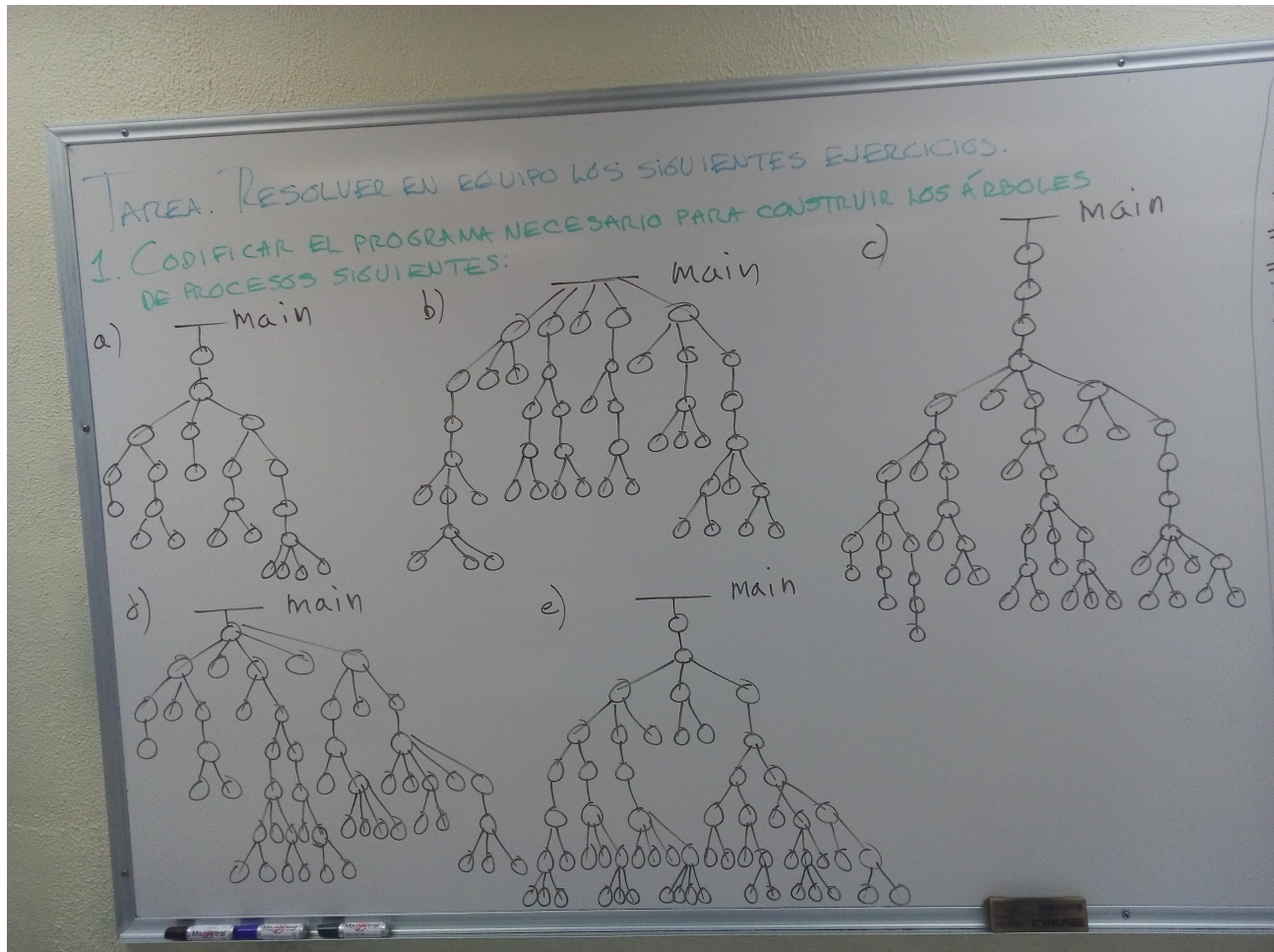


1. Codificar el programa necesario para construir los árboles de procesos siguientes:



2. Árbol A)

```

1  #include <stdio.h>
2  #include <unistd.h>
3  #include <stdlib.h>
4
5  int main(void){
6      int id_proc, i, j, k, l, m, w, x, y;
7
8      for(i=0; i<2; i++){//2 procesos
9          if((id_proc=fork())==0){//Creo 2 procesos verticales
10             printf("Soy el proceso hijo\n");
11
12             if(i==1){//Segundo arbol vertical
13                 for(j=0; j<3; j++){//3 procesos
14                     if((id_proc=fork())==0){//Creo 3 procesos horizontales
15                         printf("Soy el proceso hijo\n");
16
17                         if(j==0){//Primer proceso - izquierda

```

```

18     for(k=0; k<2; k++){//2 procesos
19         if((id_proc=fork())==0){//Creo 2 procesos horizontales
20             printf("Soy el proceso hijo\n");
21
22             if(k==0){//Primer proceso - izquierda
23                 if((id_proc=fork())==0){//1 proceso vertical
24                     printf("Soy el proceso hijo\n");
25                     exit(0);
26                 }//ACABAMOS
27             }
28             else if(k==1){//Segundo proceso - derecha
29                 if((id_proc=fork())==0){//1 proceso vertical
30                     printf("Soy el proceso hijo\n");
31
32                     for(l=0; l<2; l++){//2 procesos
33                         if((id_proc=fork())==0){//Creamos 2 procesos horizontales
34                             printf("Soy el proceso hijo\n");
35                             exit(0);
36                         }//ACABAMOS
37                     }
38                     exit(0);
39                 }
40             }
41             exit(0);
42         }
43     }
44 }
45 else if(j==1){//Segundo proceso - centro
46     if((id_proc=fork())==0){//1 proceso vertical
