Frama-C (Program Reasoning – 19CSE205)

1) **Write the postcondtion and precondition for the following**

```
int add(int x, int y) {
    return x+y ;
}
```

CODING:

```
#include<limits.h>
/*@
    requires INT_MIN<a+b<INT_MAX;
    ensures \result==a+b;
*/
int addition(int a,int b)
{
   return a+b;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 % frama-c -wp -wp-rte addition.c
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing addition.c (with preprocessing)
[rte] annotating function addition
[wp] 3 goals scheduled
[wp] Proved goals:    3 / 3
     Qed:             1
     Alt-Ergo:        2  (7ms) (7)
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 %
```

2) **Write the postcondtion and precondition for the following**

```
int max(int a, int b){
    return (a > b)?a:b;
}
```

CODING:

```
#include<limits.h>
/*@
  requires a<INT_MAX && b<INT_MAX;
  requires a>INT_MIN && b>INT_MIN;
  ensures \result==\max(a,b);
*/
int max(int a,int b)
{
   return a>b?a:b;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 % frama-c -wp -wp-rte max.c
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing max.c (with preprocessing)
[rte] annotating function max
[wp] 1 goal scheduled
[wp] Proved goals:    1 / 1
     Qed:             1  (0.65ms)
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 %
```

3) **Write the postcondtion and precondition for the following**

```
int distance(int a, int b){
     if(a < b) return b - a;
    else return a - b;
}
```

CODING:

```
#include<limits.h>
/*@
  requires INT_MIN<b-a<INT_MAX;
  ensures \result>=0;
  ensures (a<=b ==> \result==b-a) &&
        (b<a ==> \result==a-b);
*/
int distance(int a,int b)
{
   if(a<b)
     return b-a;
   else
     return a-b;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 % frama-c -wp -wp-rte distance.c
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing distance.c (with preprocessing)
[rte] annotating function distance
[wp] 6 goals scheduled
[wp] Proved goals:    6 / 6
     Qed:             2  (0.13ms-0.49ms-1ms)
     Alt-Ergo:        4  (6ms-7ms) (8)
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 %
```

4) Write the postcondtion and precondition for the following (USING BEHAVIORS)

```
int distance(int a, int b){
     if(a < b) return b - a;
    else return a - b;
}
```

CODING:

```
#include<limits.h>
/*@
  requires INT_MIN<b-a<INT_MAX;
  behavior one:
        assumes a<b;
        ensures \result==b-a;
  behavior two:
        assumes b<a;
        ensures \result==a-b;
  behavior three:
        assumes b==a;
        ensures \result==0;
*/
int distance(int a,int b)
{
  if(a<b)
    return b-a;
  else
    return a-b;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 % frama-c -wp -wp-rte distance2.c
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing distance2.c (with preprocessing)
[rte] annotating function distance
[wp] 7 goals scheduled
[wp] Proved goals:    7 / 7
     Qed:             4  (0.15ms-0.88ms-3ms)
     Alt-Ergo:        3  (7ms) (7)
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 %
```

5) **Write the postcondtion and precondition for the following**

void div_rem(unsigned x, unsigned y, unsigned *q, unsigned *r){
***q = x / y ;**
***r = x % y ;**
}

CODING:

```
#include<limits.h>
/*@
  requires INT_MIN<x/y<INT_MAX && y!=0;
  requires INT_MIN<x%y<INT_MAX && y!=0;
  requires \separated(q,r);
  requires \valid(q) && \valid(r);
  ensures *q==x/y;
  ensures *q>=0;
  ensures *r==x%y;
  ensures *r>=0;
*/
void div_rem(unsigned x,unsigned y,unsigned *q,unsigned *r)
{
  *q=x/y;
  *r=x%y;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 % frama-c -wp -wp-rte div.c
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing div.c (with preprocessing)
[rte] annotating function div_rem
[wp] 8 goals scheduled
[wp] Proved goals:    8 / 8
     Qed:             5  (0.19ms-0.76ms-2ms)
     Alt-Ergo:        3  (12ms-17ms) (32)
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 %
```

