// Implicit type conversions

console.log (8+’9’); //’89’

console.log (8\*’9’); //72

console.log (8 /false); // Infinity

console.log (8 -1 true); //9

console.log (8- ‘nine’); //; //NAN

console.log (8 + null); //8

console.log (8 + undefined); //NAN

console.log (true + ‘love’); //truelove’’

\*use object functions for explicit type conversion

console.log (8 + Number (‘9’)); //11

(string (8) + String(true)); //true

functions

function max (item1, item2) {

if (item1 > item2) {

return item1;

} else if (item2 > item1) {

return item2;

}

}

console.log (max (10,20));

console.log (max (‘zero’, ‘one’));

max (1,1) //undefined

console.log(true, false);

\*arguments object (only valid inside a function)

Function f(C) {

var result = arguments[0];

for(let index = 1; index < arguments.length; index++){

result t= arguments[index];

}

return result;

}

var res1 -f1(1,2,3,4,5);

var res2 - f1(‘a’,’b’,’c’);

// default parameters values

~~function~~

// “rest” parameters

Function f3(arg1, arg2... otherArgs)

Console.log(arg1);

Console.log(arg2);

Console.log(otherArgs);

// function expressions

var f4 = function (a,b){ return

a+b; };

f4(1,2);

“arrow” function

~~var f5 = (a,b) => {var p=a\*b~~

//functions are “first-class objects”

Function operation(operation switch(operator){

Case ‘t’ :return(a,b)=>a+b;

}

}

var op = Operation (‘\*’);

op (10,20);

Operation(‘1’)(10,20);

// nesting functions

function outerFunc(){

function innerFunc(){

//nested function

}

//arrays

}RESEARCH ABOUT ARRAYS