47         printf("Soy el proceso hijo\n");
48         exit(0);
49     }//ACABAMOS
50 }
51 }
52 else if(j==2){//Tercer proceso - derecha
53     for(m=0; m<2; m++){//2 procesos
54         if((id_proc=fork())==0){//Creo 2 procesos horizontales
55             printf("Soy el proceso hijo\n");
56
57             if(m==0){//Primer proceso - izquierda
58                 if((id_proc=fork())==0){//1 proceso vertical
59                     printf("Soy el proceso hijo\n");
60
61                     for(l=0; l<2; l++){//2 procesos
62                         if((id_proc=fork())==0){//Creo 2 procesos horizontales
63                             printf("Soy el proceso hijo\n");
64                             exit(0);
65                         }//ACABAMOS
66                     }
67                     exit(0);
68                 }
69             }
70             else if(m==1){//Segundo proceso - derecha
71                 for(w=0; w<2; w++){//2 procesos
72                     if((id_proc=fork())==0){//Creamos 2 procesos verticales
73                         printf("Soy el proceso hijo\n");
74                         if(w==1){//Segundo proceso
75                             for(x=0; x<4; x++){//4 procesos
76                                 if((id_proc=fork())==0){//Creamos 4 procesos horizontales
77                                     printf("Soy el proceso hijo\n");
78                                     exit(0);
79                                 }//ACABAMOS
80                             }
81                         }
82                     }
83                 }
84                 else{
85                     break;
86                     exit(0);
87                 }
88             }
89         }
90     }
91 }

```

```

87         }
88     }
89     exit(0);
90 }
91 }
92 }
93 exit(0);
94 }
95 }
96 }
97 }
98 else{
99     break;
100    exit(0);
101 }
102 }
103 exit(0);
104 }

