

Zaawansowane systemy decyzyjne

Laboratorium

prowadzący: Andrzej Czajkowski¹

Zmienne i bloki logiczne

1 Cel ćwiczenia.

Celem ćwiczenia jest zapoznanie się z możliwościami tworzenia zmiennych i bloków logicznych w programie Corvid Exsys w celu wykorzystania przy tworzeniu systemu ekspertowego.

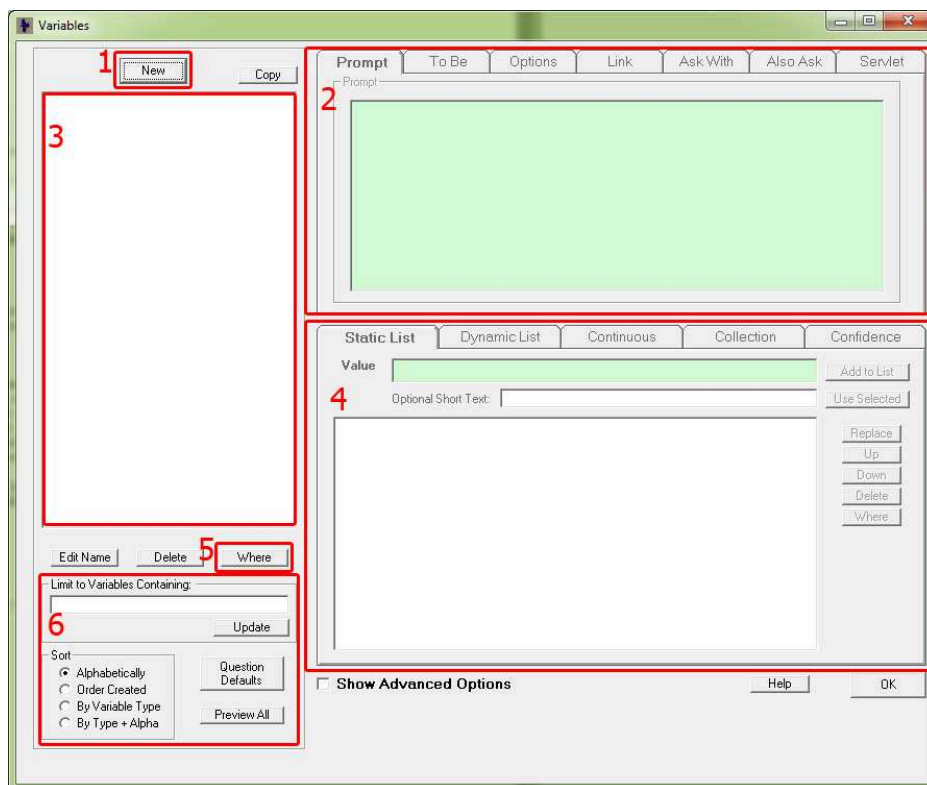
2 Instrukcje do wykonania.

2.1 Tworzenie zmiennych.

Zmienne są podstawą każdego budowanego systemu za pomocą Exsys Corvid. Zmienne są wykorzystywane podczas zadawania pytań, wyświetlania wyników jak i wnioskowania na podstawie stworzonej logiki działania.

Zmienne są wykorzystywane do budowania reguł oraz bloków logicznych. **Nie można przypisać zmiennych do reguł zanim te zmienne nie zostaną utworzone !!!**. Dlatego przed przystąpieniem do budowy systemu należy zaplanować jaką system będzie mieć funkcjonalność i jakich zmiennych będziemy potrzebować. Można oczywiście dodawać zmienne w trakcie powstawania zapotrzebowania podczas konstruowania reguł ale może to prowadzić do nieczytelności projektu.

Tworzenie zmiennych odbywa się poprzez okno zmiennych, które uruchamiamy przyciskiem  lub poprzez menu Windows->Variables.



