```
FSSP(S_0, G)
1  S ← S_0
2  π ← empty plan
3  loop
4      if G ⊆ S
5          return π
6      actions ← set of applicable actions in S
7      if actions is empty
8          return failure
9      choose a ∈ actions          ▷ choose an action
10     S ← γ(S, a)                 ▷ progress to new state
11     π ← π ∘ a                   ▷ update plan
```

Steps for generating plan @ 11
info for movegen: available
NO search strategy
Node - State of the world
movegen @ 6
non-det selection @ 9
In full impltn:
-- non-det selection handled by:
----choose 1/1 while backtracking
------on failure

```
BSSP(S_0, G)
1  G' ← G
2  π ← empty plan
3  loop
4      if G' ⊆ S_0
5          if G ⊆ γ(S_0, π) then return π
6          else return failure
7      actions ← set of relevant actions for G'
8      if actions is empty
9          return failure
10     choose a ∈ actions          ▷ choose an action
11     G' ← γ^{-1}(G', a)          ▷ regress to new goal
12     π ← a ∘ π                   ▷ update plan
```

Steps for generating plan @ 12
info for movegen: available
NO search strategy
Node - Goal/subgoal
movegen @ 7
non-det selection @ 10
In full impltn:
-- non-det selection handled by:
----choose 1/1 while backtracking
------on failure

```
GSP(givenState, givenGoal, actions)
1  π ← empty plan
2  stack ← empty stack
3  S ← givenState
4  PushSet(givenGoal, stack)
5  while stack is not empty
6      X ← pop stack
7      if X is an action a
8          π ← π ∘ a
9          S ← Progress(S, a)
10     else if X is a compound goal G, and G is not true in S
11         PushSet(G, stack)
12     else if X is a goal g, and g is not true in S
13         a ← choose a relevant action that achieves g
14         if a is null then return failure
15         push a to stack
16         PushSet(pre(a), stack)

PushSet(G, stack)
1  push G to stack
2  for each goal g in G
3      push g to stack
4  return

Progress(S, a)
1  return (S ∪ add-effects(a)) \ del-effects(a)
```

Steps for generating plan @ 8
info for movegen: available
Yes search strategy: Stack
Node - Goal/subgoal
movegen @ 13
non-det selection @ 13
In full impltn:
-- non-det selection handled by:
----choose 1/1 while backtracking
------on failure

```
PSP(π)
1  flaws ← OpenGoals(π) ∪ Threats(π)
2  if flaws is empty
3      return π                    ▷ a plan with no flaws
4  choose φ ∈ flaws                ▷ choose a flaw
5  resolvers ← Resolve(φ, π)
6  if resolvers is empty
7      return failure              ▷ a flaw with no resolvers
8  choose ρ ∈ resolvers            ▷ choose a resolver
9  π' ← Refine(ρ, π)
10 return PSP(π')
```

Steps for generating plan @ 9
info for movegen: available
No search strategy
Node - Partial plan
movegen @ 1,4,5
non-det selection @ 4,8
In full impltn:
-- non-det selection handled by:
----choose 1/1 while backtracking
------on failure