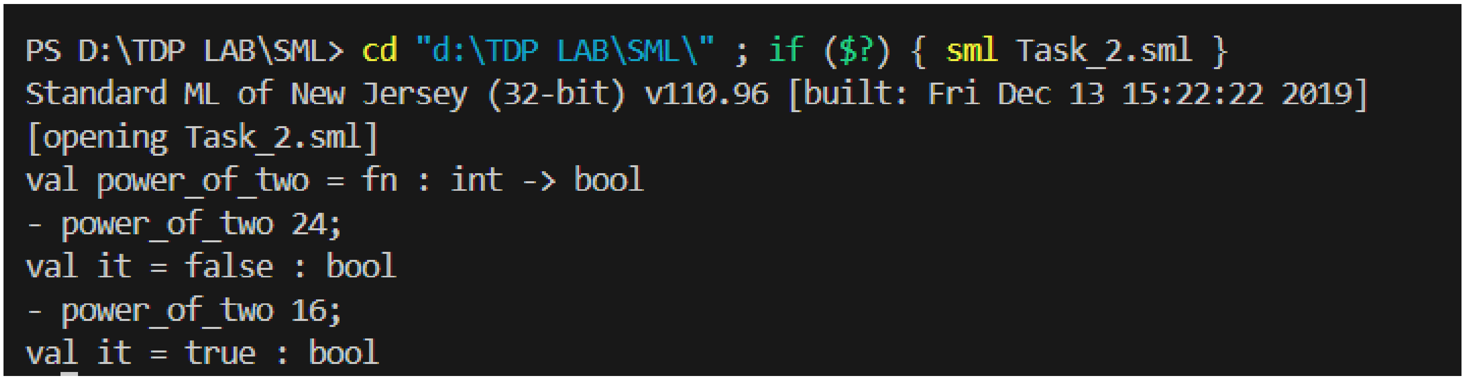
Task_2.sml

```
(* ------------------QUESTION-1--------------------- *)

hd([])= hd mean head of the list,which means it gives first element of the list
        .But [] is an empty list so error
Correction: Add some elements in list like hd([1,2,3])

explode(["toto"])=string to list of chars,but here toto is in a list
Correction:do explode("toto") ans will be ["t","o","t","o"]

implode("a","b")=List of chars concats to a string,but here we dont have a list
Correction:do implode(["a","b"]) and output is "ab"

["t]::["o","p"]=typo correction ["t]  should be ["t"]
Correction: "t" :: ["o","p"]; to add "t" as the head of the list ["o", "p"]


(* ------------------QUESTION-2--------------------- *)

The lists [3,4] and [3,4,5] are both lists of integers.So they are of same type
(3,4) and (3,4,5) are tuples which have fixed no.of elements
(3,4) has the type int * int (a pair of two integers)
(3,4,5) has the type int * int * int (a tuple of three integers)
so they are not of same type.

(* ------------------QUESTION-3--------------------- *)

fun new_fact n = new_if(n = 0,1,n*new_fact(n-1));
The reason new_fact does not compute the factorial correctly is because Standard ML uses eager evaluation.
In eager evaluation,all arguments to a function are evaluated before the function itself is executed.

(* ------------------QUESTION-4--------------------- *)

val {a=_,b=x,c=_} = {a=3,b=0,c=false};
val [_,_,x,_,_] = [~2, ~1, 0, 1, 2];
val [( _, _),(x, _)]=[(3,1), (0,9)];

(* ------------------QUESTION-5--------------------- *)

fun power_of_two n = if n=1 then true
                     else if n<=0 orelse n mod 2 <>0 then false
                     else power_of_two (n div 2);

(* ------------------QUESTION-6--------------------- *)

fun f (x, y, z, t) =
  if x = y then z + 1
  else if x > y then z
  else y + t;
The type of the function is fn : int * int * int * int -> int

(* ------------------QUESTION-7--------------------- *)

fun even 0=true|even n=odd(n-1)
and odd 0=false|odd n=even(n-1)

(* ------------------QUESTION-8--------------------- *)

val x=2 and y=x+1;
Results an error,because x in y=x+1 is not yet defined

val x=1;local val x=2 in val y=x+1 end; val z=x+1;
Results y=3 and z=2;

let val x=1 in let val x=2 and y=x in x+y end end;
Results x+y=3

(* ------------------QUESTION-9--------------------- *)

val x=1 and y=2 and z=3;
Results x=1,y=2,z=3

let val x=x+1 and z=x+4 in x+z end;
Results x=2;z=5 so x+z=7

let val t=x+1 in let val x=x+1 in x end end;
t=2;x=2;

(* ------------------QUESTION-10--------------------- *)

fun insert (n,[])=[n]
        | insert (n,x::xs)=
        if n<=x then n::x::xs
        else x::insert(n,xs);

(* ------------------QUESTION-11--------------------- *)

fun interclass ([],ys)=ys
   |interclass (xs,[])=xs
   |interclass(x::xs,y::ys)=
        if x<=y then x::interclass(xs,y::ys)
        else y::interclass(x::xs,ys);

(* ------------------QUESTION-12--------------------- *)

fun insert (x,[])=[x]
        | insert (x,y::ys)=
        if x<=y then x::y::ys
        else y::insert(x,ys);
fun insertion_sort []=[]
   |insertion_sort(x::xs)=insert(x,insertion_sort xs);

(* ------------------QUESTION-13---------------------- *)

fun iteration(cond,process,data)=
        if cond data then data
        else iteration(cond,process,process data);
fun is_sorted []=true
   |is_sorted [_]=true
   |is_sorted (x::y::ys) =
      if x<=y then is_sorted(y::ys)
      else false;
fun bubble xs =
    let
        fun bubble_once[] =[]
           |bubble_once [x]=[x]
           |bubble_once (x::y::ys) =
              if x>y then y::bubble_once(x::ys)
              else x::bubble_once(y::ys)
    in
        iteration (is_sorted, bubble_once, xs)
    end;

(* ------------------QUESTION-14---------------------- *)

fun subsets []=[[]]
   |subsets(x::xs)=
   let
        val restSubsets=subsets xs
        val xSubsets=List.map(fn subset=>x::subset) restSubsets
   in
        restSubsets @ xSubsets
   end;

(* ------------------QUESTION-15---------------------- *)

fun C f g x = f(g,x);
Type  fn : ('a * 'b -> 'c) -> 'a -> 'b -> 'c

(* ------------------QUESTION-16---------------------- *)

fun G(cond,[])=[]
   |G(cond,x::xs)= if cond x then x::G(cond,xs)
   else G(cond,xs);
fun is_even n=n mod 2=0;
val even_num=G(is_even,[1,2,3,4,5,6]);
```

```
fun conc (s1:string,s2:string):string=s1^s2;
fun F[]=""
|F(s::[])=s
|F(s::ss)=conc(s,F ss);
val conc_str=F["a","b","c","d"];

fun fold F nil y=y
| fold F(x::l)y=F(x,(fold F l y));
Type is : (val fold = fn : ('a * 'b -> 'b) -> 'a list -> 'b -> 'b)

(* ------------------QUESTION-17 ---------------------- *)

fun f(x,nil)=nil
  | f(x,a::aa)=if x(a) then a::f(x,aa) else f(x,aa);

'a=T_x * T_{nil} it says list is empty and 'a consists of T_x the type of x
                and T_{nil} is type of nil

'a=T_x*T_{a::aa} says T_x is the type of x and T_{a::aa} is type of non empty list
'b=T_{nil} says the result of the function is nil the return type 'b is nil
T_{x(a)}=bool says  applying function x on a mus be of type bool

Type:  val f = fn : ('a -> bool) * 'a list -> 'a list

(* ------------------QUESTION-19 ---------------------- *)

datatype COORDS=Coords of real*real*real;
fun distance (Coords(a1,a2,a3),Coords(b1,b2,b3))=
        Math.sqrt((b1-a1)*(b1-a1)+(b2-a2)*(b2-a2)+(b3-a3)*(b3-a3));
val coord1=Coords(1.0,3.0,4.0);
val coord2=Coords(5.0,2.0,7.0);
val dist=distance(coord1,coord2);

(* ------------------QUESTION-20--------------------- *)

datatype PERSON = Person of{
    name:string,
    fname:string,
    age:int,
    datebirth:int*int*int
};

(* ------------------QUESTION-21--------------------- *)

val i= ref 10;
i := !i + 1;
val res=i;
i := !i - 1;
val res=i;
i := 20;
```