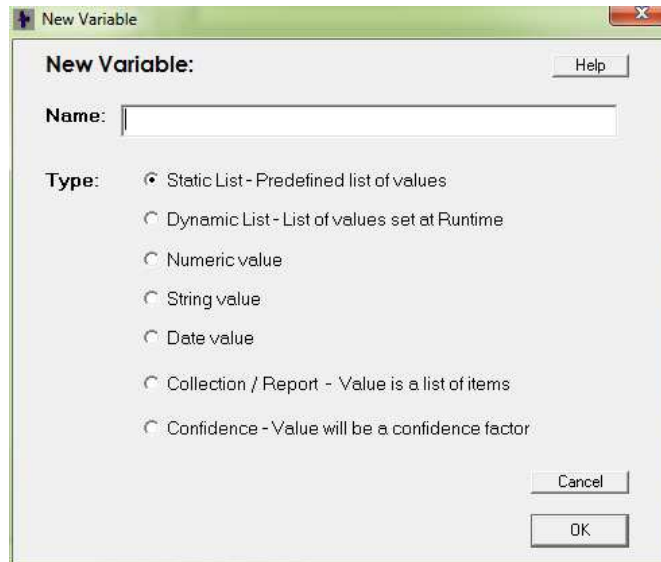
Najważniejsze składowe okna zmiennych:

1. przycisk tworzenia nowej zmiennej,

¹Andrzej Czajkowski, Institute of Control and Computation Engineering, University of Zielona Góra, ul. Podgórna 50, 65-246 Zielona Góra, Poland. Email: a.czajkowski@issi.uz.zgora.pl

2. obszar wspólnych opcji dla wszystkich typów zmiennych,
3. pole w którym są wyświetlane istniejące w projekcie zmienne,
4. obszar opcji dla wybranego typu zmiennej,
5. przycisk wyszukiwania gdzie zaznaczona zmienna jest wykorzystywana,
6. obszar filtrowania i sortowania istniejących zmiennych (po utworzeniu zmiennych należy samodzielnie sprawdzić poszczególne opcje).

Przy tworzeniu nowej zmiennej wyświetlane jest okno wyboru typu zmiennej oraz nadania jej nazwy:



Typy zmiennych:

- static list - lista statyczna, przeważnie służy do przechowywania pytań z wieloma odpowiedziami lub zmiennych, które wykorzystują stałą liczbę statycznych wartości (np. zmienna przełącznik z wartościami wł. i wył.),
- dynamic list - lista dynamiczna, podobnie jak lista statyczna przechowuje listę wartości z tym, że mogą one być modyfikowane i dodawane w trakcie działania systemu np. na skutek przeprowadzonego wnioskowania,
- numeric value - zmienna numeryczna, zmienna przechowująca wartość liczbową, która może być wykorzystana w wyrażeniach logicznych (np. jeśli zmienna_1 mniejsza od zmienna_2 to ...). Ponadto zmienna numeryczna powinna być wykorzystywana wszędzie gdzie mamy do czynienia z obliczeniami matematycznymi.
- string value - zmienna o postaci tekstu, wykorzystywane głównie w raportowaniu wyników (istnieje wiele funkcji wbudowanych do manipulowania zmiennymi znakowymi).
- date value - zmienna przechowująca datę. Istnieje również wiele funkcji wbudowanych do sprawdzania zakresu data czy obliczania przyszłej lub przeszłej daty.
- collection/Report - zmienna przechowująca listę wartości tworzona przez system podczas wnioskowania zazwyczaj w bloku THEN reguł logicznych. Przeważnie wykorzystywana w celu skonstruowania raportu, uwag czy porad dla użytkownika systemu. Kolekcje mogą być konstruowane w dowolny sposób np. w celu wygenerowania raportu w formacie HTML czy RTF.
- Confidence - zmienna przechowująca wartość dotycząca ufności, prawdopodobieństwa czy pewności. Przeważnie opis zmiennej tego typu opisuje zalecaną czynność lub możliwe zaistnienie konkretnego zdarzenia, które zostanie wskazane przez system. Wartość zmiennej opisują jak prawdopodobne bądź zalecane jest wykonanie tej czynności w sytuacji, która została utworzona poprzez udzielanie odpowiedzi przez użytkownika (np. w przypadku urządzenia z 3 elementami - silnik, przekładnia, komputer istnieją 3 zmienne dotyczące ufności - usterka silnika, usterka przekładni i usterka komputera. Po udzieleniu przez użytkownika wszystkich odpowiedzi, zgodnie z zaimplementowaną logiką wnioskującą zostaną ustawione zmienne ufności. Następnie wskazana zostanie usterka elementu o najwyższej wartości odpowiedniej zmiennej ufności).


