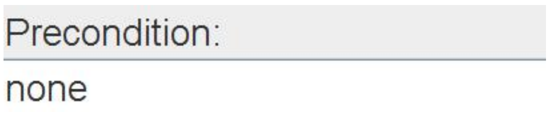
# Raptor – Input Guide

**Please Note:** the guide has been produced with the intent to show users the syntax of the inputs required to use Raptor.

## PRECONDITION

You must type a formula that represents the precondition to the program that will be specified in the Input section for the Program. Additionally, you may prefix the precondition with declarations for the program variables you will be using, of the form: “int var;” (note the semi-colon). Declarations must come before the precondition and you must declare the variables involved in the precondition before specifying the precondition.

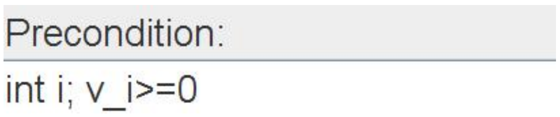
Examples:



Precondition:

none

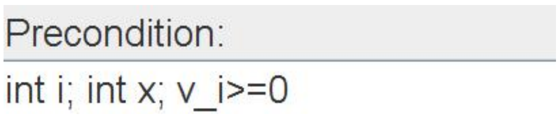
If you have that your program does not have any precondition, use none (as shown above), top or true to express this.



Precondition:

int i; v_i>=0

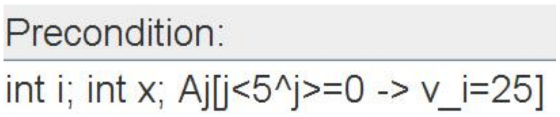
Any variables that are used in the precondition must be declared before the precondition.



Precondition:

int i; int x; v_i>=0

You are permitted to declare additional variables that are used only in the program code. This is particularly relevant when you have initiated a proof for a method that has been imported into your proof as you may have parameters to the method which the precondition is not concerned with.



Precondition:

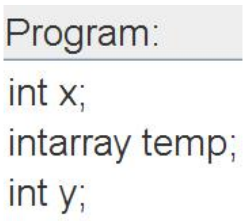
int i; int x; Aj[j<5^j>=0 -> v_i=25]

This example shows how to use quantifiers in your precondition.

**PROGRAM**

You must input the program for which you would like to reason. There are various items that can be typed as input for a program in Raptor:

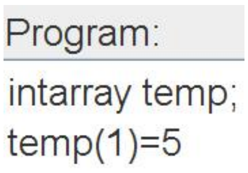
Examples:



Program:

```
int x;
intarray temp;
int y;
```

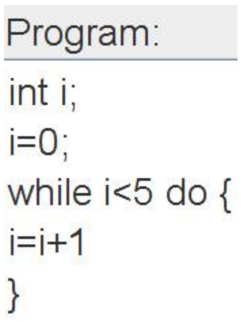
All declarations for variables must be present before the code which uses them.



Program:

```
intarray temp;
temp(1)=5
```

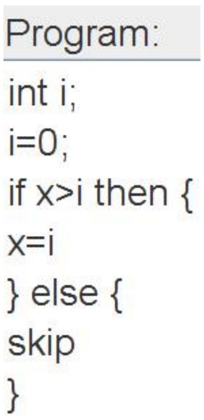
This example shows how to form an assignment statement involving arrays. Please note the use of round brackets to access an array element.



Program:

```
int i;
i=0;
while i<5 do {
i=i+1
}
```

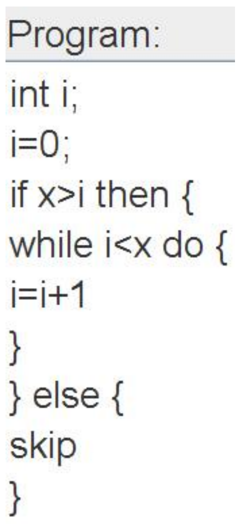
The construct of a while loop is shown above. Please note that the last program line in each block of code (the line “i=i+1” in the example shown) must not have a semi-colon at the end of it, else Raptor will not accept your input.



Program:

```
int i;
i=0;
if x>i then {
x=i
} else {
skip
}
```

This example shows the use of an if-statement.



Program:

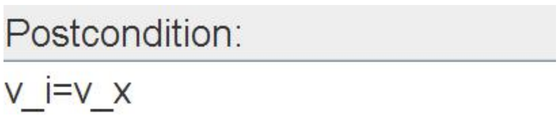
```
int i;
i=0;
if x>i then {
while i<x do {
i=i+1
}
} else {
skip
}
```

An example of how to nest a while loop in an if-statement. Again, note that in the various bodies of code, the last program line does not end with a semi-colon.