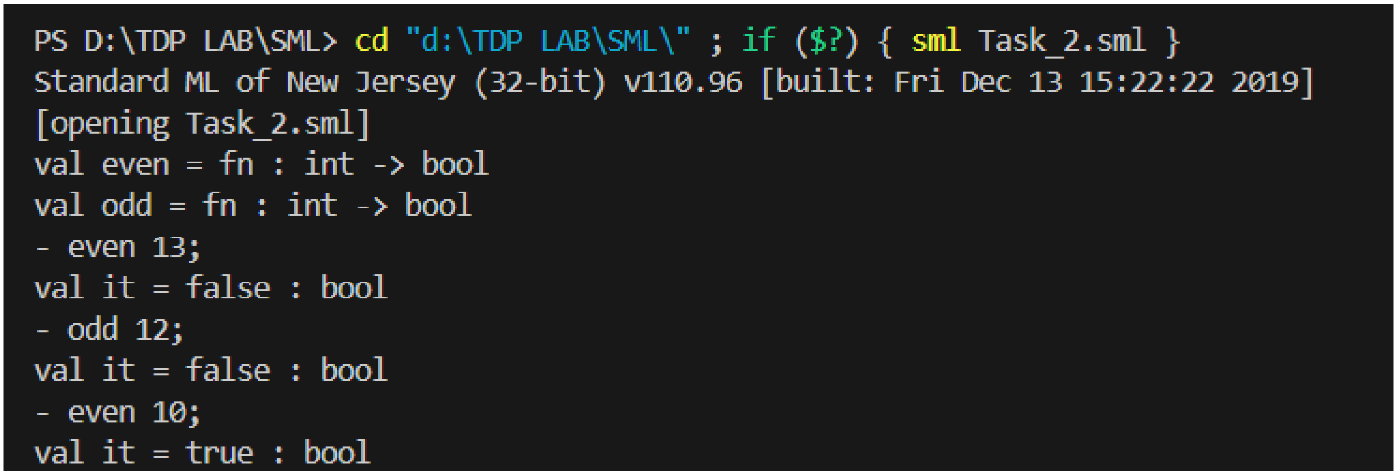
```
val res=i;

(* ------------------QUESTION-22---------------------- *)

fun afficher i = print (Int.toString i ^ "\n");
fun whileloop (i:int ref)=
    if !i>10 then()
    else(
        afficher(!i);
        i := !i+1;
        whileloop i
    );
val i=ref 1;
whileloop i;

(* ------------------QUESTION-23---------------------- *)

exception NegFact of int;
fun fact n=
        if n<0 then raise NegFact n
        else if n=0 then 1
        else let
            fun fact 0 = 1
                |fact n=n*fact(n-1);
        in
            fact n
        end;
```

