```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "vecinos.h"
#include <math.h>
#ifndef VACIO
#define VACIO 0
#endif
#define NOVACIO 1
int AbrirArchivo(char* path, FILE **fp) {
    *fp = fopen(path, "r");
    if(*fp == NULL)
        return 1;
    return 0;
}

int CrearArchivoOutput(char *archivo, FILE **fop) {
    if((*fop = fopen(archivo, "w+b")) == NULL) {
        /*Tiro error de apertura de archivo*/
        return 1;
    }
    return 0;
}

void CerrarArchivo(FILE *fp) {
    fclose(fp);
}

unsigned int getPosicion(unsigned int i, unsigned int j, unsigned int N){
    return i*N+j;
}

void MostrarUso(){
    printf("Uso:\n");
    printf(" conway -h\n conway -V\n conway i M N inputfile [-o outputprefix]\n");
    printf("Opciones:\n -h, --help\t Imprime este mensaje\n");
    printf(" -V, --version\t Da la versiÃ³n del programa\n");
    printf(" -o\tPrefijo de los archivos de salida\n");
}

void ImprimirError(char* msj){
    fprintf(stderr,"%s",msj);
}

void InvocacionIncorrecta(){
    ImprimirError("InvocaciÃ³n del programa incorrecta\n");
}

void GrabarArchivoSalida(unsigned char *matriz, unsigned int M, unsigned int N,
char *prefijoSalida, unsigned int j) {
    unsigned int a = 0, b = 0,c = 0, d = 0;
    FILE *archivoSalida;
    int nDigits;
    char buf[5];
    if(j > 0)
        nDigits = floor(log10(j)) + 1;
    else
        nDigits = 1;

    char *nombreArchivoSalida = malloc(strlen(prefijoSalida)+(sizeof(char)*5)+si
zeof(char)*nDigits);
    strcpy(nombreArchivoSalida,prefijoSalida);
    strcat(nombreArchivoSalida,"_");
    sprintf(buf, "%u", j);
    strcat(nombreArchivoSalida,buf),
    strcat(nombreArchivoSalida,".pbm");
    if(CrearArchivoOutput(nombreArchivoSalida,&archivoSalida) == 0){
        printf("Grabando estado %s\n",nombreArchivoSalida);
        //Grabar Encabezado de archivo
        fprintf(archivoSalida,"P1\n");
        fprintf(archivoSalida,"%u %u\n",M*16,N*16);
```

```
70          //Recorrer Matriz y grabar archivo
71          for(a = 0;a < M;a++){
72              for (d = 0; d < 16; d++){
73                  for(b = 0;b < N;b++) {
74                      //Grabo posicion
75                      for (c = 0; c < 16; c++)
76                          fprintf(archivoSalida,"%u",matriz[getPosicion(a,b,N)]);
77                  }
78                  fprintf(archivoSalida,"\n");
79              }
80          }
81      }
82      CerrarArchivo(archivoSalida);
83      free(nombreArchivoSalida);
84  }
85
86  void abortarCargaMatriz(char* mensaje, unsigned char* matriz, FILE* archivo) {
87      ImprimirError(mensaje);
88      free(matriz);
89      CerrarArchivo(archivo);
90      fclose(stdout);
91      fclose(stderr);
92      exit(-1);
93  }
94
95  unsigned char obtenerNumeroFila(unsigned char* matriz, FILE *archivo, unsigned i
    nt M) {
96      unsigned char fila = 0;
97      fscanf(archivo, "%hhu", &fila);
98      if (fgetc(archivo) ≠ ' ') {
99          abortarCargaMatriz("\n\nError en el formato del archivo de entrada\n\n", matriz, archivo
    );
100     }
101     int proximoCaracter = fgetc(archivo);
102     if (¬((proximoCaracter ≥ '0') ∧ (proximoCaracter ≤ '9'))) {