6) **Write the postcondtion and precondition for the following**

```
int abs(int val){
    if(val < 0) return -val;
    return val;
}
```

CODING:

```
#include <limits.h>
/*@
   requires INT_MIN < val;

   ensures \result >= 0;
   ensures (val >= 0 ==> \result == val) &&
         (val < 0 ==> \result == -val);
*/
int abs(int val)
{
   if(val<0)
       return -val;
   return val;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte abs.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing abs.c (with preprocessing)
[rte] annotating function abs
[wp] 3 goals scheduled
[wp] Proved goals:    3 / 3
     Qed:             3
kumarlaxmikant@Kumars-MacBook-Pro practice % _
```

7) **Write the postcondtion and precondition for the following (USING BEHAVIORS)**

```
int abs(int val){
    if(val < 0) return -val;
    return val;
}
```

CODING:

```
#include<limits.h>
/*@
  requires val>INT_MIN;
  assigns \nothing;
  ensures \result>=0;
  behavior one:
        assumes 0<=val;
        ensures \result==val;
  behavior two:
        assumes val<0;
        ensures \result==-val;
  complete behaviors;
  disjoint behaviors;
*/
int abs(int val)
{
  if(val<0)
    return -val;
  else
    return val;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 % frama-c -wp -wp-rte abs.c
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing abs.c (with preprocessing)
[rte] annotating function abs
[wp] 8 goals scheduled
[wp] Proved goals:    8 / 8
     Qed:             8
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 %
```

8) Write the postcondtion and precondition for the following

```
void max_ptr(int* a, int* b){
    if(*a < *b){
        int tmp = *b ;
        *b = *a ;
        *a = tmp ;
    }
}
```

CODING (VERSION 1):

```
/*@
   requires \valid(a) && \valid(b);
   assigns  *a, *b ;
   ensures  \old(*a) < \old(*b)  ==> *a == \old(*b) && *b == \old(*a) ;
   ensures  \old(*a) >= \old(*b) ==> *a == \old(*a) && *b == \old(*b) ;
*/
void max_ptr(int *a,int *b) {
   if(*a<*b) {
      int temp=*b;
      *b=*a;
      *a=temp;
   }
 }
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte max_ptr.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing max_ptr.c (with preprocessing)
[rte] annotating function max_ptr
[wp] 11 goals scheduled
[wp] Proved goals:   11 / 11
     Qed:             8  (0.16ms-0.43ms-1ms)
     Alt-Ergo:        3  (10ms-14ms) (18)
kumarlaxmikant@Kumars-MacBook-Pro practice %
```

CODING (VERSION 2):

```
/*@
    requires \valid(a) && \valid(b);
    behavior one:
            assumes *a<*b;
            ensures A:*\old(a)==\old(*b);
            ensures B:*\old(b)==\old(*a);
            assigns *a,*b;
*/
void max_ptr(int *a,int *b) {
   if(*a<*b) {
      int temp=*b;
      *b=*a;
      *a=temp;
   }
 }
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 % frama-c -wp -wp-rte max_ptr.c
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing max_ptr.c (with preprocessing)
[rte] annotating function max_ptr
[wp] 11 goals scheduled
[wp] Proved goals:   11 / 11
     Qed:             8  (0.10ms-0.89ms-3ms)
     Alt-Ergo:        3  (8ms-9ms-12ms) (18)
kumarlaxmikant@Kumars-MacBook-Pro 17sep2021 %
```

9) Write the postcondtion and precondition for the function that returns the Quadrant for a given point.