5.



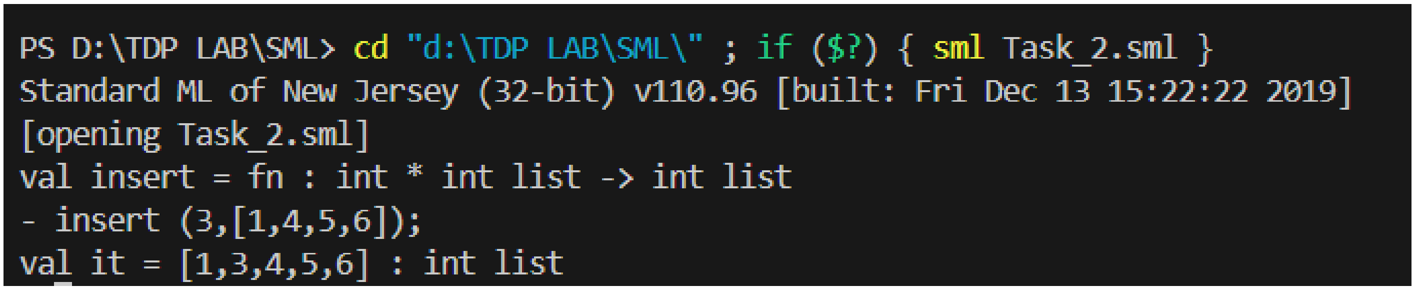
```
PS D:\TDP LAB\SML> cd "d:\TDP LAB\SML\" ; if ($?) { sml Task_2.sml }
Standard ML of New Jersey (32-bit) v110.96 [built: Fri Dec 13 15:22:22 2019]
[opening Task_2.sml]
val power_of_two = fn : int -> bool
- power_of_two 24;
val it = false : bool
- power_of_two 16;
val it = true : bool
```

7.