103         abortarCargaMatriz("\n\nError en el formato del archivo de entrada\n\n", matriz, archivo
    );
104     }
105     fseek(archivo, -1, SEEK_CUR);
106     if (fila > M-1) {
107         abortarCargaMatriz("\n\nUn numero de fila en el archivo de entrada es mayor que el maximo perm
    itido\n\n", matriz, archivo);
108     }
109     return fila;
110 }
111
112 unsigned char obtenerNumeroColumna(unsigned char* matriz, FILE *archivo, unsigne
    d int N) {
113     unsigned char columna = 0;
114     fscanf(archivo, "%hhu", &columna);
115     if (fgetc(archivo) ≠ '\n') {
116         abortarCargaMatriz("\n\nError en el formato del archivo de entrada\n\n", matriz, archivo
    );
117     }
118     int proximoCaracter = fgetc(archivo);
119     if ((¬((proximoCaracter ≥ '0') ∧ (proximoCaracter ≤ '9'))) ∧ (proximoCaracte
    r ≠ EOF)) {
120         abortarCargaMatriz("\n\nError en el formato del archivo de entrada\n\n", matriz, archivo
    );
121     }
122     if (proximoCaracter ≠ EOF) {
123         fseek(archivo, -1, SEEK_CUR);
124     }
125     if (columna > N-1) {
126         abortarCargaMatriz("\n\nUn numero de columna en el archivo de entrada es mayor que el maximo
    permitido\n\n", matriz, archivo);
127     }
128     return columna;
129 }
130
131 void cargarMatriz(unsigned char* matriz, FILE *archivo, unsigned int M, unsigned
```

```
 int N) {
    rewind(archivo);
    while (¬feof(archivo)) {
        unsigned char fila = obtenerNumeroFila(matriz, archivo, M);
        unsigned char columna = obtenerNumeroColumna(matriz, archivo, N);
        matriz[fila*N + columna] = NOVACIO;
    }
}

int main(int argc, char *argv[]) {
    unsigned int i = 0,M = 0,N = 0,a = 0,n = 0,v,pos,j,b;
    char* prefijoSalida;
    FILE *fp;
    unsigned char* matriz;
    //Lectura y carga de argumentos
    if (argc < 5){
        if(argc ≡ 2){
            if (strcmp(argv[1],"−V") ≡ 0∨ strcmp(argv[1],"−−version") ≡ 0) {
                printf("Version 2.0\n");
                return 0;
            } else if (strcmp(argv[1],"−h") ≡ 0∨ strcmp(argv[1],"−−help") ≡ 0) {
                MostrarUso();
                return 0;
            } else {
                InvocacionIncorrecta();
                MostrarUso();
                return 1;
            }
        } else {
            InvocacionIncorrecta();
            MostrarUso();
            return 1;
        }
    } else if (argc ≡ 5 ∨ argc ≡ 7) {
        i = (unsigned int)atoi(argv[1]); //cantidad de ciclos
        if (i ≡ 0){
            ImprimirError("La cantidad de estados (i) debe ser un nÃºmero mayor a 0.\r\n");
            return 1;
        }
        M = (unsigned int)atoi(argv[2]);
        if (M ≡ 0){
            ImprimirError("La cantidad de filas (M) debe ser un nÃºmero mayor a 0.\r\n");
            return 1;
        }

        N = (unsigned int)atoi(argv[3]);
        if (N ≡ 0){
            ImprimirError("El tamaÃ±o de las columnas (N) debe ser un nÃºmero mayor a 0.\r\n");
            return 1;
        }

    } else {
        InvocacionIncorrecta();
        MostrarUso();
        return 1;
    }
    //Argumentos opcionales
    prefijoSalida = argv[4];
