Lab 4 - Weakest Precondition - if-else
Prepared by : Ms. K.P. Jevitha

Concepts:

- Weakest Preconditions

Tools Required:

- Alt-Ergo Theorem Prover
  Available online : https://alt-ergo.ocamlpro.com/try.html

Instructions:

1. Every question provides 3 components :
    a. Input condition - I
    b. Statement - S
    c. Post-condition - O
2. Steps to solve :
    a. Manually derive the weakest precondition for the given statement **S - wp(S,O)**
    b. For the given input condition I, using Alt-Ergo tool find whether **I ⇒ wp(S,O)**
3. If I ⇒ wp(S,O) is valid, show the rules of inference

Summary :

- To prove that a program P is correct with respect to its **contract** which is stated as a **pre-condition I** and **post-condition O**.
- The Weakest Precondition of a **statement S** w.r.t. a **post-condition O** is written as **wp(S, O)**.
- If the **input condition** for program P is **I**, then we want the following theorem to be true:
  I ==> wp(S, O)
- Weakest Preconditions to be done for the following code constructs :
    - **Assignment Statement S : wp(S, O)**
        - *wp (x = expr, O) = O [x← expr]* (replace all occurrences of x in O by expr.)
    - **Sequence of Statements S1;S2; : wp(S1 ; S2, O).**
        - ***wp (S1;S2; , O) = wp (S1, wp(S2,O))***
- if statement : wp(if (B) S1, O).
    - ***wp(if (B) S1 , O) = B ⇒ wp(S1,O) && not(B) ⇒ O (or) B && wp(S1,O) || not(B) && O***

      If part → wp(S1, O)

      Else part → O
    - ***wp(if (B) S1 , O) = (B && wp(S1,O)) || (not(B) && O)***

      If part → wp(S1, O)

      Else part → O

- **If-Else : wp(if (B) S1 else S2, O).**
  - *wp(if (B) S1 else S2, O) = B* ***⇒ wp(S1,O) && not(B) ⇒ wp(S2,O)***

    If part → wp(S1, O)
    Else part → wp(S2,O)

  - ***wp(if (B) S1 else S2, O) = (B && wp(S1,O)) || (not(B) && wp(S2,O))***

    If part → wp(S1, O)
    Else part → wp(S2,O)
- **Else-If:**

```
If(B1)
   S1;
else if(B2)
   S2;
else if(B3)
  S3;
..
Else
    Sn;
```

  - ***wp(if (B1) S1*** **else if(B2)** ***S2*** **else if(B3) S3 …. else Sn** ***, O) =***

    B1 && wp(S1,O)
    || not(B1) && B2 && wp(S2,O)
    || not(B1) && not(B2) && B3 && wp(S3,O)
    ..
    ..
    ..
    not(B1 || B2 || … || Bn-1) && wp(Sn,O)

  - ***wp(if (B1) S1*** **else if(B2)** ***S2*** **else if(B3) S3 …. else Sn** ***, O) =***

    B1 && wp(S1,O)
    || not(B1) && B2 && wp(S2,O)
    || not(B1 || B2) && B3 && wp(S3,O)
    ..
    ..
    ..
    not(B1 || B2 || … || Bn-1) && wp(Sn,O)

Examples

Find the weakest precondition for the given problems by assuming appropriate input and output conditions (3 each for every problem) and perform the validity check using alt-ergo.

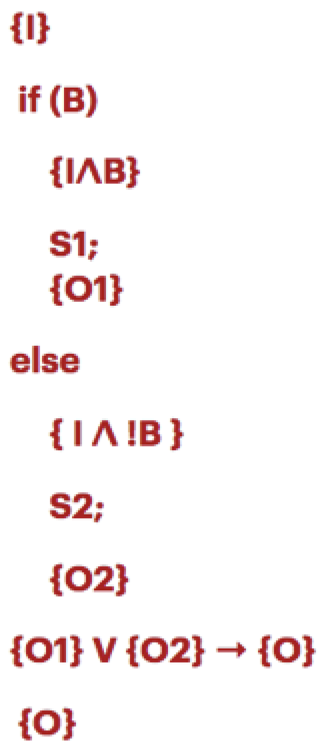
Example 1 : Write a program to find the maximum between two numbers. Write the output condition for max and find the WP.