W przypadku tego ćwiczenia należy stworzyć zmienne:

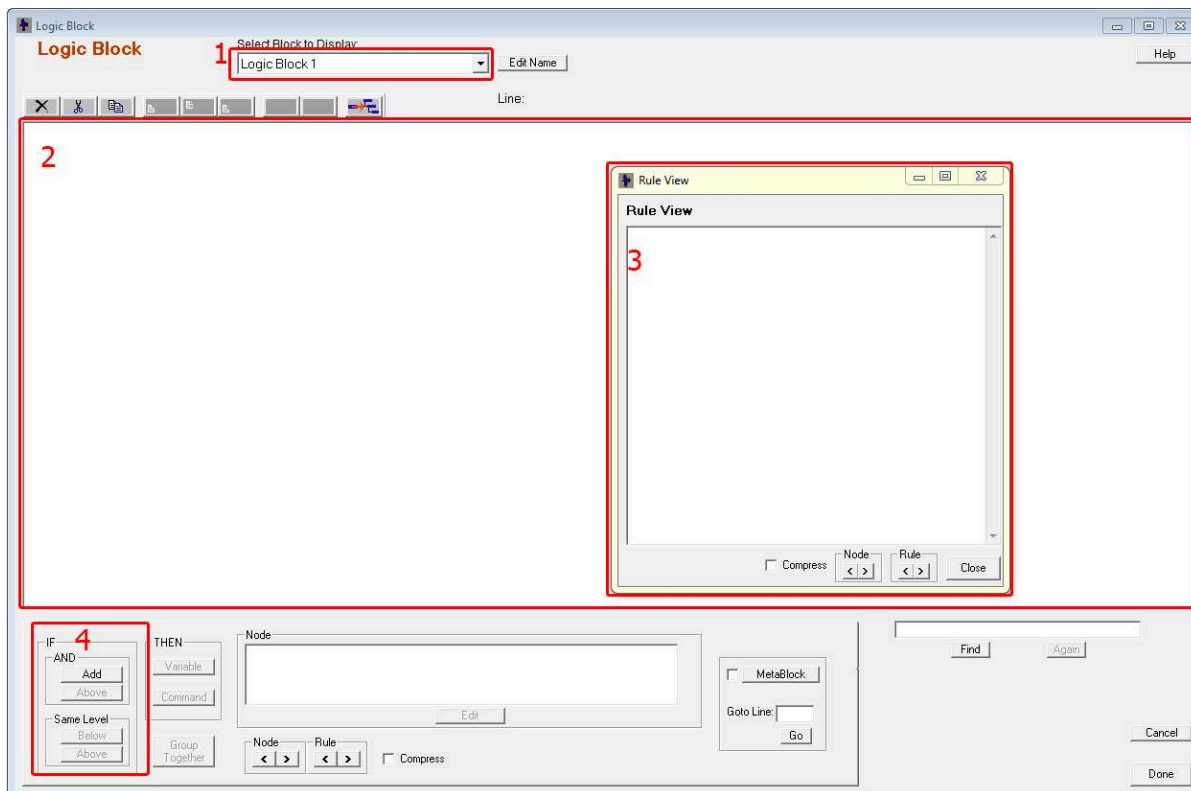
- lista statyczna o nazwie **dzień tygodnia** (nazwa może zawierać również polskie znaki, niedozwolone są znaki operatorów i spacje). W polu **Prompt** należy wpisać tekst odnoszący się do wartości zmiennej np. “dzisiaj jest”, można również wykorzystać znak dwukropka “dzień tygodnia:”. Do zmiennej w panelu opcji dotyczącym zmiennych typu **static list** należy dodać opcję: poniedziałek, wtorek ... niedziela. Dodawanie opcji wykonywane jest poprzez przycisk **Add to list** po uprzednim wpisaniu opcji w polu tekstowym. **Zanim zmienna zostanie wykorzystana w systemie można zmienić jej typ.** Opcja **Optional Short Text** służy do wprowadzenia skróconego tekstu opcji. Skrócony tekst jest wykorzystywany w diagramach drzew decyzyjnych w celu zwiększenia przejrzystości, skrócona nazwa nie jest dostępna dla użytkownika końcowego. Dodatkowo w opcjach w górnym bloku można ustawić wartość domyślną oraz w zakładce **Ask With** sposób odpytywania o zmienną (zostawiamy radio button gdyż nasza zmienna będzie miała tylko jedną możliwą wartość w dany czasie).
- lista statyczna o nazwie **Pogoda**. W **Prompt** wpisujemy “Na dworze jest”. Dodajemy opcje do listy: słonecznie, deszczowo, pochmurnie, zamieć.
- zmienną numeryczną **temperatura**. W **Prompt** wpisujemy “Temperatura na dworze wynosi”. W zakładce **Continuous** można ustawić limit górny i dolny, w razie złego wprowadzenia wartości przez użytkownika system odrzuci odpowiedź i ponowi zapytanie. Można zaznaczyć również opcję **Integers Only** w celu wymuszenia liczb całkowitych.
- zmienna dotycząca ufności o nazwie **idź na plażę**. W zakładce **Confidence** można określić jak będzie wyliczana wartość ufności dla zmiennej. W przypadku zwykłej zmiennej numerycznej po przypisaniu do zmiennej 4 a potem 2, zmienna będzie przechowywać wartość 2. W przypadku zmiennej ufności ustawionej na wyliczanie za pomocą sumy wartość będzie wynosiła 6. Natomiast gdy ustawimy sposób wyliczania **Average** czyli średnia, zmienna będzie wynosić 3. Dzięki temu za pomocą kolejnych reguł można zwiększać lub zmniejszać prawdopodobieństwo zdarzenia jakim np. w naszym przypadku jest wymiana źródła światła. Zmienne ufności mogą być wyliczane w przeróżny sposób np. na podstawie zależnego i niezależnego prawdopodobieństwa statystycznego.
- zmienna dotycząca ufności o nazwie **idź do pracy**. Sposób wyliczania za pomocą sumy.
- zmienna dotycząca ufności o nazwie **zostań w domu**. Sposób wyliczania za pomocą sumy.