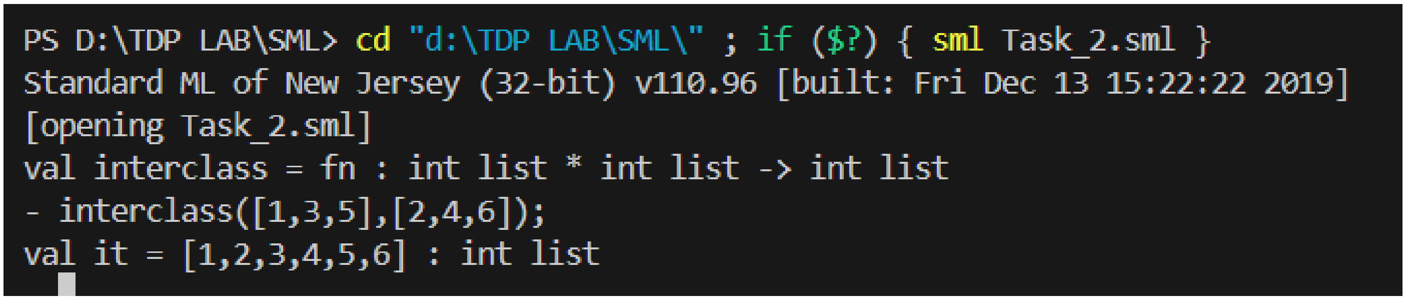
```
PS D:\TDP LAB\SML> cd "d:\TDP LAB\SML\" ; if ($?) { sml Task_2.sml }
Standard ML of New Jersey (32-bit) v110.96 [built: Fri Dec 13 15:22:22 2019]
[opening Task_2.sml]
val even = fn : int -> bool
val odd = fn : int -> bool
- even 13;
val it = false : bool
- odd 12;
val it = false : bool
- even 10;
val it = true : bool
```

10.



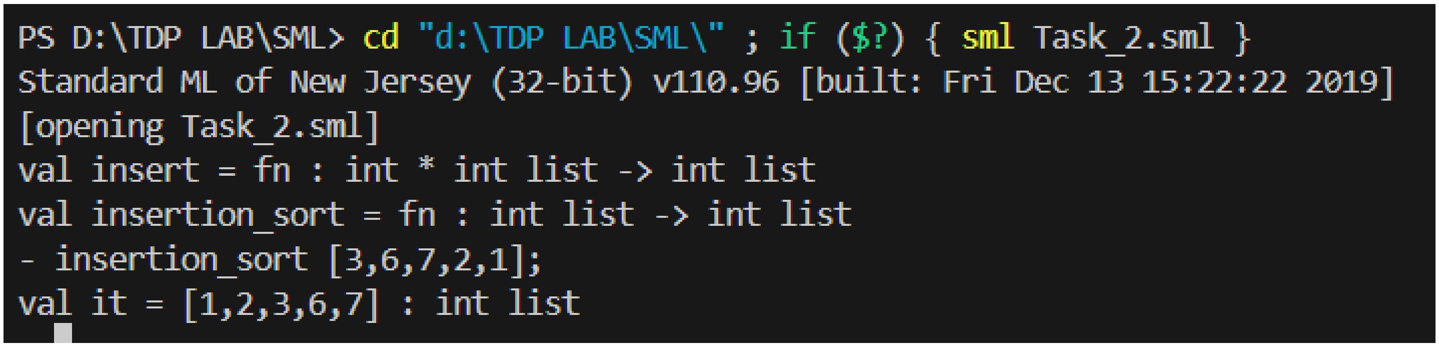
```
PS D:\TDP LAB\SML> cd "d:\TDP LAB\SML\" ; if ($?) { sml Task_2.sml }
Standard ML of New Jersey (32-bit) v110.96 [built: Fri Dec 13 15:22:22 2019]
[opening Task_2.sml]
val insert = fn : int * int list -> int list
- insert (3,[1,4,5,6]);
val it = [1,3,4,5,6] : int list
```

11.