    if(argc ≡ 7){
        if(strcmp(argv[5],"−o") ≡ 0){
            prefijoSalida = argv[6];
            if(strcmp(prefijoSalida,"") ≡ 0){
                InvocacionIncorrecta();
                ImprimirError("El prefijo de los archivos de salida no puede ser vacio.");
                MostrarUso();
                return 1;
            }
        } else {
            InvocacionIncorrecta();
            MostrarUso();
            return 1;
```

```
        }
    }
    //Carga inicial
    n = AbrirArchivo(argv[4], &fp);
    if(n ≡ 1){
        //No se pudo abrir el archivo
        fprintf(stderr, "El archivo %s no pudo ser abierto.\r\n", argv[4]);
        return 1;
    }

    // Creo matriz
    matriz = malloc(M*N*sizeof(unsigned char));

    for(a = 0; a < M*N; a++){
        matriz[a] = VACIO;
    }

    //Cargar matriz con archivo de entrada
    printf("%s/n","Leyendo estado inicial...");
    cargarMatriz(matriz, fp, M, N);

    GrabarArchivoSalida(matriz,M,N,prefijoSalida,0);
    //Recorro las iteraciones
    for(j = 0; j < i; j++){
        unsigned char* copia = malloc(M*N*sizeof(unsigned char));
        for (a = 0; a < M; a++){
            for(b = 0; b < N; b++){
                //Recorro los casilleros
                v = vecinos(matriz, a, b, M, N);
                pos = getPosicion(a,b,N);
                copia[pos] = matriz[pos];
                if(matriz[pos] ≡ VACIO){
                    //Estaba muerta
                    if (v ≡ 3){
                        copia[pos] = NOVACIO;
                    }
                } else {
                    //Estaba viva
                    if (v < 2 ∨ v > 3){
                        //Muere
                        copia[pos] = VACIO;
                    }
                }
            }
        }
        free(matriz);
        matriz = copia;
        GrabarArchivoSalida(matriz,M,N,prefijoSalida,j+1);
    }
    printf("Listo\r\n");
    free(matriz);
    CerrarArchivo(fp);
    fclose(stdout);
    fclose(stderr);
    return 0;
}
```

```c
#ifndef VACIO
#define VACIO 0
#endif
unsigned int vecinos(unsigned char *a,unsigned int i, unsigned int j, unsigned int M, unsigned int N){
    unsigned int v = 0;
    //Modelo Toroidal
    if(i ≠ 0){
        //Chequeo arriba
        if (a[(i-1)*N+j] ≠ VACIO){
            v++;
        }
        //arriba a la izquierda
        if (j ≠ 0){
            if (a[(i-1)*N+(j-1)] ≠ VACIO){
                v++;
            }
        } else {
            if (a[(i-1)*N+(N-1)] ≠ VACIO){
                v++;
            }
        }
        if (j ≠ (N-1)){
            //arriba a la derecha
            if (a[(i-1)*N+(j+1)] ≠ VACIO){
                v++;
            }
        } else {
            if (a[(i-1)*N] ≠ VACIO){
                v++;
            }
        }
    } else {
        //Chequeo "arriba"
        if (a[(M-1)*N+j] ≠ VACIO){
            v++;
        }
        //arriba a la izquierda
        if (j ≠ 0){
            if (a[(M-1)*N+(j-1)] ≠ VACIO){
                v++;
            }
        } else {
            if (a[(M-1)*N+(N-1)] ≠ VACIO){
                v++;
            }
        }
        //arriba a la derecha
