DOCUMENTACIÓ

Laboratorio de Lenguaje Máquina ARM

Enunciado de la práctica:

guerra de barcos

PROGRAMADOR1: Bernat Boscà Candel
(bernat.bosca@estudiants.urv.cat)
PROGRAMADOR2: Albert Cañellas Solé
(albert.canellas@estudiants.urv.cat)
PROFESSOR: David Gámez Alari
ASSIGNATURA: F. Computadors
ENSENYAMENT: GEI-Biotec
UNIVERSITAT: Universitat Rovira i Virgili
CURS: 2015/2016
GRUP: T3
CONVOCATÒRIA: 1ª

DATA: 25/05/2016

Codi en #C:

Barcos_p

```
#define NUM_PARTIDAS 150;

char nd8, nd9, nd9;
char matriz_barcos[100]
char matriz_disparos[100];

void principal()
{
realitzar_partida(8, matriz_barcos, matriz_disparos, &nd8);
realitzar_partida(9, matriz_barcos, matriz_disparos,  &nd9);
realitzar_partida(10, matriz_barcos, matriz_disparos,  &nd10);
}
```

La funció principal crida a les funcions realitzar partida per els diferents taulells (8x8, 9x9, 10x10).

```
void realitzar_partidas(int dim, char tablero_barcos[],char tablero_disparos[], char
*var_promedio)
{
int dis_partida; dis_Totals=0; npartidas=NUM_PARTIDAS; ok=0;
while (npartidas > 0) ;
        {
        while(ok==0)
                {
                ok=B_inicialitzar_barcos(dim, tablero_barcos);
                }
        B_inicialitzar_disparos(dim, tablero_disparos);
        dis_partida=jugar(dim, tablero_disparos);
        dis_Totals+=dis_paridas;
        npartidas --;
        }
        *var_promedio=dis_Totals/NUM_PARTIDAS;
        }
}
```

Fa un NUM_PARTIDES de un tipus de dimensió en concret. Aquesta funció cridarà a les funcions: B_inicialitzar_barcos, B_inicialitzar_disparos, jugar. També calcula la mitjana de dispars de un numero de partides.

```
void B_inicialitzar_disparos(int dim, tablero_disparos[])
{
int dim_total, i;
dim_total=dim*dim;
for(i=0;i<dim_total;i++)
        {
        tablero_disparos[i]='?';
        }
}
```

Omple el taulell de dispars amb '?'.

Barcos_j

```
int jugar(int dim, char tablero_disparos[])
{
int final_barcos:=0; flag12:=0; flag5:=0;tocats_conse=0;
int c, f, total_dispars, res_tir:=0; c_tocat; f_tocat;

while(final_barcos < 10)
        {
        fer_un_tret(&f, &c, dim, tablero_disparos);
        res_tir=efectuar_disparo(dim, tablero_disparos, f, c);
        if (res_tir<2)&(flag12>0)
                {flag5++;}
        if(res_tir==2)
                {tocats_conse++; flag12=1; c_tocat=c; f_tocat=f;}
        if(flag5==2)
                {tocats_conse=1;}
        if(flag5>=3)
                {tocats_conse++;}
        if(tocats_conse>=4)
                {tocats_conse=0;}
        if(res_tir==3)
                {final_barcos++; falg12=0; tocats_conse=0; flag5=0;}
        }
total_dispars=B_num_disparos();
}
```

Realitza una partida al complet per a un determinat tipus de dimensió. En aquesta rutina es crida a les funcions: fer_un_tret, efectuar_disparo.

```
void fer_un_tret(int *f, int *c, int dim, char tablero_disparos[], int flag12, int tocats_conse)
{
	int coordena= f*dim + c; cor=0; i=0; formal=0; col=0;
	if(flag12!=0)&(tablero_disparos[coordena]==’@’)
	{
		if(c_tocat<dim-1)&(cordena+1==’?’)
			{c=c_tocat+1; f=f_tocat;}
		else
		{	if(c_tocat>0)
			{
				cor=coordena-1;
				if(tocats_conse>=2)
				{
					cor=coordena-1;
					if(tocats_conse>=3)
						{cor-=1;}
				}
				if(tablero_disparos[cor]==’?’)
				{
					c=c_tocat-1; f=f_tocat;
					if(tocats_conse>=2)
					{
						c_tocat--;
						if(tocats_conse>=3)
							{c_tocat--;}
					}
				}
			}
			else
			{
				if(f_tocat<dim-1)
				{
					formal+=dim;
					while(tablero_disparos[formal]==’@’)
						{i++; formal+=dim;}
					if (tablero_disparos[formal]==’?’)
						{c=c_tocat; f=f_tocat+i;}
				}
				else
				{
					if(fila_tocat>0)
					{
						col=coordena-dim;
						if(tocats_conse>=2)
						{
							col-=dim
```

```
                                        if(tocats_conse>=3)
                                        {
                                                col-=dim
                                                if(tocats_cons>=4)
                                                        {col-=dim}
                                        }
                                }
                                if(tablero_disparos[col]==?)
                                {
                                        c=c_tocat; f=f_tocat+1;
                                        if(tocats_conse>=2)
                                        {
                                                f-=1;
                                                if(tocats_conse>=3)
                                                {
                                                        f-=1;
                                                        if(tocats_conse>=4)
                                                              { f-=1;}
                                                }
                                        }
                                }
                        }
                }
        }
        else
        {
                i=0;
                while(i<1)
                {
                        f=mod_random(dim);
                        c=mod_random(dim);
                        coordena= f *dim+c;
                        if(tablero_disparo[coordena]=='?')
                                {i++;}
                }
        }
}
```

