```
init module{
    knowledge{
        clear(table)  . clear(X) :- block(X), not(on(_, X)), not(holding(X)) .
        …
    }
    % no initial beliefs about block configuration.
    goals{
        on(a,b), on(b,c), on(c,table), on(d,e), on(e,f), on(f,table).
    }
    actionspec{
        pickup(X) { pre{ clear(X), not(holding(_)) }  post{ true }  }
        …
    }
}

% moving X on top of Y is a constructive move if that move results in X being in position.
#define constructiveMove(X, Y) a-goal(tower([X, Y|T])), … .

main module{
    program{
        if a-goal( holding(X) ) then pickup(X) . % put a block you're holding down.
        if bel( holding(X) ) then {
                if constructiveMove(X,Y) then putdown(X, Y) .
                if true then putdown(X, table) .
        }
    }
}

event module{
    program{
        #define inPosition(X) goal-a( tower([X|T]) ) . % block in position if it achieves a goal.

        % rules for processing percepts (assumes full observability).
        forall bel( block(X), not(percept(block(X))) ) do delete( block(X) ) .
        forall bel( percept(block(X)), not(block(X)) ) do insert( block(X) ) .
        …
    }
}

module adoptgoal{
                …
}
```

The **init** module initializes the agent, here by defining knowledge, an initial goal, and action specifications

Macro definitions to create more readable code.

The **main** module is used to code the agent's deliberation using rules for selecting actions.

Rules in the **event** module are used to process percepts and messages that the agent receives.

User-defined modules.