        if (j ≠ (N-1)){
            if (a[(M-1)*N+(j+1)] ≠ VACIO){
                v++;
            }
        } else {
            if (a[(M-1)*N] ≠ VACIO){
                v++;
            }
        }
    }
    if (i ≠ (M-1)){
        //Chequeo abajo
        if (a[(i+1)*N+j] ≠ VACIO){
            v++;
        }
        //abajo a la izquierda
        if (j ≠ 0){
            if (a[(i+1)*N+(j-1)] ≠ VACIO){
                v++;
            }
        } else {
            if (a[(i+1)*N+(N-1)] ≠ VACIO){
                v++;
```

```
            }
        }
        //abajo a la derecha
        if (j ≠ (N-1)){
            if (a[(i+1)*N+(j+1)] ≠ VACIO){
                v++;
            }
        } else {
            if (a[(i+1)*N] ≠ VACIO){
                v++;
            }
        }
    } else {
        //Chequeo abajo
        if (a[j] ≠ VACIO){
            v++;
        }
        //abajo a la izquierda
        if (j ≠ 0){
            if (a[j-1] ≠ VACIO){
                v++;
            }
        } else {
            if (a[N-1] ≠ VACIO){
                v++;
            }
        }
        //abajo a la derecha
        if (j ≠ (N-1)){
            if (a[j+1] ≠ VACIO){
                v++;
            }
        } else {
            if (a[0] ≠ VACIO){
                v++;
            }
        }
    }
    //Chequeo izquierda
    if (j ≠ 0){
        if (a[i*N+(j-1)] ≠ VACIO){
            v++;
        }
    } else {
        if (a[i*N+(N-1)] ≠ VACIO){
            v++;
        }
    }
    //Chequeo derecha
    if (j ≠ (N-1)){
        if (a[i*N+(j+1)] ≠ VACIO){
            v++;
        }
    } else {
        if (a[i*N] ≠ VACIO){
            v++;
        }
    }
    return v;
}
```

```
#include <mips/regdef.h>
    .text
    .align  2
    .global vecinos
    .ent    vecinos
#define V 0
#define GP 8
#define FP 12
#define A 16

#define I 20
#define J 24
#define M 28
#define N 32
#ifndef VACIO
#define VACIO 0
#endif
vecinos:
    #ABI
    subu    sp,sp,16
    sw      $fp,FP(sp)
    sw      gp,GP(sp)
    move    $fp,sp

    sw      a0,A(sp)
    sw      a1,I(sp)
    sw      a2,J(sp)
    sw      a3,M(sp)

    sw      zero,V(sp)
    #Empieza la funciÃ³n
    lw      t0,I(sp)
    beq     t0,zero,esprimerafila
    # i != 0
    lw      t0,I(sp)
    addu    t1,t0,-1
    lw      t0,N(sp)
    mult    t1,t0
    mflo    t1
    lw      t0,J(sp)
    addu    t1,t1,t0
    lw      t0,A(sp)
    addu    t0,t1,t0
    lbu     t0,0(t0) #t0 = a[(i-1)*N+j]
    beq     t0,zero,verarrizq
    lw      t0,V(sp)
    addu    t0,t0,1
    sw      t0,V(sp)
verarrizq:
    lw      t0,J(sp)
    beq     t0,zero,verarrizqprimcol
    lw      t0,I(sp)
    addu    t1,t0,-1
    lw      t0,N(sp)
    mult    t1,t0
    mflo    t1
    lw      t0,J(sp)
    addu    t1,t1,t0
    lw      t0,A(sp)
    addu    t0,t1,t0
    addu    t0,t0,-1
    lbu     t0,0(t0) #t0 = a[(i-1)*N+j-1]
    beq     t0,zero,verarrder
    lw      t0,V(sp)
    addu    t0,t0,1
    sw      t0,V(sp)
    b       verarrder
verarrizqprimcol:
    lw      t0,I(sp)
    addu    t1,t0,-1
    lw      t0,N(sp)
```

```
72      mult    t1,t0
73      mflo    t1
74      lw      t0,N(sp)