```

3. Árbol B)

```

1  // 04 de Octubre 2018
2  // Para generar muchos procesos HIJO
3  #include<unistd.h>
4  #include<stdio.h>
5  #include<stdlib.h>
6  #include <sys/types.h>
7  int i = 10;
8
9  void hijosVertical(int n)
10 {
11     if (n == 0)
12         exit(0);
13     printf("proceso padre: %d\n", getppid());
14     printf("proceso actual %d\n", getpid());
15     int id_proc = fork();
16     if (id_proc == 0)
17     {
18         printf("proceso hijo %d\n", getpid());
19         nodosEnColados(--n);
20         exit(0);
21     }
22     wait(0);
23 }
24
25 void hijosHorizontal(int n) {
26     printf("proceso actual %d\n", getpid());
27     int id_proc=0;
28     for (int i = 0; i < n; i++)
29     {
30         if ((id_proc = fork()) == 0)
31         {
32             printf("proceso padre: %d\n", getppid());
33             printf("proceso hijo: %d\n", getpid());
34             exit(0);
35         }
36         wait(0);
37     }
38 }
39 int main(void)
40 {
41     int id_proc;
42     for(i = 0; i<5; i++) // CREAR 5 PROCESOS HORIZONTALES
43     {
44         id_proc = fork();

```

```

45     if(id_proc == 0)
46     {
47         printf("Soy el proceso hijo %i\n", i);
48         if(i == 0) // PARA EL PRIMER HIJO
49         {
50             for(j = 0; j<3; j++) // CREAM 3 PROCESOS HORIZONTALES
51             {
52                 id_proc = fork();
53                 if(id_proc == 0)
54                 {
55                     printf("Soy el proceso hijo\n");
56                     if(j == 0) // PARA EL PRIMER HIJO
57                     {
58                         for(k = 0; k<2; k++) // CREA 2 PROCESOS VERTICALES
59                         {
60                             id_proc = fork();
61                             if(id_proc == 0)
62                             {
63                                 printf("Soy el proceso hijo %i\n", i);
64                                 if(k == 1) // EN EL ULTIMO PROCESO
65                                 {
66                                     for(l = 0; l<3; l++) // CREA PROCESOS HORIZONTALES
67                                     {
68                                         id_proc = fork();
69                                         if(id_proc == 0)
70                                         {
71                                             if(l == 1)
72                                             {
73
74                                                 id_proc = fork();
75                                                 if(id_proc == 0)
76                                                 {
77                                                     hijosHorizontal(3);
78                                                 }
79                                             }
80                                         }
81                                     }
82                                     exit(0);
83                                 }
84                             }
85                             else
86                             {
87                                 break; // Sale del ciclo
88                             }
89                         }
90                     }
91                 } // Fin if - primer hijo
92             }
93             exit(0);
94         }
95         exit(0);
96     } // Fin if - Hijo 0
97
98     if(i == 1) // PARA EL SEGUNDO HIJO
99     {
100         id_proc = fork(); // Crear un hijo
101         if(id_proc == 0)
102         {
103             for(j = 0; j<2; j++) // Crear dos hijos horizontales
104             {
105                 id_proc = fork();
106                 if(id_proc == 0)
107                 {
108                     if(j == 0) // Para el primer hijo
109                     {
110                         id_proc = fork();
111                         if(id_proc == 0) // Crear un hijo
112                         {
113                             hijosHorizontal(2);

```

```

114         }
115     }
116     if(j == 1) // Para el segundo hijo
117     {
118         id_proc = fork();
119         if(id_proc == 0) // Crear un hijo
120         {
121             hijosHorizontal(2);
122         }
123     }
124 }
125 }
126 exit(0);
127 }
128 }
129 if(j == 2) // PARA EL TERCER HIJO
130 {
131     exit(0); // No tiene hijos
132 }
133 if(j == 3) // PARA EL CUARTO HIJO
134 {
135     id_proc = fork();
136     if(id_proc == 0) // Crear un hijo
137     {
138         for(k = 0; k<2; k++) // Crear dos hijos
139         {
140             id_proc = fork();
141             if(id_proc == 0)
142             {
143                 if(k == 0) // Para el hijo 1
144                 {
145                     exit(0);
146                 }
147                 if (k == 1) // Para el hijo 2
148                 {
149                     id_proc = fork();
150                     if(id_proc == 0)
151                     {
152                         hijosHorizontal(2);
153                     }
154                 }
155             }
156         }
157         exit(0);
158     }
159 }
160 if(j == 4)
161 {
162     for(k = 0; k<3; k++) // Crear 3 hijos
163     {
164         id_proc = fork();
165         if(id_proc == 0)
166         {
167             if(k == 0) // Para el hijo 1
168             {
169                 exit(0);
170             }
171             if(k == 1) // Para el hijo 2
172             {
173                 id_proc = fork();
174                 if(id_proc == 0) // Crear un hijo
175                 {
176                     hijosHorizontal(3);
177                 }
178             }
179             if(k == 2) // Para el hijo 3
180             {
181                 for(l = 0; l<2; l++) // Crea 2 hijos verticales
182                 {

