//In the future will be user's input on AR table

function AR(action\_num, description, startDate, dueDate, category, subCategory, owner, secondaryOwner, priorty, arStatus, statusDetails, comments) {

this.action\_num = action\_num;//NOT IN CODE

this.description = description;

this.startDate = startDate;

this.dueDate = dueDate;

this.category = category;

this.subCategory = subCategory;

this.owner = owner;

this.secondaryOwner = secondaryOwner;

this.priorty = priorty;

this.arStatus = arStatus;//PROBLEM save word

this.statusDetails = statusDetails;

this.comments = comments;

}

var AR5 = new AR("5", "wifi", new Date('2017-01-01'), new Date('2017-01-20'), "HW", "thermal", null, null, "5","open", null, null);

//In the future will be taken from worker's profile

var Hana\_qualifacations = ["KPI"];

//In the future will be taken from worker's profile

var Talya\_qualifacations = ["KPI", "thermal"];

//In the future will be taken from worker's profile

var Oren\_qualifacations = ["KPI"];

//In the future will be taken from worker's profile

var Ido\_qualifacations = ["thermal"];

var qualified = [];

//In the future will be general for all profiles

if ((Hana\_qualifacations.indexOf(AR5.category) >= 0) || (Hana\_qualifacations.indexOf(AR5.subCategory) >= 0)) {

qualified.push("Hana");

}

//In the future will be general for all profiles

if ((Talya\_qualifacations.indexOf(AR5.category) >= 0) || (Talya\_qualifacations.indexOf(AR5.subCategory) >= 0)) {

qualified.push("Talya");

}

if ((Oren\_qualifacations.indexOf(AR5.category) >= 0) || (Talya\_qualifacations.indexOf(AR5.subCategory) >= 0)) {

qualified.push("Talya");

}

if ((Ido\_qualifacations.indexOf(AR5.category) >= 0) || (Talya\_qualifacations.indexOf(AR5.subCategory) >= 0)) {

qualified.push("Talya");

}

//Temporaly print the result

//console.log(qualified);

//All cuurent ARs.

//In the future will be taken from the DB

var AR1 = new AR("1", "a", new Date('2017-01-03'), new Date('2017-01-13'),"Services", null, "Hana", "Ofer", "3","open", null, null);

var AR2 = new AR("2", "b", new Date('2016-12-15'), new Date('2017-01-10'),null, "Data Path", "Hana", null, "3","open", null, null);

var AR3 = new AR("3", "c", new Date('2017-01-07'), new Date('2017-01-22'), "HW", "thermal", "Talya", "Nevo", "5","open", null, null);

var AR4 = new AR("4", "d", new Date('2016-12-15'), new Date('2017-01-25'),"HW", "TpT", "Talya", null, "5","open", null, null);

var AR6 = new AR("3", "c", new Date('2017-01-07'), new Date('2017-01-22'), "HW", "thermal", "Ido", "Nevo", "5", "open", null, null);

var AR7 = new AR("4", "d", new Date('2016-12-15'), new Date('2017-01-25'), "HW", "TpT", "Oren", null, "5", "open", null, null);

//Array with all the employee current tasks.

////In the future will be taken from worker's profile/user DB

var Hana\_tasks = [AR1,AR2];

var Talya\_tasks = [AR3, AR4];

var Oren\_tasks = [AR6];

var Ido\_tasks = [AR7];

var Hana\_tasks\_tmp = Hana\_tasks;

var Talya\_tasks\_tmp = Talya\_tasks;

var Oren\_tasks\_tmp = Oren\_tasks;

var Ido\_tasks\_tmp = Ido\_tasks;

//Creates array with only relevant ARs of each worker (only between SD to DD) , and if needed - changes start/due date accordingly

var Hana\_relevant\_tasks = [];

var Talya\_relevant\_tasks = [];

var Oren\_relevant\_tasks = [];

var Ido\_relevant\_tasks = [];

for (j = 0, len = Hana\_tasks\_tmp.length; j < len; j++) {

var i = Hana\_tasks\_tmp[j];

if ((i.startDate > AR5.dueDate) || (i.dueDate < AR5.startDate)) {

continue;

}

else if ((i.startDate <= AR5.startDate) && (i.dueDate <= AR5.dueDate)) {

i.startDate = AR5.startDate;

Hana\_relevant\_tasks.push(i);

}

else if ((i.startDate >= AR5.startDate) && (i.dueDate >= AR5.dueDate)) {

i.dueDate = AR5.dueDate;

Hana\_relevant\_tasks.push(i);

}

else if ((i.startDate <= AR5.startDate) && (i.dueDate >= AR5.dueDate)) {

i.startDate = AR5.startDate;

i.dueDate = AR5.dueDate;

Hana\_relevant\_tasks.push(i);

}

else if ((i.startDate >= AR5.startDate) && (i.dueDate <= AR5.dueDate)) {

Hana\_relevant\_tasks.push(i);

}

}

for (j = 0, len = Talya\_tasks\_tmp.length; j < len; j++) {

var i = Talya\_tasks\_tmp[j];

if ((i.startDate > AR5.dueDate) || (i.dueDate < AR5.startDate)) {

continue;

}

else if ((i.startDate <= AR5.startDate) && (i.dueDate <= AR5.dueDate)) {

i.startDate = AR5.startDate;

Talya\_relevant\_tasks.push(i);