CODING:

```
#include<limits.h>
/*@
   requires INT_MIN < x < INT_MAX && x!=0;
   requires INT_MIN < y < INT_MAX && y!=0;
   behavior first:
      assumes x>0 && y>0;
      ensures \result == 1;
   behavior second:
      assumes x<0 && y>0;
      ensures \result == 2;
   behavior third:
      assumes x<0 && y<0;
      ensures \result == 3;
   behavior fourth:
      assumes x>0 && y<0;
      ensures \result == 4;
   disjoint behaviors first,second,third,fourth;
   complete behaviors;
*/
int quadrant(int x,int y)
{
   if(x>0 && y>0)
      return 1;
   else if(x<0 && y>0)
      return 2;
   else if(x<0 && y<0)
      return 3;
   else if(x>0 && y<0)
      return 4;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte quadrant.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing quadrant.c (with preprocessing)
quadrant.c:26:[kernel] warning: Body of function quadrant falls-through. Adding a return statement
[rte] annotating function quadrant
[wp] 7 goals scheduled
[wp] Proved goals:    7 / 7
     Qed:             4  (0.31ms-1ms-5ms)
     Alt-Ergo:        3  (8ms-9ms) (12)
kumarlaxmikant@Kumars-MacBook-Pro practice % _
```

10) Write the postcondtion and precondition for the function that implements Bubble-Sort.

CODING:

```
#include<limits.h>
/*@
   requires 0<n<INT_MAX;
   requires \valid(arr+(0..n-1));
   ensures \forall integer i;
      0<=i<n-1 ==> arr[n-1]>=arr[i];
*/
void BubbleSort(int arr[],int n)
{
   /*@
      loop invariant \forall integer x;
        0<=x<=i ==> arr[x]<=arr[i];
      loop invariant 0<=i<=n-1;
      loop assigns i,arr[0..n-1];
      loop variant n-1-i;
   */
   for(int i=0;i<n-1;i++)
   {
      if(arr[i]>arr[i+1])
      {
         int temp = arr[i];
         arr[i] = arr[i+1];
```

```
            arr[i+1] = temp;
          }
        }
      }
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte bubbleSort.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing bubbleSort.c (with preprocessing)
[rte] annotating function BubbleSort
[wp] 23 goals scheduled
[wp] Proved goals:   23 / 23
     Qed:            11  (0.14ms-2ms-8ms)
     Alt-Ergo:       12  (11ms-24ms-57ms) (135)
kumarlaxmikant@Kumars-MacBook-Pro practice % _
```

11) Write the postcondtion and precondition for the function that implements Selection-Sort.

CODING:

```
#include<limits.h>
/*@
   requires 0<n<INT_MAX;
   requires \valid(arr+(0..n-1));
   ensures \forall integer i,j;
     0<=j<i<n ==> arr[j]<=arr[i];
*/
void SelectionSort(int arr[],int n)
{
   int i,j,temp;
   /*@
      loop invariant \forall integer p,q;
        0<=p<=q<n ==> arr[p]<=arr[q];
      loop invariant 0<=i<=n;
      loop assigns temp,i,j,arr[0..n-1];
      loop variant n-i;
   */
```

```
    for(i=0;i<n;i++)
    {
      /*@
        loop invariant \forall integer p;
          i+1<=p<n ==> arr[p]<=arr[n];
        loop invariant i+1<=j<=n;
        loop assigns temp,j,arr[0..n-1];
        loop variant n-j;
      */
      for(j=i+1;j<n;j++)
      {
        if(arr[i]>arr[j])
        {
          temp = arr[i];
          arr[i] = arr[j];
          arr[j] = arr[i];
        }
      }
    }
  }
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro assignment_3 % frama-c -wp -wp-rte SelectionSort.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing SelectionSort.c (with preprocessing)
[rte] annotating function SelectionSort
[wp] 30 goals scheduled
[wp] [Alt-Ergo] Goal typed_SelectionSort_loop_inv_established : Unknown (63ms)
[wp] [Alt-Ergo] Goal typed_SelectionSort_loop_inv_preserved : Unknown (Qed:2ms) (344ms)
[wp] [Alt-Ergo] Goal typed_SelectionSort_loop_inv_3_established : Unknown (Qed:0.55ms) (114ms)
[wp] Proved goals:   27 / 30
     Qed:            13  (0.23ms-3ms-7ms)
     Alt-Ergo:       14  (9ms-25ms-62ms) (134) (unknown: 3)
kumarlaxmikant@Kumars-MacBook-Pro assignment_3 % _
```

12) Write the postcondtion and precondition for the function that resets all Array Elements to ZERO.