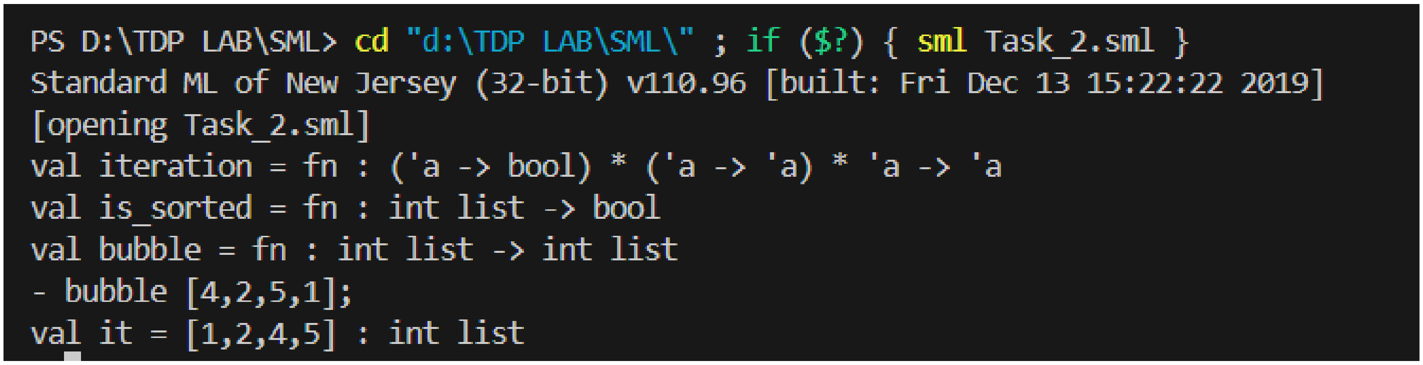
```
PS D:\TDP LAB\SML> cd "d:\TDP LAB\SML\" ; if ($?) { sml Task_2.sml }
Standard ML of New Jersey (32-bit) v110.96 [built: Fri Dec 13 15:22:22 2019]
[opening Task_2.sml]
val interclass = fn : int list * int list -> int list
- interclass([1,3,5],[2,4,6]);
val it = [1,2,3,4,5,6] : int list
```

12.



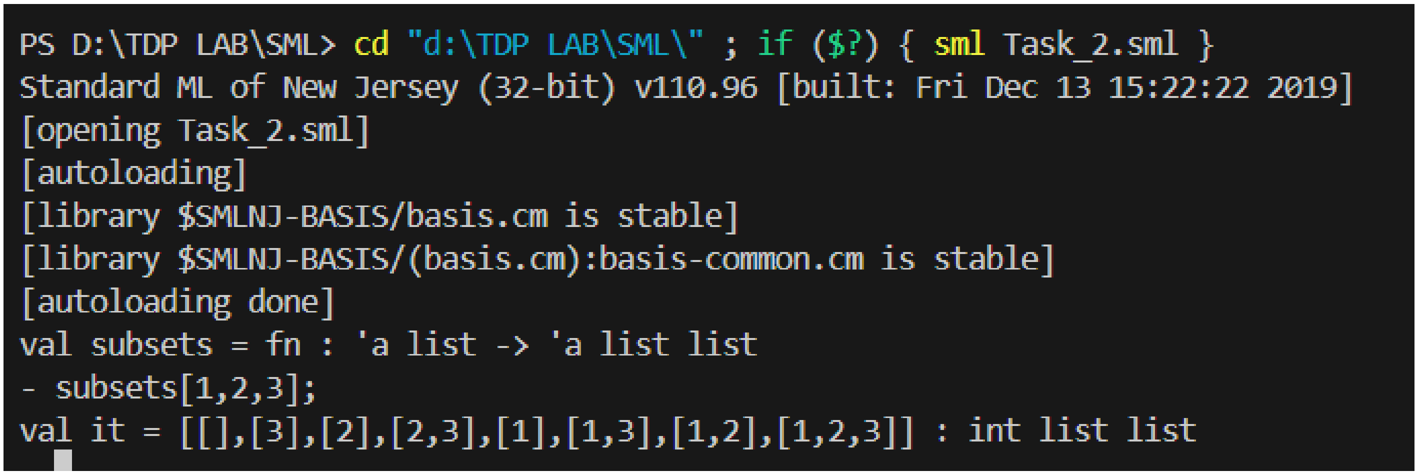
```
PS D:\TDP LAB\SML> cd "d:\TDP LAB\SML\" ; if ($?) { sml Task_2.sml }
Standard ML of New Jersey (32-bit) v110.96 [built: Fri Dec 13 15:22:22 2019]
[opening Task_2.sml]
val insert = fn : int * int list -> int list
val insertion_sort = fn : int list -> int list
- insertion_sort [3,6,7,2,1];
val it = [1,2,3,6,7] : int list
```

13.