**POSTCONDITION:**

Input is similar to that of the precondition except that no declarations are permitted now.

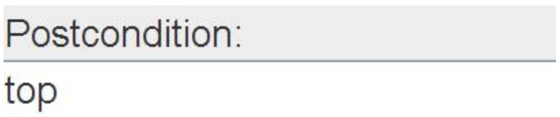
Examples:



Postcondition:

v_i=v_x

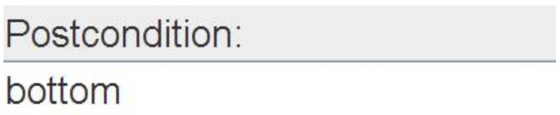
Above is an example displaying equality in the post condition.



Postcondition:

top

This is an example of how to input top (⊤) in the post condition.  Alternatively, you may use true instead of top.



Postcondition:

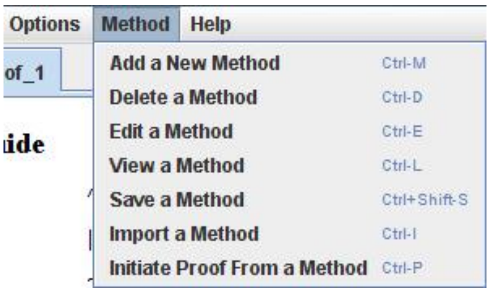
bottom

This is an example of how to input bottom (⊥) in the post condition.

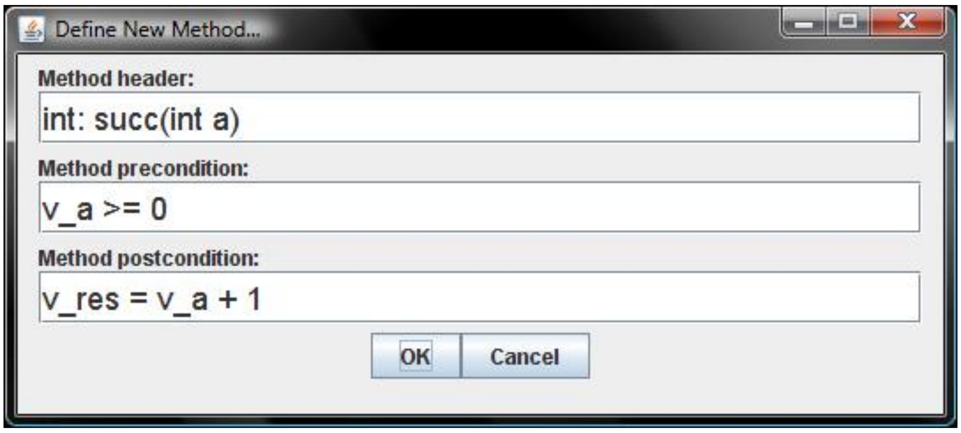
## DECLARING A METHOD:

To be able to use method calls in your program code you must provide the method header, precondition and post condition for that method.

Examples:



This is the menu available to you to manipulate the methods that you would like to make use of in the proof that you have open. Each proof has its own set of included methods, if you open another proof, you must import the methods you would like to use into this proof.
You can use “Initiate Proof From a Method” to launch a proof from a method definition.

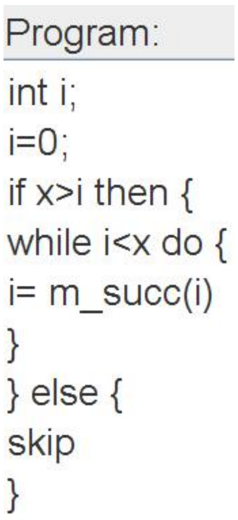


Method header: int: succ(int a)

Method precondition: v_a >= 0

Method postcondition: v_res = v_a + 1

When defining a method you must define its return type, name and parameters in the fashion shown above. The precondition may only refer to the parameters that can be passed to the method, and no others. The post condition must only refer to the input variables and the reserved variable “res” in terms of the return type, e.g. for the return type int the post condition must make reference to “v_res”.


Program:

```
int i;
i=0;
if x>i then {
while i<x do {
i= m_succ(i)
}
} else {
skip
}
```

Calls to declared methods must be prefixed with “m_”. Note: if you make a call to a method that you have not added to your proof, Raptor will not accept the input.