Calcula les coordenades del pròxim tret, a partir del tret anterior. Si el dispar a tocat un vaixell la rutina disparà a les caselles amb més possibilitat de que hi hagi una part d'aquest vaixell(dalt, baix, esquerra, dreta).

```
int efectuar_disparo(int dim, char tablero_disparos[], int f, int c) return int
{
int res_tir, coordena, i, dim_total;
char fila;
fila=f+65;
res_tir=B_dispara(fila, c);
coordena= f*dim + c;
dim_total=dim*dim;
switch(res_tir)
        {
        Case -1: break;                                          /*Error*/
        Case  0: break;                                          /*Repetit*/
        Case 1: tablero_disparos[coordena]='.'; break;           /*Agua*/
        Case 2:                                                  /*Tocat*/
                tablero_disparos[coordena]='@';
                int coordena_ad_a ( coordena, dim, tablero_disparos, f, c);
                int coordena_ad_s ( coordena, dim, tablero_disparos, f, c);
                int coordena_sd_a ( coordena, dim, tablero_disparos, f, c);
                int coordena_sd_s ( coordena, dim, tablero_disparos, f, c);
                break;
        Case 3:                                                  /*Enfonsat*/
                tablero_disparos[coordena]='@';
                for(i=0;i<dim_total;i++)
                {
                        if ((tablero_disparos[i]=='@')
                        {
                                voltejar_aigua(i, dim, tablero_disparos);
                        }
                }
                break;
        Default: break;                                          /*Error*/
        }
return res_tir;
}
```

Actualitza els dos taulell a partir del llançament seleccionat per fer_un_tret. Si el resultat de B_dispara es aigua actualitza el taulell amb aigua('.'), si el resultat es un tocat actualitza el taulell amb un tocat('@') i fica aigua a les diagonals, si el resultat es un enfonsat actualitza el taulell amb un tocat('@') i rodeja el vaixell amb aigua.

```
void voltejar_aigua( int coordena, int dim, char tablero_disparos[], int f, int c)
{
        int coordena_sd_s ( coordena, dim, tablero_disparos,  f, c);
        int coordena_sd ( coordena, dim, tablero_disparos, f, c);
        int coordena_sd_a ( coordena, dim, tablero_disparos, f, c);
        int coordena_s ( coordena, dim, tablero_disparos, f, c);
        int coordena_a ( coordena, dim, tablero_disparos, f,  c);
        int coordena_ad_s ( coordena, dim, tablero_disparos, f, c);
        int coordena_sd ( coordena, dim, tablero_disparos, f, c);
        int coordena_as_a ( coordena, dim, tablero_disparos,  f,  c);
}
```

Crida a les diferents funcions que rodejaran el vaixell enfonsat amb aigua.

```
int coordena_ad_a (int coordena, int dim, char tablero_disparos[], int f, int c)
{
        if(f<dim-1)&(c<dim-1)&(tablero_disparos [dim+coordena+1]=='?')
        {tablero_disparos[dim+coorde+1]='.';}
}
```

Actualitza la casella [coordena+dim+1] amb aigua si compleix les condicions del taulell.

```
int coordena_ad_s (int coordena, int dim, char tablero_disparos[], int f, int c)
{
        if(f<dim-1)&(c>0)&(tablero_disparos [dim+coordena-1]=='?')
        {tablero_disparos[dim+coorde-1]='.';}
}
```

Actualitza la casella [coordena+dim-1] amb aigua si compleix les condicions del taulell.

```
int coordena_sd_s (int coordena, int dim, char tablero_disparos[], int f, int c)
{
        if(f>0)&(c>0)&(tablero_disparos [dim-coordena-1]=='?')
        {tablero_disparos[dim-coorde-1]='.';}
}
```

Actualitza la casella [coordena-dim-1] amb aigua si compleix les condicions del taulell.

```
int coordena_sd_a (int coordena, int dim, char tablero_disparos[], int f, int c)
{
        if(f>0)&(c<dim-1)&(tablero_disparos [dim-coordena+1]=='?')
        {tablero_disparos[dim-coorde+1]='.';}
}
```

Actualitza la casella [coordena-dim+1] amb aigua si compleix les condicions del taulell.

```
int coordena_sd (int coordena, int dim, char tablero_disparos[], int f, int c)
{
        if(f>0)&(tablero_disparos [dim-coordena]=='?')
        {tablero_disparos[dim-coorde]='.';}
}
```