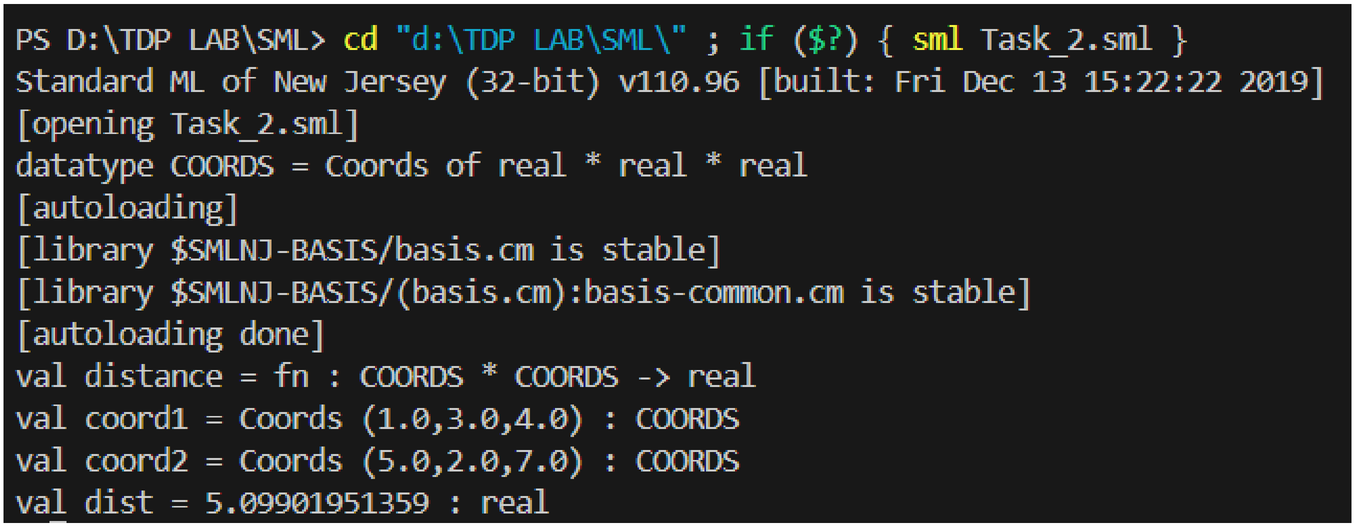
```
PS D:\TDP LAB\SML> cd "d:\TDP LAB\SML\" ; if ($?) { sml Task_2.sml }
Standard ML of New Jersey (32-bit) v110.96 [built: Fri Dec 13 15:22:22 2019]
[opening Task_2.sml]
val iteration = fn : ('a -> bool) * ('a -> 'a) * 'a -> 'a
val is_sorted = fn : int list -> bool
val bubble = fn : int list -> int list
- bubble [4,2,5,1];
val it = [1,2,4,5] : int list
```

14.