}

else if ((i.startDate >= AR5.startDate) && (i.dueDate >= AR5.dueDate)) {

i.dueDate = AR5.dueDate;

Talya\_relevant\_tasks.push(i);

}

else if ((i.startDate <= AR5.startDate) && (i.dueDate >= AR5.dueDate)) {

i.startDate = AR5.startDate;

i.dueDate = AR5.dueDate;

Talya\_relevant\_tasks.push(i);

}

else if ((i.startDate >= AR5.startDate) && (i.dueDate <= AR5.dueDate)) {

Talya\_relevant\_tasks.push(i);

}

}

for (j = 0, len = Oren\_tasks\_tmp.length; j < len; j++) {

var i = Oren\_tasks\_tmp[j];

if ((i.startDate > AR5.dueDate) || (i.dueDate < AR5.startDate)) {

continue;

}

else if ((i.startDate <= AR5.startDate) && (i.dueDate <= AR5.dueDate)) {

i.startDate = AR5.startDate;

Oren\_relevant\_tasks.push(i);

}

else if ((i.startDate >= AR5.startDate) && (i.dueDate >= AR5.dueDate)) {

i.dueDate = AR5.dueDate;

Oren\_relevant\_tasks.push(i);

}

else if ((i.startDate <= AR5.startDate) && (i.dueDate >= AR5.dueDate)) {

i.startDate = AR5.startDate;

i.dueDate = AR5.dueDate;

Oren\_relevant\_tasks.push(i);

}

else if ((i.startDate >= AR5.startDate) && (i.dueDate <= AR5.dueDate)) {

Oren\_relevant\_tasks.push(i);

}

}

for (j = 0, len = Oren\_tasks\_tmp.length; j < len; j++) {

var i = Ido\_tasks\_tmp[j];

if ((i.startDate > AR5.dueDate) || (i.dueDate < AR5.startDate)) {

continue;

}

else if ((i.startDate <= AR5.startDate) && (i.dueDate <= AR5.dueDate)) {

i.startDate = AR5.startDate;

Ido\_relevant\_tasks.push(i);

}

else if ((i.startDate >= AR5.startDate) && (i.dueDate >= AR5.dueDate)) {

i.dueDate = AR5.dueDate;

Ido\_relevant\_tasks.push(i);

}

else if ((i.startDate <= AR5.startDate) && (i.dueDate >= AR5.dueDate)) {

i.startDate = AR5.startDate;

i.dueDate = AR5.dueDate;

Ido\_relevant\_tasks.push(i);

}

else if ((i.startDate >= AR5.startDate) && (i.dueDate <= AR5.dueDate)) {

Ido\_relevant\_tasks.push(i);

}

}

function calculate\_length(An\_AR)

{

var oneDay = 24 \* 60 \* 60 \* 1000; // hours\*minutes\*seconds\*milliseconds

var length = 0;

length = Math.round(Math.abs((An\_AR.startDate.getTime() - An\_AR.dueDate.getTime()) / (oneDay)));

return length;

}

//array with all araays of the employees tasks

var tasks = [Talya\_relevant\_tasks, Hana\_relevant\_tasks, Oren\_relevant\_tasks,Ido\_relevant\_tasks];

var name\_weight = [];

//function for calculate all tasks weight of one employee

function sum\_of\_weight(tasks) {

for (i = 0; i < tasks.length; i++) {

var sum = 0;

for (j = 0; j < tasks[i].length; j++) {

var weight = (calculate\_length(tasks[i][j])) \* (tasks[i][j].priorty);

sum += weight;

if (j == (tasks[i].length)-1) {

name\_weight.push({ name: tasks[i][j].owner, weight: sum });

}

}

}

}

(sum\_of\_weight(tasks));

var name\_numOfArs = [];

function sum\_of\_ars(tasks) {

for (i = 0; i < tasks.length; i++) {

var count = 0;

for (j = 0; j < tasks[i].length; j++) {

count++;

var x = tasks[i][j].owner;

}

name\_numOfArs.push({ name: x, num: count});

}

return name\_numOfArs;

}

sum\_of\_ars(tasks);

var final\_array = []

//sort the employees according to they sum

function best\_match(name\_numOfArs, name\_weight) {

for (i = 0; i < name\_numOfArs.length; i++) {

for (t = 0; t < name\_weight.length; t++) {

if (name\_numOfArs[i].name == name\_weight[t].name) {

var x = (name\_numOfArs[i].num) \* 10;

var final\_value = x + name\_weight[t].weight;

final\_array.push({ name: name\_numOfArs[i].name , weight: final\_value });

}

}

}

return final\_array;

}

best\_match(name\_numOfArs, name\_weight);

//console.log("final\_array")

//console.log(final\_array)

function sort(final\_array) {

function dynamicSort(property) {

var sortOrder = 1;

if (property[0] === "-") {

sortOrder = -1;

property = property.substr(1);

}

return function (a, b) {

var result = (a[property] < b[property]) ? -1 : (a[property] > b[property]) ? 1 : 0;

return result \* sortOrder;

}

}

final\_array.sort(dynamicSort("weight"));

}

console.log("final\_array no sorted");

console.log(final\_array);

sort(final\_array);

console.log("final\_array sorted");

console.log(final\_array);

var final1=[]

function final(final\_array) {

for (i = 0; i < final\_array.length; i++) {

final1.push(final\_array[i].name)

}

return final

}

final(final\_array)

console.log(final1)

//console.log(sort(name\_weight));