```

```

183         id_proc = fork();
184         if(id_proc == 0)
185         {
186             if(l == 1)
187             {
188                 for(m = 0; m<3; m++)
189                 {
190                     id_proc = fork();
191                     if(id_proc == 0)
192                     {
193                         if(m == 0 || m == 2)
194                         {
195                             hijosHorizontal(2);
196                         }
197                         if(m == 1)
198                         {
199                             exit(0);
200                         }
201                     }
202                 }
203             }
204         }
205         else
206         {
207             break;
208         }
209     }
210 }
211 }
212 }
213 }
214     exit(0);
215 }
216 }
217 } // Fin 5 procesos horizontales
218 exit(0);
219 }

```

4. Árbol C)

```

1 //Pal jueves 12-Oct-2018
2 #include <stdio.h>
3 #include <stdlib.h>
4
5 int main(void) {
6     int id_proc, i, j, k, l, m, p;
7     for(i=0; i<4; i++)
8     {
9         id_proc=fork(); //1
10        if(id_proc==0) {
11            printf("\nSoy un proceso hijo")
12            if(i==3) {
13                for(j=0; j<4; j++) {
14                    id_proc=fork(); //2
15                    if(id_proc==0) {
16                        if(j==0) { //2.1
17                            id_proc=fork();
18                            if(id_proc==0) {
19                                for(k=0; k<3; k++) {
20                                    id_proc=fork();
21                                    if(id_proc==0) {
22                                        printf("\nSoy un proceso hijo");
23                                        if(k==0) { //2.1.1
24                                            id_proc=fork();
25                                            if(id_proc==0) {

```

```

26         for(l=0;l<3;l++){
27             id_proc=fork();
28             if(id_proc==0){
29                 printf("\nSoy un proceso hijo");
30                 if(l==0){//2.1.1.1
31                     id_proc=fork();
32                     if(id_proc==0){
33                         printf("\nSoy un proceso hijo");
34                         exit(0);
35                     }
36                 }
37             }
38             else if(l==1){//2.1.1.2
39                 printf("\nSoy un proceso hijo");
40                 for(m=0,m<2;m++){
41                     id_proc=fork();
42                     if(id_proc==0){
43                         printf("\nSoy un proceso hijo");
44                         //exit(0);
45                     }
46                     else{
47                         break;
48                     }
49                 }
50                 exit(0);
51             }
52             else if(l==2){//2.1.1.3
53                 printf("\nSoy un proceso hijo");
54                 for(m=0,m<3;m++){
55                     id_proc=fork();
56                     if(id_proc==0){
57                         printf("\nSoy un proceso hijo");
58                         exit(0);
59                     }
60                     else{
61                         printf("\nSoy un proceso padre");
62                         break;
63                     }
64                 }
65                 exit(0);
66             }
67             exit(0);
68         }
69         /*else{
70             printf("\nSoy un proceso padre");
71             //exit(0);
72         }*/
73     }
74 }
75 /*else{
76     printf("\nSoy un proceso padre");
77     //exit(0);
78 }*/
79 }
80 else if(k==1){//2.1.2
81     printf("\nSoy un proceso hijo");
82     exit(0);
83 }
84 else if(k==2){//2.1.3
85     id_proc=fork();
86     if(id_proc==0){
87         printf("\nSoy un proceso hijo");
88         for(l=0;l<2;l++){
89             id_proc=fork();
90             if(id_proc==0){
91                 if(l==0){//2.1.3.1
92                     printf("\nSoy un proceso hijo");
93                     exit(0);
94                 }