CODING:

```
#include<limits.h>
/*@
   requires 0<n<INT_MAX;
   requires \valid(arr+(0..n-1));
   ensures \forall integer i;
     0<=i<n ==> arr[i]==0;
*/
void ResetToZero(int arr[],int n)
{
   /*@
      loop invariant \forall integer x;
        0<=x<i ==> arr[x]==0;
      loop invariant 0<=i<=n;
      loop assigns i,arr[0..i-1];
      loop variant n-i;
   */
   for(int i=0;i<n;i++)
   {
      arr[i]=0;
   }
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro 29oct2021 % frama-c -wp -wp-rte resetArrayToZero.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing resetArrayToZero.c (with preprocessing)
[rte] annotating function ResetToZero
[wp] 12 goals scheduled
[wp] Proved goals:   12 / 12
     Qed:             5  (0.08ms-0.91ms-3ms)
     Alt-Ergo:        7  (10ms-17ms-28ms) (69)
kumarlaxmikant@Kumars-MacBook-Pro 29oct2021 % _
```

13) Write the postcondtion and precondition for the function that Compares the length of Two Array (If Equal Or Not).

CODING:

```
/*@
   requires \valid_read(arr1+(0..sizeof(arr1)-1)) &&
\valid_read(arr2+(0..sizeof(arr2)-1));
   behavior Length_NotEqual:
      assumes (sizeof(arr1)/sizeof(arr1[0]))!=(sizeof(arr2)/sizeof(arr2[0]));
      ensures \result==0;
   behavior Length_Equal:
      assumes (sizeof(arr1)/sizeof(arr1[0]))==(sizeof(arr2)/sizeof(arr2[0]));
      ensures \result==1;
   complete behaviors;
   disjoint behaviors Length_NotEqual,Length_Equal;
*/
int CompareLength(int arr1[],int arr2[])
{
   int len1 = sizeof(arr1)/sizeof(arr1[0]);
   int len2 = sizeof(arr2)/sizeof(arr2[0]);
   if(len1==len2)
      return 1;
   else
      return 0;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro assignment_3 % frama-c -wp -wp-rte Compare_Array_Length.c;
                        [kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing Compare_Array_Length.c (with preprocessing)
[rte] annotating function CompareLength
[wp] 4 goals scheduled
[wp] Proved goals:    4 / 4
     Qed:             4
kumarlaxmikant@Kumars-MacBook-Pro assignment_3 %
```

14) Write the postcondtion and precondition for the function that checks if all Array Elements are equal to its Index Value. (Eg. Arr[2]==2)

CODING:

```
#include<limits.h>
/*@
   requires 0<n<INT_MAX;
   requires \valid_read(arr+(0..n-1));
   behavior equal:
     assumes \forall integer i;
         0<=i<n ==> arr[i]==i;
      ensures \result==1;
   behavior not_equal:
     assumes \exists integer i;
         0<=i<n && arr[i]!=i;
      ensures \result==0;
   complete behaviors;
   disjoint behaviors equal,not_equal;
*/
int ElementEqualIndex(int arr[],int n)
{
   int i=0;
   /*@
      loop invariant \forall integer j;
         0<=j<i ==> arr[j]==j;
      loop invariant 0<=i<=n;
      loop assigns i;
      loop variant n-i;
   */
   for(i=0;i<n;i++)
   {
      if(arr[i]!=i)
      {
         return 0;
```

```
    }
  }
  return 1;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro assignment_3 % frama-c -wp -wp-rte Element_Equal_To_Index.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing Element_Equal_To_Index.c (with preprocessing)
[rte] annotating function ElementEqualIndex
[wp] 13 goals scheduled
[wp] Proved goals:   13 / 13
     Qed:             5  (0.24ms-1ms-4ms)
     Alt-Ergo:        8  (9ms-13ms-21ms) (47)
kumarlaxmikant@Kumars-MacBook-Pro assignment_3 % _
```

15) Write the postcondtion and precondition for the function that checks if a character is Alphabet or NOT.