75      addu    t1,t1,t0
76      lw      t0,A(sp)
77      addu    t0,t1,t0
78      addu    t0,t0,-1
79      lbu     t0,0(t0) #t0 = a[(i-1)*N+N-1]
80      beq     t0,zero,verarrder
81      lw      t0,V(sp)
82      addu    t0,t0,1
83      sw      t0,V(sp)
84  verarrder:
85      lw      t0,N(sp)
86      addu    t1,t0,-1
87      lw      t0,J(sp)
88      beq     t0,t1,verarrderultcol
89      lw      t0,I(sp)
90      addu    t1,t0,-1
91      lw      t0,N(sp)
92      mult    t1,t0
93      mflo    t1
94      lw      t0,J(sp)
95      addu    t1,t1,t0
96      lw      t0,A(sp)
97      addu    t0,t1,t0
98      addu    t0,t0,1
99      lbu     t0,0(t0) #t0 = a[(i-1)*N+j+1]
100     beq     t0,zero,verultimafila
101     lw      t0,V(sp)
102     addu    t0,t0,1
103     sw      t0,V(sp)
104     b   verultimafila
105 verarrderultcol:
106     lw      t0,I(sp)
107     addu    t1,t0,-1
108     lw      t0,N(sp)
109     mult    t1,t0
110     mflo    t1
111     lw      t0,A(sp)
112     addu    t0,t1,t0
113     lbu     t0,0(t0) #t0 = a[(i-1)*N]
114     beq     t0,zero,verultimafila
115     lw      t0,V(sp)
116     addu    t0,t0,1
117     sw      t0,V(sp)
118     b       verultimafila
119 esprimerafila:
120     lw      t0,M(sp)
121     addu    t1,t0,-1
122     lw      t0,N(sp)
123     mult    t1,t0
124     mflo    t1
125     lw      t0,J(sp)
126     addu    t1,t1,t0
127     lw      t0,A(sp)
128     addu    t0,t1,t0
129     lbu     t0,0(t0) #t0 = a[(M-1)*N+j]
130     beq     t0,zero,verarribaizqpf
131     lw      t0,V(sp)
132     addu    t0,t0,1
133     sw      t0,V(sp)
134 verarribaizqpf:
135     lw      t0,J(sp)
136     beq     t0,zero,verarribaizqprimcolprimfila
137     lw      t0,M(sp)
138     addu    t1,t0,-1
139     lw      t0,N(sp)
140     mult    t1,t0
141     mflo    t1
142     lw      t0,J(sp)
```

```
143     addu    t1,t1,t0
144     lw      t0,A(sp)
145     addu    t0,t1,t0
146     addu    t0,t0,-1
147     lbu     t0,0(t0) #t0 = a[(M-1)*N+j-1]
148     beq     t0,zero,verarribaderpf
149     lw      t0,V(sp)
150     addu    t0,t0,1
151     sw      t0,V(sp)
152     b       verarribaderpf
153 verarribaizqprimcolprimfila:
154     lw      t0,M(sp)
155     addu    t1,t0,-1
156     lw      t0,N(sp)
157     mult    t1,t0
158     mflo    t1
159     lw      t0,N(sp)
160     addu    t1,t1,t0
161     lw      t0,A(sp)
162     addu    t0,t1,t0
163     addu    t0,t0,-1
164     lbu     t0,0(t0) #t0 = a[(M-1)*N+N-1]
165     beq     t0,zero,verarribaderpf
166     lw      t0,V(sp)
167     addu    t0,t0,1
168     sw      t0,V(sp)
169 verarribaderpf:
170     lw      t0,N(sp)
171     addu    t1,t0,-1
172     lw      t0,J(sp)
173     beq     t0,t1,verarribaderultcolpf
174     lw      t0,M(sp)
175     addu    t1,t0,-1
176     lw      t0,N(sp)
177     mult    t1,t0
178     mflo    t1
179     lw      t0,J(sp)
180     addu    t1,t1,t0
181     lw      t0,A(sp)
182     addu    t0,t1,t0
183     addu    t0,t0,1
184     lbu     t0,0(t0) #t0 = a[(M-1)*N+j+1]
185     beq     t0,zero,verultimafila
186     lw      t0,V(sp)
187     addu    t0,t0,1
188     sw      t0,V(sp)
189     b       verultimafila
190 verarribaderultcolpf:
191     lw      t0,M(sp)
192     addu    t1,t0,-1
193     lw      t0,N(sp)
194     mult    t1,t0
195     mflo    t1
196     lw      t0,A(sp)
197     addu    t0,t1,t0
198     lbu     t0,0(t0) #t0 = a[(M-1)*N]
199     beq     t0,zero,verultimafila
200     lw      t0,V(sp)
201     addu    t0,t0,1
202     sw      t0,V(sp)
203 verultimafila:
204     lw      t0,M(sp)
205     addu    t1,t0,-1
206     lw      t0,I(sp)
207     beq     t0,t1,verabajoultimafila
208     lw      t0,I(sp)
209     addu    t1,t0,1
210     lw      t0,N(sp)
211     mult    t1,t0
212     mflo    t1
213     lw      t0,J(sp)
```

```
214     addu    t1,t1,t0
215     lw      t0,A(sp)
216     addu    t0,t1,t0
217     lbu     t0,0(t0) #t0 = a[(i+1)*N+j]
218     beq     t0,zero,verabajoizqultfila
219     lw      t0,V(sp)
220     addu    t0,t0,1
221     sw      t0,V(sp)
222 verabajoizqultfila:
223     lw      t0,J(sp)
224     beq     t0,zero,verabajoizqprimeracol
225     lw      t0,I(sp)
226     addu    t1,t0,1
227     lw      t0,N(sp)
228     mult    t1,t0
229     mflo    t1
230     lw      t0,J(sp)
231     addu    t1,t1,t0
232     lw      t0,A(sp)
233     addu    t0,t1,t0
234     addu    t0,t0,-1
235     lbu     t0,0(t0) #t0 = a[(i+1)*N+(j-1)]
236     beq     t0,zero,verabajoderechaultfil
237     lw      t0,V(sp)
238     addu    t0,t0,1
239     sw      t0,V(sp)
240     b       verabajoderechaultfil
241 verabajoizqprimeracol:
242     lw      t0,I(sp)
243     addu    t1,t0,1
244     lw      t0,N(sp)
245     mult    t1,t0
246     mflo    t1
247     lw      t0,N(sp)
248     addu    t1,t1,t0
249     lw      t0,A(sp)
250     addu    t0,t1,t0
251     addu    t0,t0,-1
252     lbu     t0,0(t0) #t0 = a[(i+1)*N+(N-1)]
253     beq     t0,zero,verabajoderechaultfil
254     lw      t0,V(sp)
255     addu    t0,t0,1
256     sw      t0,V(sp)
257 verabajoderechaultfil:
258     lw      t0,N(sp)
259     addu    t1,t0,-1
260     lw      t0,J(sp)
261     beq     t0,t1,verproximafilacol0
262     lw      t0,I(sp)
263     addu    t1,t0,1
264     lw      t0,N(sp)
265     mult    t1,t0
266     mflo    t1
267     lw      t0,J(sp)
268     addu    t1,t1,t0
269     lw      t0,A(sp)
270     addu    t0,t1,t0
271     addu    t0,t0,1
272     lbu     t0,0(t0) #t0 = a[(i+1)*N+(j+1)]
273     beq     t0,zero,verizquierda
274     lw      t0,V(sp)
275     addu    t0,t0,1
276     sw      t0,V(sp)
277     b       verizquierda
278 verproximafilacol0:
279     lw      t0,I(sp)
280     addu    t1,t0,1
281     lw      t0,N(sp)
282     mult    t1,t0
283     mflo    t1
284     lw      t0,A(sp)
```

```
285      addu    t0,t1,t0
286      lbu     t0,0(t0) #t0 = a[(i+1)*N]
287      beq     t0,zero,verizquierda
288      lw      t0,V(sp)
289      addu    t0,t0,1
290      sw      t0,V(sp)
291      b       verizquierda
292  verabajoultimafila:
293      lw      t1,A(sp)
294      lw      t0,J(sp)
295      addu    t0,t1,t0
296      lbu     t0,0(t0) #t0 = a[j]
297      beq     t0,zero,vercerocolanterior
298      lw      t0,V(sp)
299      addu    t0,t0,1
300      sw      t0,V(sp)
301  vercerocolanterior:
302      lw      t0,J(sp)
303      beq     t0,zero,verceroabajodetodo
