# CLIPS podstawy

CLIPS to prosty system ekspertowy oparty o język LIPS z cechami języka C i możliwością pisania dodatków/funkcji właśnie w języku C.

Poniżej umieszczono listę charakterystycznych poleceń języka CLIPS:

## Fakty

- (facts x) - wyświetla listę faktów począwszy od x
- (assert (jablko)) - ogólniej (assert (fakt1) (fakt2) (fakt3)) - dodaje nowy fakt
- (reset) - resetuje listę faktów
- (retract x) - usunięcie X'tego faktu, gdzie x to nr faktu
- (deffacts nazwa „opis" (fakt1) (fakt2) .. ) - definiuje listę faktów o określonej nazwie zawsze po wykonaniu polecenia (reset)
- (undeffacts nazwa) - cofie definicję określonych faktów
- (modify adres_faktu (nowa_wartość))
- (deftemplate nazwa (slot nazwa1) (slot nazwa2) (multislot nazwa3)) - pozwala na definiowanie schematu faktów (rekordu) który składa się z pól nazwa1, nazwa2, nazwa3, przy czym nazwa1 i nazwa2 mogą przyjmować tylko pojedynczą wartość, natomiast nazwa3 może przyjmować kilka wartości,
- (ppfact adres_faktu) - wyświetla tekst faktu
- (fact-index adres_faktu) – zwraca numer faktu
- (save-facts nazwa_pliku) - zapisanie faktów do pliku
- (load-facts nazwa-pliku) - wczytanie faktów z pliku

## Reguły

- (rules) - wyświetla listę reguł
- (defrule nazwa „komentarz" (przeslanka1) (przeslanka2) .. ⇒ (wniosek1) (wniosek2)) - definiowanie reguły przy czym przeslankaX i wniosekX to najczęściej fakty, w szczególności wniosekX to (assert) (retract)
- (ppdefrule nazwa) - umożliwia podgląd reguły
- (refresh) - ponowno sprawdzenie aktywacji reguły która została aktywowana. Normalnie reguły są aktywowane po zmianie stanu tabeli faktów, więc po jednokrotnej aktywacji reguły nie zostanie ona aktywowana więcej, chyba że wydamy komendę (refresh)
- (matchs nazwa) - pozwala podejrzeć stan reguły, tzn które wzorce są aktywowane
- (not (fakt)) - przy definiowaniu reguł możemy używać negacji typu (defrule xxx (not (zaplon tak)) ⇒ (assert (zepsute auto)))
- (or (fakt1) (fakt2) … )
- (and (fakt1) (fakt2) … )
- (test (typ fakt wartość)) - pozwala na realizację testów typu > < itp
- (save nazwa_pliku) - zapisuje reguły
- (load nazwa_pliku) - wczytuje reguły

## Ogólne

- (clear) - czyści całą listę faktów i reguł

- (watch co) - włącza podgląd np. (watch facts) (watch rules)(watch activations) (watch all) - podglądaj wszystko
- (unwatch) przerwij podgląd np. (unwatch all) – przerwij podgląd wszystkiego, szczegóły patrz wyżej
- (agenda) - wyświetla informacje o regule aktywowanej przez dane przesłanki
- (run) - uruchamia system
- ?nazwa - tworzy zmienną o nazwie nazwa, można też użyć samego ? wówczas jest to dowolna zmienna (nie mamy do niej dostępu bo nie znamy nazwy) która jest wymagana
- ?zmienna ← (fakt) - przepisanie adresu faktu do zmiennej zmienna (zmienna ta zawiera adres w postaci np. fact-0), zmienna jest zmienną lokalną więc dostęp do niej jest np. wewnątrz reguł
- $? - zmienna która nie jest wymagana
- (bind zmienna wartość) – przypisanie wartości do zmiennej .
- ~ - negacja
- | - „lub”/„or” w notacji clipsa
- (read) - wczytanie danych przez użytkownika
- (printout t „napis” crlf) – wypisanie wiersza tekstu na ekranie (terminalu „t”/konsoli)

## Do wykonania

Korzystając z powyżej określonych poleceń spróbuj rozwiązać poniższe zadania. W sprawozdaniu umieść odpowiednie polecenie niezbędne do rozwiązania poszczególnych zadań.

Zad 1)

Utwórz fakt (dzisiaj jest sroda) – *(assert)*, następnie sprawdź czy fakt został utworzony *(facts)*. Ostatecznie usuń utworzony fakt – *(retract)*. Spróbuj dodać inne fakty typu wczoraj był… jutro jest ….

Zad 2)

Zresetuj listę faktów *(reset).* Utwórz fakty będące inicjowane zawsze po restarcie systemu typu *(deffacts).* Sprawdź czy system działa poprawnie dodając jakieś fakty i resetując CLIPS’a

Zad 3)

Utwórz regułę korzystając z polecenia *(defrule)*, która sprawdza czy dzisiaj jest środa, jeśli reguła jest spełniona powinien pojawić się nowy fakt *(jutro jest czwartek)*

Zad 4)

Stwórz nową regułę w której przesłanką jest fakt (dzis jest sroda) wypisującą na ekranie komunikat *„Dziś jest środa”*

Zad 5)

Reguła stworzona w Zad 3 i 4 działaja tylko na dzien środa. Utwórz nową regułę „uniwersalną” wypisująca na ekranie komunikat dla dowolnego dnia tygodnia. W tym celu stwórz regułę wykorzystującą zmienne. Wykorzystując polecenie „defrule” utwórz regułę zastępując dzień tygodnia symbolem *?dzien*. Następnie w konkluzji wypisz na ekranie komunikat „Zmienna ma wartość” ?dzien.

Zad 6)

W celu usunięcia określonego faktu możesz się również posłużyć zmiennymi. W tym celu korzystając z operatora <- możesz przepisać adres faktu do zmiennej, tak aby później móc wykorzystać polecenie (retract zmienna), które usunie niechciana regułę. Ponieważ tworzone zmienne mają zasięg lokalny, więc polecenie to jest do wykorzystania wewnątrz reguł, np. chcąc usunąć z bazy faktów fakt odpowiedzialny za uruchomienie reguły możesz przy definicji reguły stworzyć zapis *„(…) ?zmienna<- (fakt) => (retract ?zmienna) (…)*" Opisz działanie takiej konstrukcji.

Zad 7)

Spróbuj wykorzystać polecenie (read), tak aby dwa komunikaty w konkluzji reguły następowały dopiero po naciśnięciu enter

Zad 8)

Stwórz system regułowy który będzie przechodził po wszystkich dniach tygodnia i wypisywał na ekranie komunikat „dzisiaj jest xxx", gdzie xxx to określony dzień.