CODING:

```
/*@
  requires 0<=c<=255;
  behavior Alphabet:
     assumes 97<=c<=122 || 65<=c<=90;
     ensures \result==1;
  behavior NotAlphabet:
     assumes 0<=c<65 || 90<c<97 || 122<c<=255;
     ensures \result==0;
  complete behaviors Alphabet,NotAlphabet;
  disjoint behaviors;
*/
int alphabet_letter(char c) {
   if(('a'<=c && c<='z')||('A'<=c && c<='Z'))
        return 1;
   else
        return 0;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte alphabet_letter.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing alphabet_letter.c (with preprocessing)
[rte] annotating function alphabet_letter
[wp] 4 goals scheduled
[wp] Proved goals:    4 / 4
     Qed:             1  (0.14ms-1ms)
     Alt-Ergo:        3  (8ms-9ms) (12)
kumarlaxmikant@Kumars-MacBook-Pro practice % _
```

16) This function receives two values of angle in input and returns the value of the last angleconsidering that the sum of the three angles must be 180. Write the postcondition that expressesthat the sum of the three angles is 180.

```
int last_angle(int first,int second)
{
    return 180- first- second ;
}
```

CODING:

```
/*@
   requires 0<=first<=180;
   requires 0<=second<=180;
   ensures \result==180-first-second;
*/
int last_angle(int first,int second)
{
   return 180 - first - second;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte last_angle.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing last_angle.c (with preprocessing)
[rte] annotating function last_angle
[wp] 4 goals scheduled
[wp] Proved goals:    4 / 4
     Qed:             1
     Alt-Ergo:        3  (7ms) (9)
kumarlaxmikant@Kumars-MacBook-Pro practice % _
```

17) The following function should check the order of 3 input values in increasing order. Write the corresponding code and specification of the function:

```
void order_3(int* a,int* b,int* c)
{
    // CODE
}
```

CODING:

```
#include<limits.h>
/*@
  requires \valid(a) && \valid(b) && \valid(c);
  requires \separated(a,b,c);
  behavior TRUE:
    assumes (*a)<=(*b) && (*b)<=(*c);
    ensures \result == 1;
    assigns \nothing;
  behavior FALSE:
    assumes !((*a)<=(*b) && (*b)<=(*c));
    ensures \result == 0;
    assigns \nothing;
  disjoint behaviors TRUE,FALSE;
  complete behaviors;
*/
int order_3(int *a,int *b,int *c)
{
  if((*a)<=(*b) && (*b)<=(*c))
  {
    return 1;
  }
  else
  {
    return 0;
  }
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte check_pointer_order3.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing check_pointer_order3.c (with preprocessing)
[rte] annotating function order_3
[wp] 12 goals scheduled
[wp] Proved goals:   12 / 12
     Qed:             7  (0.29ms-0.92ms-3ms)
     Alt-Ergo:        5  (9ms-11ms-15ms) (24)
kumarlaxmikant@Kumars-MacBook-Pro practice % _
```

18) Specify the function leap year that returns true if the year received as an input is leap. Write the corresponding code and specification of the function:

```
int leap(int y)
{
    return((y%4==0) && (y %100!=0)) || (y %400==0) ;
}
```

CODING:

```
#include<limits.h>
/*@
   requires INT_MIN < y < INT_MAX;
   behavior LEAP:
      assumes ((y%4==0) && (y%100!=0)) || (y%400==0);
      ensures \result == 1;
   behavior NOT_LEAP:
      assumes !(((y%4==0) && (y%100!=0)) || (y%400==0));
      ensures \result == 0;
   disjoint behaviors LEAP,NOT_LEAP;
   complete behaviors;
*/
int leap(int y)
{
   return ((y%4==0) && (y%100!=0)) || (y%400==0) ;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte leap_year.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing leap_year.c (with preprocessing)
[rte] annotating function leap
[wp] 4 goals scheduled
[wp] Proved goals:    4 / 4
     Qed:             2  (0.09ms-0.49ms-1ms)
     Alt-Ergo:        2  (7ms) (9)
kumarlaxmikant@Kumars-MacBook-Pro practice % _
```