2.2 Bloki logiczne

W środowisku Exsys Corvid wszystkie zbudowane systemy ekspertowe opierają się o reguły typu if/Then. Reguły te opisują poszczególne kroki jakie musi rozważyć ekspert podczas podejmowania decyzji. W wyniku takiego działania podjęta decyzja jest kombinacją wielu również bardzo skomplikowanych reguł. Poszczególne reguły mogą być bardzo skomplikowane lub złożone z innych reguł. Odzwierciedla to proces podejmowania decyzji w świecie rzeczywistym. Reguły w środowisku Exsys Corvid są zorganizowane w bloki logiczne. Dzięki czemu zbiory reguł są tworzone w prosty i przejrzysty sposób, jak również zarządzanie nimi jest intuicyjne. Nie ma określonych zasad budowy bloków logicznych jak i świecie rzeczywistym różni eksperci różnie mogą podejmować swoje decyzje. Narzędzia, które są udostępnione pozwalają na dowolną strategię budowy systemu. Dzięki temu możliwe jest odzwierciedlenie dowolnego sposobu myślenia przy rozwiązywaniu danego problemu.

Blok logiczny może być pojedynczą regułą jak również skomplikowanym drzewem decyzyjnym. Cały system może mieć jeden jak również wiele bloków logicznych. W większości zastosowań pojedynczy blok logiczny powinien zawierać wszystkie reguły dotyczące konkretnego problemu czy podejmowanej decyzji. Wielu projektantów systemów ekspertowych preferują budowę wielu bloków logicznych odpowiedzialnych za cząstkowe części systemów natomiast część woli budować duże bloki pokrywające duże porcje systemu. Wszystko zależy od rozwiązywanego problemu i sposobu myślenia danego projektanta. Podsumowując bloki logiczne należy budować zgodnie z zasadą aby były one ustrukturyzowane i zorganizowane tak by łatwo można było zrozumieć logikę jaka w nich jest zawarta. Istotne jest również aby bloki logiczne były kompletne i łatwe w zarządzaniu.

Tworzenie bloków logicznych odbywa się poprzez okno bloków logicznych, które uruchamiamy przyciskiem  lub poprzez menu **Windows->Logic Block**.



