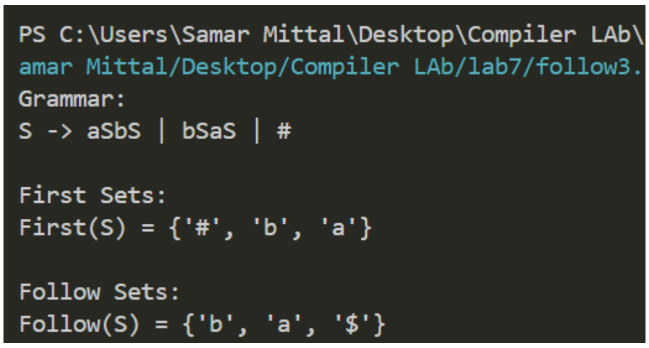
CD CODE

```
def first(c, prods, fs):
    if c not in fs: fs[c] = set()
    if c.islower():
         fs[c].add(c)
         return fs[c]
    for prod in prods.get(c, []):
        if prod == '#': fs[c].add('#')
        elif prod[0].islower():
            fs[c].add(prod[0])
        else:
            for char in prod:
                res = first(char, prods, fs)
                fs[c].update(res - {'#'})
                if '#' not in res:
                    break
                if char == prod[-1]:
                    fs[c].add('#')
    return fs[c]

def follow(c, start, prods, fs, follows):
    if c not in follows:
        follows[c] = set()
    if c == start:
        follows[c].add('$')
    for head, rules in prods.items():
        for prod in rules:
            if c in prod:
                idx = prod.index(c)
                if idx == len(prod) - 1:
                    if head != c:
                        follows[c].update(follow(head, start, prods, fs, follows))
                else:
                    next_sym = prod[idx + 1]
                    next_first = first(next_sym, prods, fs)
                    follows[c].update(next_first - {'#'})
                    if '#' in next_first:
                        follows[c].update(follow(head, start, prods, fs, follows))
    return follows[c]

def main():
    prods = {'S': ['ABC'],
              'A': ['DEF'],
                'B': ['#'],
                'C': ['#'],
                'D': ['#'],
                'E': ['#']
            }
    start = 'S'
    first_sets, follow_sets = {}, {}
    print("Grammar:")
    for nt, rules in prods.items():
```

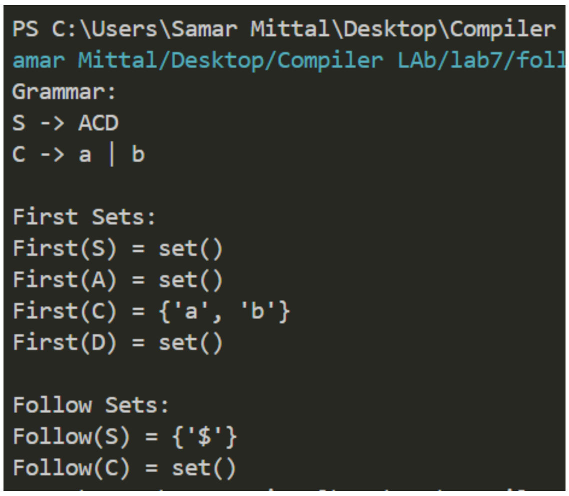
```python
        print(f"{nt} -> {' | '.join(rules)}")


    for nt in prods:
        first(nt, prods, first_sets)
    for nt in prods:
        follow(nt, start, prods, first_sets, follow_sets)

    print("\nFirst Sets:")
    for nt, fs in first_sets.items():
        if nt.isupper():
            print(f"First({nt}) = {fs}")

    print("\nFollow Sets:")
    for nt, fs in follow_sets.items():
        print(f"Follow({nt}) = {fs}")

if __name__ == "__main__": main()
```

OUTPUT :