19) Write the contract of the function:

```
int plus_5(int *a)
{
   return *a+5;
}
```

CODING:

```
#include<limits.h>
/*@
   requires \valid_read(a);
   requires *a<=INT_MAX-5;
   assigns \nothing;
   ensures \result == *a+5;
*/
int plus_5(int *a)
{
   return *a+5;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte plus_5.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing plus_5.c (with preprocessing)
[rte] annotating function plus_5
[wp] 4 goals scheduled
[wp] Proved goals:    4 / 4
     Qed:             4
kumarlaxmikant@Kumars-MacBook-Pro practice % _
```

20) Write the contract of the function that swaps two pointer values:

```
void swap(int*a,int*b)
{
    int tmp= *a;
    *a= *b;
    *b=tmp;
    return;
}
```

CODING:

```
/*@
   requires \valid(a) && \valid(b);
   ensures A:*\old(a)==\old(*b);
   ensures B:*\old(b)==\old(*a);
   assigns *a,*b;
*/
void swap(int*a,int*b)
{
    int tmp= *a;
    *a= *b;
    *b=tmp;
    return;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte swap.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing swap.c (with preprocessing)
[rte] annotating function swap
[wp] 9 goals scheduled
[wp] Proved goals:    9 / 9
     Qed:             6  (0.13ms-0.44ms-1ms)
     Alt-Ergo:        3  (6ms-13ms) (17)
kumarlaxmikant@Kumars-MacBook-Pro practice %
```

21) Write function definition and contract to find maximum element of an array.

CODING:

```
#include<limits.h>
/*@
   requires 0<n<INT_MAX;
   requires \valid_read(arr+(0..n-1));
   ensures \forall integer i;
      0<=i<n ==> \result>=arr[i];
*/
int Max_in_Array(int arr[],int n)
{
   int max=arr[0];
   /*@
      loop invariant \forall integer i;
      0<=i<j ==> max>=arr[i];
      loop invariant 0<=j<=n;
      loop assigns j,max;
      loop variant n-j;
   */
   for(int j=0;j<n;j++)
   {
      if(arr[j]>max)
      {
         max=arr[j];
      }
   }
   return max;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte max_in_array.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing max_in_array.c (with preprocessing)
[rte] annotating function Max_in_Array
[wp] 12 goals scheduled
[wp] Proved goals:   12 / 12
     Qed:             6  (0.11ms-3ms)
     Alt-Ergo:        6  (9ms-14ms-27ms) (45)
kumarlaxmikant@Kumars-MacBook-Pro practice % _
```

22) Write function definition and contract to find minimum element of an array.

CODING:

```
#include<limits.h>
/*@
   requires 0<n<INT_MAX;
   requires \valid_read(arr+(0..n-1));
   ensures \forall integer i;
     0<=i<n ==> \result<=arr[i];
*/
int min_in_array(int arr[],int n)
{
   int min=arr[0];
   /*@
      loop invariant \forall integer x;
         0<=x<i ==> min<=arr[x];
      loop invariant 0<=i<=n;
      loop assigns i,min;
      loop variant n-i;
   */
   for(int i=0;i<n;i++)
   {
      if(arr[i]<min)
      {
         min=arr[i];
      }
   }
   return min;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte min_in_array.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing min_in_array.c (with preprocessing)
[rte] annotating function min_in_array
[wp] 12 goals scheduled
[wp] Proved goals:   12 / 12
     Qed:             6  (0.11ms-2ms-4ms)
     Alt-Ergo:        6  (10ms-17ms-39ms) (45)
kumarlaxmikant@Kumars-MacBook-Pro practice % _
```

23) Write a contract for approximation of cosine function.