Actualitza la casella [coordena-dim] amb aigua si compleix les condicions del taulell.

```
int coordena_ad (int coordena, int dim, char tablero_disparos[], int f, int c)
{
        if(f<dim-1)&(tablero_disparos [dim+coordena]=='?')
        {tablero_disparos[dim+coordena]='.';}
}
```

Actualitza la casella [coordena+dim] amb aigua si compleix les condicions del taulell.

```
int coordena_a (int coordena, int dim, char tablero_disparos[], int f, int c)
{
        if(c<dim-1)&(tablero_disparos [dim+1]=='?')
        {tablero_disparos[dim+1]='.';}
}
```

Actualitza la casella [coordena+1] amb aiguasi compleix les condicions del taulell.

```
int coordena_s (int coordena, int dim, char tablero_disparos[], int f, int c)
{
        if(c>0)&(tablero_disparos [dim-1]=='?')
        {tablero_disparos[dim-1]='.';}
}
```

Actualitza la casella [coordena-1] amb aigua si compleix les condicions del taulell.

Joc de proves

BARCOS_P:

Principal

Funció/cas	Descripció	Resultat esperat	Correcte
principal();	Crida a la funció realizar_partida per fer un NUM_PARTIDES a cada taulell i guarda les mitjanes dels trets	Realitza les partides. I a nd8, nd9 i nd10 han de estar les mitjanes emmagatzemades	OK
1	Si NUM_PARTIDES=0	nd8=0; nd9=0; nd10=0;	OK
2	Si NUM_PARTIDES=150	nd8=38; nd9=48; nd10=62;	OK
3	Si NUM_PARTIDES=500	nd8=38; nd9=49; nd10=63;	OK

Realizar_partidas

Funció/cas	Descripció	Resultat esperat	Correcte
realizar_partidas();	Fa un NUM_PARTIDES de un tipus de dimenció en concret.	Realitzar les partides.	OK
	Calcula la mitja de tret fets per partida en aquesta dimensió i ho retorna a principal();	Calcular correctament la mitja i retornar-la.	OK
1	Si dim=8	Retorna 38	OK
2	Si dim=9	Retorna 48	OK
3	Si dim=10	Retorna 62	OK

B_inicialitza_dispars

Funció/cas	Descripció	Resultat esperat	Correcte
B_inicialitza_dispars();	Ompli el taulell de dispars amb ‘?’	El taulell ple de ‘?’	OK

BARCOS_J:

Jugar

Funció/cas	Descripció	Resultat esperat	Correcte
jugar();	Realitza una partida al complet per a un determinat tipus de dimensió.	Realitzar la partida correctament.	OK
	Retorna el total de tret de la partida	Retornar els trets fets a partir de B_num_disparos.	OK
1	Si dim=8	Retorna 36-40	OK
2	Si dim=9	Retorna 45-50	OK
3	Si dim=10	Retorna 60-65	OK

Fer_un_tret

Funció/cas	Descripció	Resultat esperat	Correcte
fer_un_tret();	Calcula les coordenades del pròxim tret, a partir del tret anterior.	Retornar el pròxim tret correctament.	OK
1	Si el tret anterior es aigua	Tir random	OK
2	Si el tret anterior es tocat	Tir tocat al mateix vaixell	OK
3	Si el tret anterior es enfonsat	Tir random	OK

Efectuar_disparo

Funció/cas	Descripció	Resultat esperat	Correcte
efectuar_disparo();	Actualitza els dos taulell a partir del llançament seleccionat per fer_un_tret.	Actualitza el taulell de vaixells.	OK
		Actualitza el taulell de dispars de forma anàloga al de vaixells.	OK
1	Si el tret anterior es aigua	Fica '.' al de dispars	OK
2	Si el tret anterior es tocat	Fica '@' al de dispars i '.' a les diagonals	OK
3	Si el tret anterior es enfonsat	Marca el vaixell i l'envolta de '.'	OK

Voltejar_aigua

Funció/cas	Descripció	Resultat esperat	Correcte
voltejar_aigua() ;	Si s'ha enfonsat un vaixell el volteja amb aigua '.'	El vaixell envoltat correctament.	OK

Coordena_*d_*

Funció/cas	Descripció	Resultat esperat	Correcte
coordena_*d_*();	Depenent de la funció coordena concreta marca una casella determina del voltant del vaixell. si es: ad -> +dim si es: sd -> -dim si es: a -> +1 si es: s <> -1	El vaixell envoltat correctament.	OK
1	coordena_sd_s	Ficar '.' a la casella de fila(-1) i columna(-1) de vaixell	OK
2	coordena_sd	Ficar '.' a la casella de fila(-1) de vaixell	OK
3	coordena_sd_a	Ficar '.' a la casella de fila(-1) i columna(+1) de vaixell	OK
4	coordena_s	Ficar '.' a la casella de columna(-1) de vaixell	OK
5	coordena_a	Ficar '.' a la casella de columna(+1) de vaixell	OK
6	coordena_ad_s	Ficar '.' a la casella de fila(+1) i columna(-1) de vaixell	OK
7	coordena_ad	Ficar '.' a la casella de fila(+1) de vaixell	OK
8	coordena_ad_a	Ficar '.' a la casella de fila(+1) i columna(+1) de vaixell	OK