```

```

95         else if(l==1){//2.1.3.2
96             printf("\nSoy un proceso hijo");
97             for(m=0;m<2;m++){
98                 id_proc=fork();
99                 if(id_proc==0){
100                     printf("\nSoy un proceso hijo");
101                     exit(0);
102                 }
103                 /*else{
104                     printf("\nSoy un proceso padre");
105                 }*/
106             }
107             exit(0);
108         }
109     }
110     /*else{
111         printf("\nSoy un proceso padre");
112         //exit(0);
113     }*/
114 }
115 exit(0);
116 }
117 exit(0);
118 /*else{
119     printf("\nSoy un proceso padre");
120     //exit(0);
121 }*/
122 }
123 }
124 /*else{
125     printf("\nSoy un proceso padre");
126     //exit(0);
127 }*/
128 }
129 }
130 /*else{
131     printf("\nSoy un proceso padre");
132     //exit(0);
133 }*/
134 }
135 else if(j==1){//2.2
136     printf("\nSoy un proceso hijo");
137     exit(0);
138 }
139 else if(j==2){//2.3
140     id_proc=fork();
141     if(id_proc==0){
142         printf("\nSoy un proceso hijo");
143         for(k=0;k<2;k++){
144             id_proc=fork();
145             if(id_proc==0){
146                 if(k==0){//2.3.1
147                     printf("\nSoy un proceso hijo");
148                     exit(0);
149                 }
150                 else if(k==1){//2.3.2
151                     id_proc=fork();
152                     if(id_proc==0){
153                         printf("\nSoy un proceso hijo");
154                         for(l=0;l<3;l++){
155                             id_proc=fork();
156                             if(id_proc==0){
157                                 printf("\nSoy un proceso hijo");
158                                 if(l==0){//2.3.2.1
159                                     id_proc=fork();
160                                     if(id_proc==0){
161                                         printf("\nSoy un proceso hijo");
162                                         for(m=0;m<2;m++){
163                                             id_proc=fork();

```



```
164         if(id_proc==0){
165             printf("\nSoy un proceso hijo");
166             exit(0);
167         }
168         /*else{
169             printf("\nSoy un proceso padre");
170             }*/
171     }
172 }
173 exit(0);
174 /*else{
175     printf("\nSoy un proceso padre");
176     //exit(0);
177 }*/
178
179 }
180 else if(l==1){ //2.3.2.2
181     id_proc=fork();
182     if(id_proc==0){
183         printf("\nSoy un proceso hijo");
184         exit(0);
185     }
186     /*else{
187         printf("\nSoy un proceso padre");
188         //exit(0);
189     }*/
190
191 }
192 else if(l==2){ //2.3.2.3
193     id_proc=fork();
194     if(id_proc==0){
195         printf("\nSoy un proceso hijo");
196         for(m=0;m<3;m++){
197             id_proc=fork();
198             if(id_proc==0){
199                 printf("\nSoy un proceso hijo");
200                 exit(0);
201             }
202             /*else{
203                 printf("\nSoy un proceso padre");
204                 //exit(0);
205             }*/
206         }
207     }
208     exit(0);
209     /*else{
210         printf("\nSoy un proceso padre");
211         //exit(0);
212     }*/
213 }
214 }
215 /*else{
216     printf("\nSoy un proceso padre");
217     //exit(0);
218 }*/
219 }
220 }
221 exit(0);
222 /*else{
223     printf("\nSoy un proceso padre");
224     //exit(0);
225 }*/
226
227 }
228 }
229 exit(0);
230 /*else{
231     printf("\nSoy un proceso padre");
232     //exit(0);
```

```

233         */
234     }
235 }
236 exit(0);
237 /*else{
238     printf("\nSoy un proceso padre");
239     //exit(0);
240 }*/
241 }
242 else if(j==3){ //2.4
243     for(p=0;p<3;p++){
244         id_proc=fork();
245         if(id_proc==0){
246             if(p<2){ //2.4.1 y 2.4.2
247                 printf("\nSoy un proceso hijo");
248                 exit(0);
249             }
250             if(p==2){ //2.4.3
251                 printf("\nSoy un proceso hijo");
252                 for(k=0;k<3;k++){
253                     id_proc=fork();
254                     if(id_proc==0){
255                         printf("\nSoy un proceso hijo");
256                         if(k==3){
257                             for(l=0;l<4;l++){
258                                 id_proc=fork();
259                                 if(id_proc==0){
260                                     if(l==0 || l==2){ //2.4.3.1 y 2.4.3.3
261                                         printf("\nSoy un proceso hijo");
262                                         exit(0);
263                                     }
264                                     else if(l==1 || l==3){ //2.4.3.2 y 2.4.3.4
265                                         printf("\nSoy un proceso hijo");
266                                         for(m=0;m<2;m++){
267                                             id_proc=fork();
268                                             if(id_proc==0){
269                                                 printf("\nSoy un proceso hijo");
270                                                 exit(0);
271                                             }
272                                         }
273                                         /*else{
274                                             printf("\nSoy un proceso padre");
275                                             //exit(0);
276                                         }*/
277                                     }
278                                     exit(0);
279                                 }
280                             }
281                         }
282                         /*else{
283                             printf("\nSoy un proceso padre");
284                             //exit(0);
285                         }*/
286                     }
287                 }
288             }
289             exit(0);
290             /*else{
291                 printf("\nSoy un proceso padre");
292                 break;
293             }*/
294         }
295     }
296 }
297 }
298 exit(0);
299 /*else{
300     printf("\nSoy un proceso padre");
301     //exit(0);

