/\* This is a

multiline

comment \*/

type point = {

x: float,

y: float,

};

let some\_string = "this is a string";

let rec length = lst =>

switch (lst) {

| [] => 0

| [head, ...tail] => 1 + length(tail)

};

type result('a, 'b) =

| Ok('a)

| Error('b);

let promisify = (res: result('a, 'b)) : Js.Promise.t('a) =>

switch (res) {

| Ok(a) => Js.Promise.resolve(a)

| Error(b) => Js.Promise.reject(b)

};

exception Test;

module MakeFFI = (T: {type t;}) => {

[@bs.module] external ffi : string => T.t = "";

};

type expression =

| Const(float)

| Var(string)

| Sum(expression, expression) /\* e1 + e2 \*/

| Diff(expression, expression) /\* e1 - e2 \*/

| Prod(expression, expression) /\* e1 \* e2 \*/

| Quot(expression, expression); /\* e1 / e2 \*/

class point = {

as \_;

val mutable x = 0;

pub get\_x = x;

pri move = d => x = x + d;

};