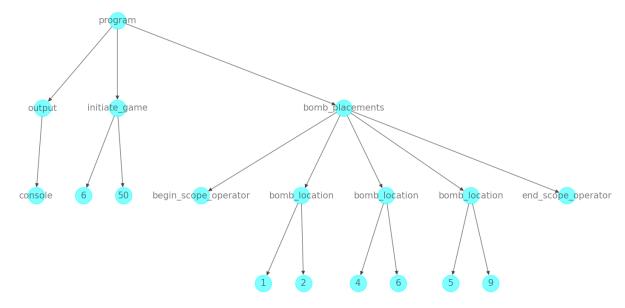
2-





```
bombs = [[False for y in range(30)] for x in range(7)]
                             bombs[0][1] = True
                             bombs[3][5] = True
                             bombs[4][7] = True
                             bombs[4][8] = True
                             def count_neighbors(x, y):
                                                                               if dx == 0 and dy == 0:
                                                                                               continue
                                                                               if \theta <= nx < len(bombs) and \theta <= ny < len(bombs[0]) and bombs[nx][ny]:
                                              return count
                             for x in range(len(bombs)):
                                              for y in range(len(bombs[0])):
                                                              if not bombs[x][y]:
                                                                               neighbors = count_neighbors(x, y)
                                                                               print(neighbors, end='')
                                             print()
\verb|C:\Pr| ogramData an a conda3| python.exe "D: University documents Term 6 \\ | Compiler HW3 \\ | example_dsl \\ | output.py \\ | ou
000011100000000000000000000000000000
00001*2221000000000000000000000
0000112**100000000000000000000
000000122100000000000000000000
```





```
bombs = [[False for y in range(50)] for x in range(6)]
bombs[0][1] = True
bombs[3][5] = True
bombs[4][8] = True
for row in bombs:
    for column in row:
        if not column:
            print('#', end_='')
        else:
        print()
```