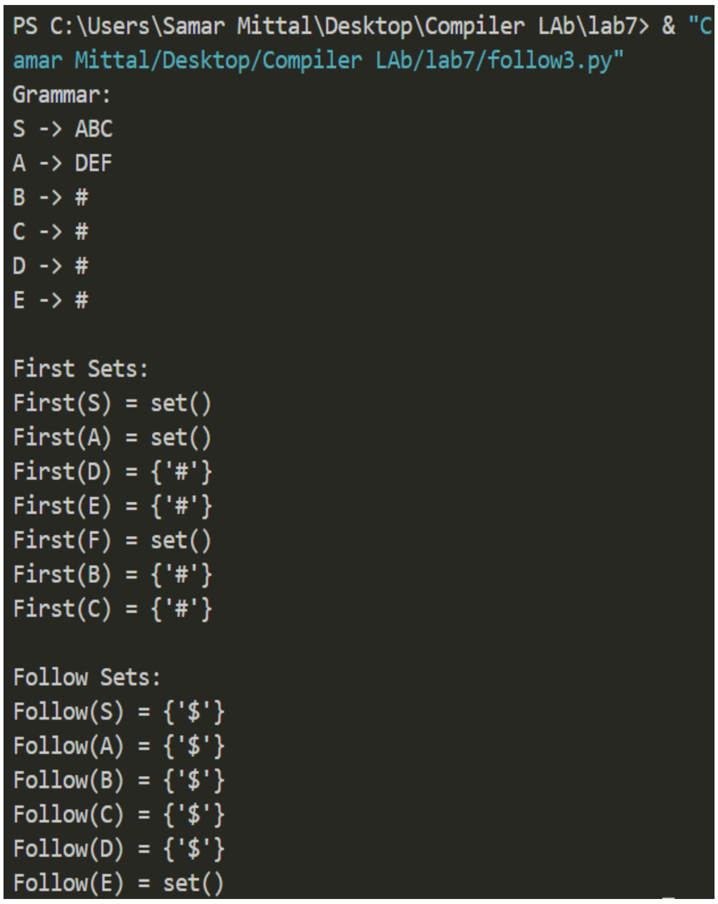
```
PS C:\Users\Samar Mittal\Desktop\Compiler LAb\
amar Mittal/Desktop/Compiler LAb/lab7/follow3.
Grammar:
S -> aSbS | bSaS | #

First Sets:
First(S) = {'#', 'b', 'a'}

Follow Sets:
Follow(S) = {'b', 'a', '$'}
```

TEST CASE 1



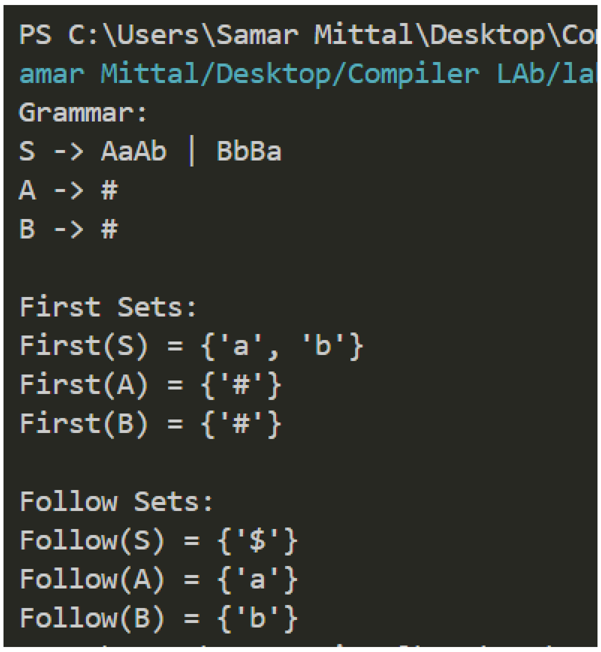
```
PS C:\Users\Samar Mittal\Desktop\Compiler
amar Mittal/Desktop/Compiler LAb/lab7/foll
Grammar:
S -> ACD
C -> a | b

First Sets:
First(S) = set()
First(A) = set()
First(C) = {'a', 'b'}
First(D) = set()

Follow Sets:
Follow(S) = {'$'}
Follow(C) = set()
```

TEST CASE 2



```
PS C:\Users\Samar Mittal\Desktop\Compiler LAb\lab7> & "C
amar Mittal/Desktop/Compiler LAb/lab7/follow3.py"
Grammar:
S -> ABC
A -> DEF
B -> #
C -> #
D -> #
E -> #

First Sets:
First(S) = set()
First(A) = set()
First(D) = {'#'}
First(E) = {'#'}
First(F) = set()
First(B) = {'#'}
First(C) = {'#'}

Follow Sets:
Follow(S) = {'$'}
Follow(A) = {'$'}
Follow(B) = {'$'}
Follow(C) = {'$'}
Follow(D) = {'$'}
Follow(E) = set()
```

TEST CASE 3



```
PS C:\Users\Samar Mittal\Desktop\Co
amar Mittal/Desktop/Compiler LAb/la
Grammar:
S -> AaAb | BbBa
A -> #
B -> #

First Sets:
First(S) = {'a', 'b'}
First(A) = {'#'}
First(B) = {'#'}

Follow Sets:
Follow(S) = {'$'}
Follow(A) = {'a'}
Follow(B) = {'b'}
```

TEST CASE 4