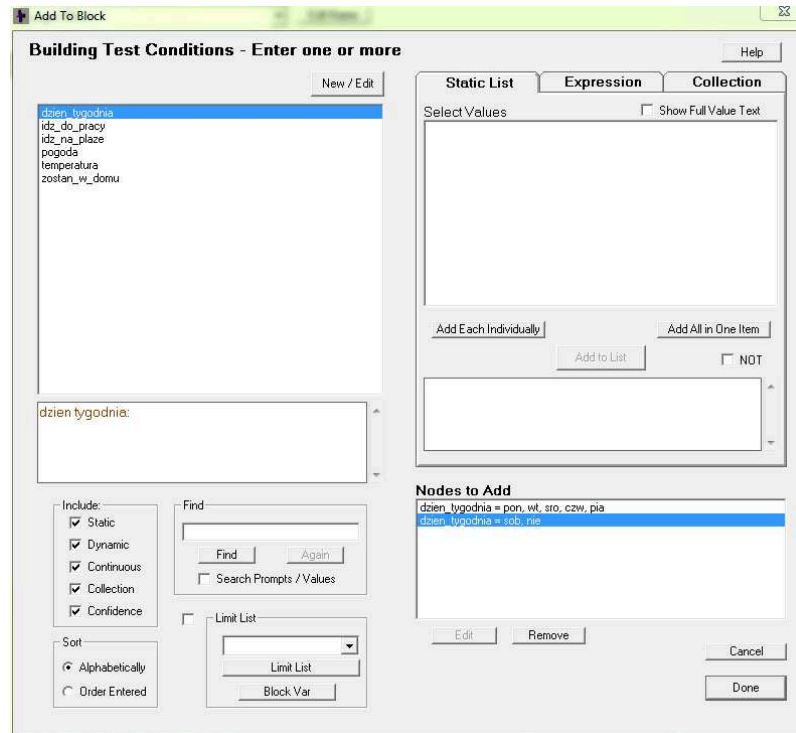
Najważniejsze składowe okna bloków logicznych:

1. nazwa wyświetlanego bloku logicznego,
2. obszar roboczy gdzie tworzona jest struktura reguły,
3. okno podglądu sworzonych reguł (po zamknięciu można je otworzyć poprzez menu Display->Rule View),
4. kontrolki do dodawania nowych węzłów.

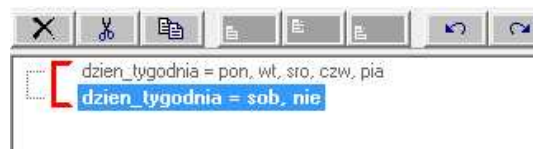
W poniższym zadaniu tworzymy blok funkcyjny o nazwie “co mam zrobić” poprzez modyfikację nazwy nadanej automatycznie **Logic block 1**.

Tworzenie reguł:

1. W pierwszej kolejności tworzymy reguły dotyczące czynności jaką jest pójście do pracy. Można utworzyć pięć oddzielnych reguł dla każdego dnia roboczego tygodnia (**if** dzisiaj jest poniedziałek **THEN** idź do pracy, **if** dzisiaj jest wtorek **THEN** idź do pracy itd), ale przejrzysiej jest zgrupować równorzędne wartości jakimi są dni tygodnia (**if** dzisiaj jest poniedziałek or wtorek or środa itd **THEN** idź do pracy). Bloki **if** i **THEN** wyglądają podobnie ale działanie jest zupełnie inne. W przypadku bloku **if** warunek “dzisiaj jest deszczowo” oznacza odpytanie użytkownika jaka jest dziś pogoda. Natomiast takie stwierdzenie w bloku **THEN** powoduje uznanie przez system, że pogoda jest deszczowa i ustawienie tej zmiennej na taką wartość. Tworzenie takiej reguły odbywa się poprzez obszar dodawania nowych węzłów. Aby dodać warunek **if** należy przycisnąć przycisk **Add**. W nowo otwartym oknie wyświetlane są zmienne dostępne w systemie. Należy wybrać “**Dzień_tygodnia**” i w zakładce **Static List** dodać nowy węzeł poprzez przycisk **Add to list** (Aby dodać węzeł złożony z kilku opcji należy wybierać je trzymając wciśnięty **CTRL**).
2. Następnie należy dodać kolejny węzeł odnoszący się do dni wolnych (Należy dodać oba węzły do jednej listy. W razie zamknięcia okna po dodaniu poprzedniego węzła należy go wybrać w głównym oknie roboczym i wybrać z menu podręcznego opcję **Edit** i powtórzyć dodawanie węzłów. W celu połączenia osobnych węzłów można tego dokonać przyciskiem **Group Together**

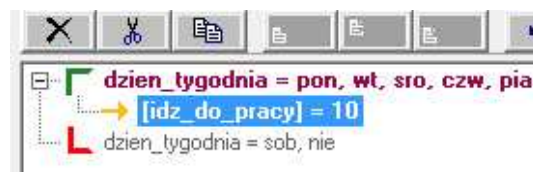


Po zatwierdzeniu w obszarze roboczym powinny pojawić się stworzone węzły:



W głównym oknie przedstawiona jest skrócona wersja węzłów natomiast w po wybraniu węzła w dolnym oknie "Node" będzie wyświetlony podgląd pełnego tekstu dotyczącego reguły.