Program:

```
if(a > b)
  S1: max = a
else
  S2: max = b
```

Output condition: max = max(a,b)
(max =a /\ a>b) \/ (max=b /\ a<=b)

Reasoning about if-else



```
{I}
 if (B)
    {I∧B}
    S1;
    {O1}
else
    { I ∧ !B }
    S2;
    {O2}
{O1} V {O2} → {O}
 {O}
```

Example – computing max of (x,y)
{true}
if (x>y)
{true /\ x>y} —> {x>y}

m=x
{O1: m=x ∧ x>y }
else
{ true ∧ x<=y} —> {x<=y}
m=y
{O2: m = y ∧ x <= y}

{O1} V {O2} → {O}
{O1 V O2} = { (m = x Λ x > y) V (m = y Λ x <= y) } → {m=max(x,y)}={O}

O1: max > 10
O2: (max = a or max =b) and max > 50

Weakest Precondition
If-else : WP - (B && wp(S1,O)) || (~B && wp(S2,O)
B: a>b
wp(S1,O) ⇒ [max > 10] {max=a} ⇒ **a** > **10**
wp(S2,O) ⇒ [max > 10] {max=b} ⇒ **b** > **10**

[(a>b) && wp(max=a,O)] || [(a<=b) && wp(max = b,O)]
⇒ (a>b) && a > 10) || (a<=b) && (b > 10)) → **Required Weakest precondition for O1**

Alt-ergo: I → wp(if-else, O)
goal a1:
forall a,b,max: int.
I1 : (a=3 and b=11) → ((a>b) and a > 10) or ((a<=b) and (b > 10)) - **Valid**

I2: (a=3 and b=4) -> ((a>b) and a > 10) or ((a<=b) and (b > 10)) - **unknown**

Example 2 : Given the following program, write the function to find a minimum of two numbers and find the WP. Assume input conditions and verify I→ WP in alt-ergo

main(){
W = 2*w

```
Z = -w
Y = V+1
x= min(y,z)
}

min(y,z){
if(y < z)
  S1: min = y
else
  S2: min = z
}
```

O1: Min < 0

Weakest Precondition

If-else : WP - (B && wp(S1,O)) || (~B && wp(S2,O)

B: y < z

wp(S1,O) ⇒ [min < 0] {min = y} ⇒ **y < 0**

wp(S2,O) ⇒ [min < 0] {min= z} ⇒ **z < 0**

Min function wp → [y < z && y<0] || [y>=z && z<0]

Y = V+1

[v+1 < z && v+1<0] || [v+1>=z && z<0]

Z = -w

[v+1 < -w && v+1<0] || [v+1>=-w && -w<0]

W = 2*w

[v+1 < -2w && **v+1 < 0**] || [(v+1>= -2w) && **-2w < 0**]

Assuming the min function is not defined:

O2: x < 0

WP [x<0] {x=min(y,z)} →

{min(y,z) < 0 } [x = min(y,z)]

min(v+1,z) <0 [y = v+1]

min(v+1,-w)<0 [z = -w]

min(v+1,-2*w)<0 [w = 2*w]

WP → [2w+v+1<0 && v < –1] || [2w+v+1 >= 0 && w > 0]

Exercises

Find the weakest precondition for the given problems by assuming appropriate input and output conditions (2 each for every problem) and perform the validity check using alt-ergo.

1) Find the maximum between three numbers.
2) Check whether a number is negative, positive or zero.
3) Check whether a number is even or odd.
4) Input week number and print week day.
5) Input the basic salary of an employee and calculate its Gross salary according to following:
    a) Basic Salary <= 10000 : HRA = 20%, DA = 80%
    b) Basic Salary <= 20000 : HRA = 25%, DA = 90%
    c) Basic Salary > 20000 : HRA = 30%, DA = 95%