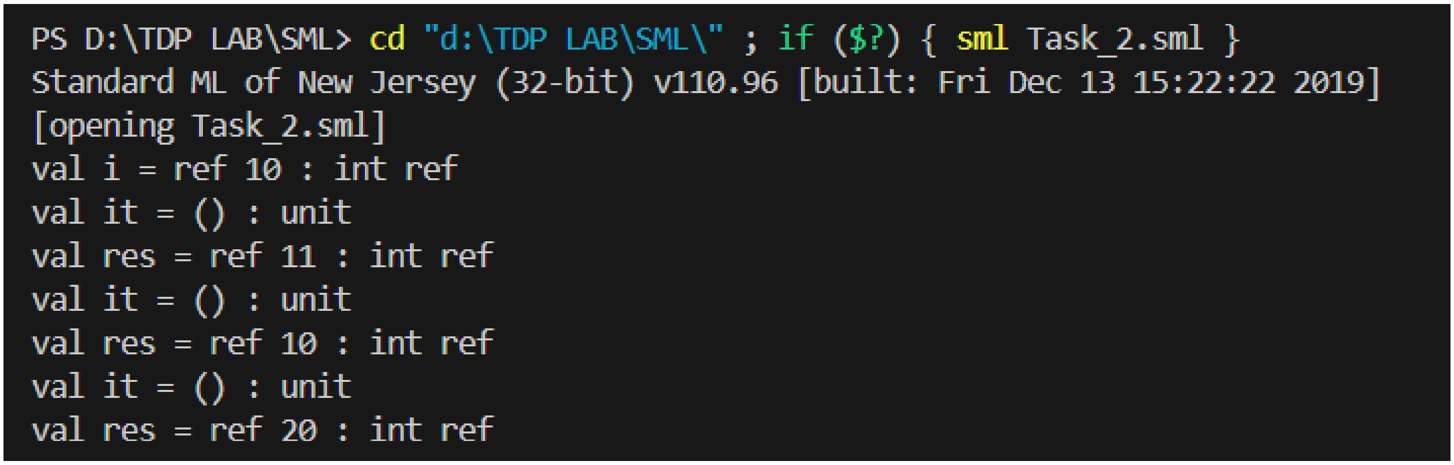
```
PS D:\TDP LAB\SML> cd "d:\TDP LAB\SML\" ; if ($?) { sml Task_2.sml }
Standard ML of New Jersey (32-bit) v110.96 [built: Fri Dec 13 15:22:22 2019]
[opening Task_2.sml]
[autoloading]
[library $SMLNJ-BASIS/basis.cm is stable]
[library $SMLNJ-BASIS/(basis.cm):basis-common.cm is stable]
[autoloading done]
val subsets = fn : 'a list -> 'a list list
- subsets[1,2,3];
val it = [[],[3],[2],[2,3],[1],[1,3],[1,2],[1,2,3]] : int list list
```

19.



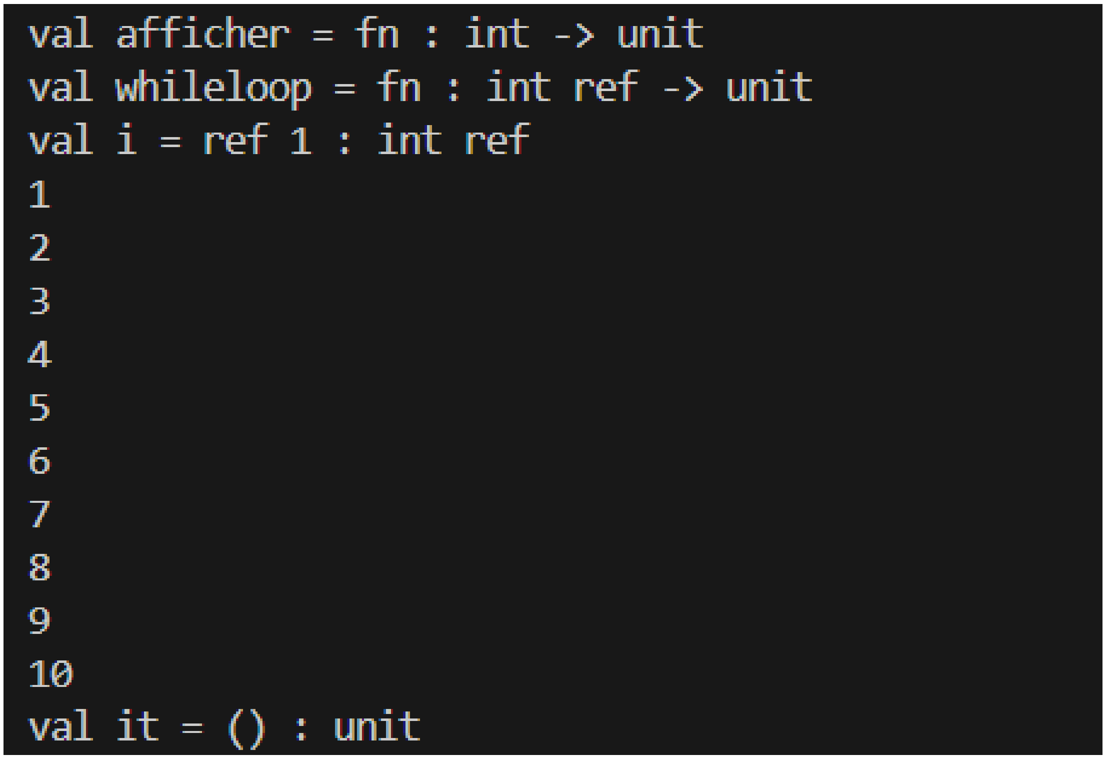
```
PS D:\TDP LAB\SML> cd "d:\TDP LAB\SML\" ; if ($?) { sml Task_2.sml }
Standard ML of New Jersey (32-bit) v110.96 [built: Fri Dec 13 15:22:22 2019]
[opening Task_2.sml]
datatype COORDS = Coords of real * real * real
[autoloading]
[library $SMLNJ-BASIS/basis.cm is stable]
[library $SMLNJ-BASIS/(basis.cm):basis-common.cm is stable]
[autoloading done]
val distance = fn : COORDS * COORDS -> real
val coord1 = Coords (1.0,3.0,4.0) : COORDS
val coord2 = Coords (5.0,2.0,7.0) : COORDS
val dist = 5.09901951359 : real
```