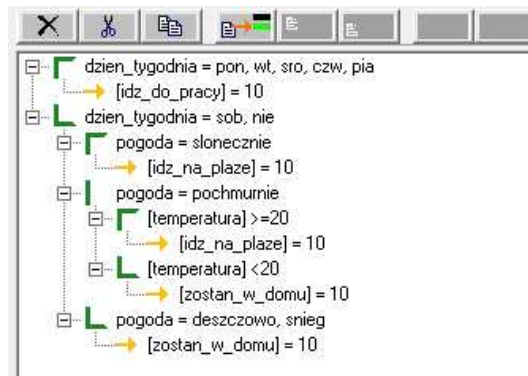
3. Następnie należy zbudować zbudować blok **THEN** w którym zostanie przypisana reguła pójścia do pracy w dzień roboczy. Aby stworzyć taką regułę należy zaznaczyć węzeł warunkowy odnoszący się do dni roboczych i następnie nacisnąć przycisk "Variable". W nowo otwartym oknie należy wybrać zmienną ufności, którą chcemy ustawić w tym bloku czyli **idź_do_pracy**. W zakładce **expression**, zmienna jest przedstawiona w nawiasach kwadratowych znaczy to, że do zmiennej będzie przypisywana wartość. Jaka wartość przedstawiająca ufnosć przy przypisywaniu do zmiennej zależy tylko i wyłącznie od projektanta systemu i jego implementacji. Należy jedynie pamiętać o strategii przypisywania wartości i trzymania się jej w całym systemie. W poniższym ćwiczeniu należy przypisać wartość 10 do zmiennej ([idź_do_pracy] = 10). Można dodać wiele elementów do bloku **THEN** czyniąc go dowolnie rozbudowanym zależnie od potrzeby ale w tym ćwiczeniu pozostaniemy na jednej instrukcji.



Dla tak stworzonej reguły można sprawdzić jej strukturę za pomocą okna **Rule view**.

Ponadto warto zwrócić uwagę na zielony wskaźnik przy pełnej regule (idź do pracy w dni robocze) i kolor czerwony wskaźnika przy regule niepełnej (co robić w dni wolne).

4. Kolejnym krokiem jest utworzenie reguł dla dni wolnych od pracy zgodnie z poniższym schematem (należy zwrócić uwagę na poziom zagnieżdżenia kolejnych instrukcji warunkowych). Należy pamiętać, że zagnieżdżanie instrukcji warunkowych powoduje wymóg spełnienia obu tych instrukcji (operator **AND**) aby blok **THEN** był zrealizowany.



W celu przetestowania systemu należy wcisnąć przycisk  i wybrać opcję zbudowania domyślnego bloku sterowania (Command Block - bloki sterowania będą dokładnie omówione na późniejszych zajęciach).

3 Zadanie do samodzielnego wykonania

Zbudować system ekspertowy dotyczący sterowania ogrzewaniem w pomieszczeniu.

Należy utworzyć następujące zmienne:

- zmienne numeryczne : temperatura żądana, temperatura w pomieszczeniu, temperatura na dworze.
- zmienne ufności: otwórz okno, włącz klimatyzację, włącz ogrzewanie.

System ma doradzać następujące czynności:

- otworzyć okno gdy temperatura zadana jest niższa niż temperatura w pomieszczeniu i wyższa niż temperatura na zewnątrz,
- otworzyć okno gdy temperatura zadana jest wyższa niż temperatura w pomieszczeniu i niższa niż temperatura na zewnątrz,
- włączyć klimatyzację gdy temperatura zadana jest niższa niż temperatura w pomieszczeniu i niższa niż temperatura na zewnątrz,
- włączyć ogrzewanie gdy temperatura zadana jest wyższa niż temperatura w pomieszczeniu i wyższa niż temperatura na zewnątrz.

Przetestować powstały system.