304      lw      t1,A(sp)
305      lw      t0,J(sp)
306      addu    t0,t1,t0
307      addu    t0,t0,-1
308      lbu     t0,0(t0) #t0 = a[j-1]
309      beq     t0,zero,ultfilaabajoder
310      lw      t0,V(sp)
311      addu    t0,t0,1
312      sw      t0,V(sp)
313      b       ultfilaabajoder
314  verceroabajodetodo:
315      lw      t1,A(sp)
316      lw      t0,N(sp)
317      addu    t0,t1,t0
318      addu    t0,t0,-1
319      lbu     t0,0(t0) #t0 = a[N-1]
320      beq     t0,zero,ultfilaabajoder
321      lw      t0,V(sp)
322      addu    t0,t0,1
323      sw      t0,V(sp)
324  ultfilaabajoder:
325      lw      t0,N(sp)
326      addu    t1,t0,-1
327      lw      t0,J(sp)
328      beq     t0,t1,ver00
329      lw      t1,A(sp)
330      lw      t0,J(sp)
331      addu    t0,t1,t0
332      addu    t0,t0,1
333      lbu     t0,0(t0) #t0 = a[j+1]
334      beq     t0,zero,verizquierda
335      lw      t0,V(sp)
336      addu    t0,t0,1
337      sw      t0,V(sp)
338      b       verizquierda
339  ver00:
340      lw      t0,A(sp)
341      lbu     t0,0(t0) #to = a[0]
342      beq     t0,zero,verizquierda
343      lw      t0,V(sp)
344      addu    t0,t0,1
345      sw      t0,V(sp)
346  verizquierda:
347      lw      t0,J(sp)
348      beq     t0,zero,izqprimeracol
349      lw      t1,I(sp)
350      lw      t0,N(sp)
351      mult    t1,t0
352      mflo    t1
353      lw      t0,J(sp)
354      addu    t1,t1,t0
355      lw      t0,A(sp)
```

```
356      addu    t0,t1,t0
357      addu    t0,t0,-1
358      lbu     t0,0(t0) #t0 = a[i*N+(j-1)]
359      beq     t0,zero,verderecha
360      lw      t0,V(sp)
361      addu    t0,t0,1
362      sw      t0,V(sp)
363      b       verderecha
364  izqprimeracol:
365      lw      t1,I(sp)
366      lw      t0,N(sp)
367      mult    t1,t0
368      mflo    t1
369      lw      t0,N(sp)
370      addu    t1,t1,t0
371      lw      t0,A(sp)
372      addu    t0,t1,t0
373      addu    t0,t0,-1
374      lbu     t0,0(t0) #t0 = a[i*N+(N-1)]
375      beq     t0,zero,verderecha
376      lw      t0,V(sp)
377      addu    t0,t0,1
378      sw      t0,V(sp)
379  verderecha:
380      lw      t0,N(sp)
381      addu    t1,t0,-1
382      lw      t0,J(sp)
383      beq     t0,t1,derultimafila
384      lw      t1,I(sp)
385      lw      t0,N(sp)
386      mult    t1,t0
387      mflo    t1
388      lw      t0,J(sp)
389      addu    t1,t1,t0
390      lw      t0,A(sp)
391      addu    t0,t1,t0
392      addu    t0,t0,1
393      lbu     t0,0(t0) #t0 = a[i*N+(j+1)]
394      beq     t0,zero,finvecinos
395      lw      t0,V(sp)
396      addu    t0,t0,1
397      sw      t0,V(sp)
398      b       finvecinos
399  derultimafila:
400      lw      t1,I(sp)
401      lw      t0,N(sp)
402      mult    t1,t0
403      mflo    t1
404      lw      t0,A(sp)
405      addu    t0,t1,t0
406      lbu     t0,0(t0) #t0 = a[i*N]
407      beq     t0,zero,finvecinos
408      lw      t0,V(sp)
409      addu    t0,t0,1
410      sw      t0,V(sp)
411
412  finvecinos:
413      lw      v0,V(sp)
414      move    sp,$fp
415      lw      $fp,FP(sp)
416      lw      gp,GP(sp)
417      addu    sp,sp,16
418      j       ra
419  .end vecinos
```