21.



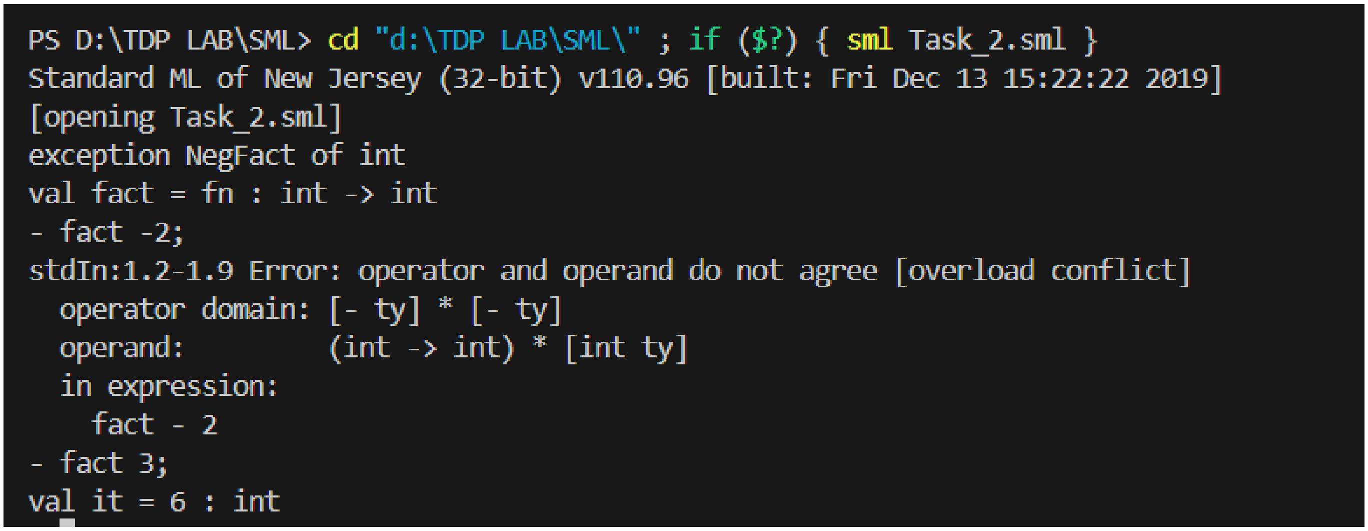
```
PS D:\TDP LAB\SML> cd "d:\TDP LAB\SML\" ; if ($?) { sml Task_2.sml }
Standard ML of New Jersey (32-bit) v110.96 [built: Fri Dec 13 15:22:22 2019]
[opening Task_2.sml]
val i = ref 10 : int ref
val it = () : unit
val res = ref 11 : int ref
val it = () : unit
val res = ref 10 : int ref
val it = () : unit
val res = ref 20 : int ref
```

22.



```
val afficher = fn : int -> unit
val whileloop = fn : int ref -> unit
val i = ref 1 : int ref
1
2
3
4
5
6
7
8
9
10
val it = () : unit
```

23.



```
PS D:\TDP LAB\SML> cd "d:\TDP LAB\SML\" ; if ($?) { sml Task_2.sml }
Standard ML of New Jersey (32-bit) v110.96 [built: Fri Dec 13 15:22:22 2019]
[opening Task_2.sml]
exception NegFact of int
val fact = fn : int -> int
- fact -2;
stdIn:1.2-1.9 Error: operator and operand do not agree [overload conflict]
  operator domain: [- ty] * [- ty]
  operand:         (int -> int) * [int ty]
  in expression:
    fact - 2
- fact 3;
val it = 6 : int
```