CODING:

```
/*@ requires \abs(\exact(x)) <= 0x1p-5;
  @ requires \round_error(x) <= 0x1p-20;
  @ ensures \abs(\exact(\result) - \cos(\exact(x))) <= 0x1p-24;
  @ ensures \round_error(\result) <= \round_error(x) + 0x3p-24;
  @
*/
float cosine(float x)
{
   return 1.0f - x * x * 0.5f;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte cosine.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing cosine.c (with preprocessing)
[rte] annotating function cosine
[wp] user error: Builtin \round_error(float32) not defined
[wp] failure: Logic '\round_error' undefined
[kernel] Current source was: cosine.c:9
        The full backtrace is:
        Raised at file "src/kernel_services/plugin_entry_points/log.ml", line 583, characters 30-31
        Called from file "src/kernel_services/plugin_entry_points/log.ml", line 577, characters 9-16
        Re-raised at file "src/kernel_services/plugin_entry_points/log.ml", line 580, characters 15-16
        Called from file "src/plugins/wp/LogicCompiler.ml", line 719, characters 20-51
        Called from file "src/plugins/wp/LogicCompiler.ml", line 769, characters 10-23
        Called from file "src/plugins/wp/LogicSemantics.ml", line 535, characters 18-44
        Called from file "src/plugins/wp/Warning.ml", line 139, characters 6-10
        Called from file "src/plugins/wp/LogicSemantics.ml", line 843, characters 12-32
        Called from file "src/plugins/wp/Context.ml", line 31, characters 12-17
        Re-raised at file "src/plugins/wp/Context.ml", line 34, characters 41-46
        Called from file "src/plugins/wp/LogicCompiler.ml", line 392, characters 20-35
        Called from file "src/plugins/wp/Warning.ml", line 139, characters 6-10
        Called from file "src/plugins/wp/LogicSemantics.ml", line 843, characters 12-32
        Called from file "src/plugins/wp/Context.ml", line 31, characters 12-17
        Re-raised at file "src/plugins/wp/Context.ml", line 34, characters 41-46
        Called from file "src/plugins/wp/LogicCompiler.ml", line 392, characters 20-35
        Called from file "src/plugins/wp/LogicSemantics.ml", line 361, characters 31-50
        Called from file "src/plugins/wp/Warning.ml", line 139, characters 6-10
        Called from file "src/plugins/wp/LogicSemantics.ml", line 854, characters 12-42
        Called from file "src/plugins/wp/Context.ml", line 31, characters 12-17
        Re-raised at file "src/plugins/wp/Context.ml", line 34, characters 41-46
        Called from file "src/plugins/wp/Warning.ml", line 155, characters 14-18
        Called from file "src/plugins/wp/cfgWP.ml", line 466, characters 23-135
        Called from file "src/plugins/wp/Context.ml", line 68, characters 14-17
        Re-raised at file "src/plugins/wp/Context.ml", line 69, characters 43-48
        Called from file "src/plugins/wp/Context.ml", line 68, characters 14-17
        Re-raised at file "src/plugins/wp/Context.ml", line 69, characters 43-48
        Called from file "src/plugins/wp/Context.ml", line 68, characters 14-17
        Re-raised at file "src/plugins/wp/Context.ml", line 69, characters 43-48
        Called from file "src/plugins/wp/Context.ml", line 68, characters 14-17
```

24) Write the function contract for following function:

```
void f(int n,int *p,int *q)
{
  if(n>0)
     *p = n;
  else
     *q = n;
}
```

CODING:

```
/*@ requires \valid(p) && \valid(q);
 @ requires \separated(p,q);
 @ behavior p_changed:
 @      assumes n>0;
 @      assigns *p;
 @          ensures *p == n;
 @ behavior q_changed:
 @      assumes n<=0;
 @      assigns *q;
 @          ensures *q==n;
 @
*/
void f(int n,int *p,int *q)
{
   if(n > 0)
      *p = n;
   else
      *q = n;
}
```

OUTPUT:

```
kumarlaxmikant@Kumars-MacBook-Pro practice % frama-c -wp -wp-rte IntToPointer.c;
[kernel] Parsing FRAMAC_SHARE/libc/__fc_builtin_for_normalization.i (no preprocessing)
[kernel] Parsing IntToPointer.c (with preprocessing)
[rte] annotating function f
[wp] 8 goals scheduled
[wp] Proved goals:    8 / 8
     Qed:             8  (0.09ms-0.49ms-1ms)
kumarlaxmikant@Kumars-MacBook-Pro practice %
```