```

```

302         } */
303     }
304
305     }
306 }
307 exit(0);
308 /*else{
309     printf("\nSoy un proceso padre");
310     //exit(0);
311 }*/
312 }
313 }
314 }
315 else{
316     break;
317     exit(0);
318 }
319 }
320 exit(0);
321 }

```

5. Árbol D)

```

1  #include <unistd.h>
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <math.h>
5
6  int main(int argc, char const *argv[])
7  {
8      int id_proc, index = 0;
9      if (fork() == 0)
10     {
11         exit(0);
12     }else{
13         // printf("First fork\n");
14         for (int i = 0; i < 3; ++i)
15         {
16             if(fork() == 0){
17                 index += pow(2,i);
18             }
19             if (index > 4)
20             {
21                 exit(0);
22             }
23         }
24         // printf("%i\n",index);
25         int aux1 = 0;
26         if (index == 0 || index == 4)
27         {
28             for (int i = 0; i < 2; ++i)
29             {
30                 if(fork() == 0){
31                     aux1 += pow(2,i);
32                 }
33                 if (aux1 > 2)
34                 {
35                     exit(0);
36                 }
37             }
38             // printf("%i:%i\n",index,aux1);
39             int aux2 = 0;
40             if (aux1 == 0 || aux1 == 2)
41             {
42                 if (fork() == 0)

```

```

43     {
44         exit(0);
45     }
46     // printf("%i:%i:%i\n",index,aux1,aux2);
47     int aux3 = 0;
48     if (index == 0 && aux1 == 2)
49     {
50         if(fork() == 0) {
51             aux3++;
52         }
53         // printf("%i:%i:%i:%i\n",index,aux1,aux2,aux3);
54     }else if (index == 4)
55     {
56         if (aux1 == 0)
57         {
58             if (fork() == 0)
59             {
60                 // printf("%i:%i:%i:%i\n",index,aux1,aux2,++aux3);
61                 int aux4 = 0;
62                 for (int i = 0; i < 2; ++i)
63                 {
64                     if(fork() == 0){
65                         aux4 += pow(2,i);
66                     }
67                 }
68                 // printf("%i:%i:%i:%i:%i\n",index,aux1,aux2,aux3,aux4);
69             }else{
70                 // printf("%i:%i:%i:%i\n",index,aux1,aux2,aux3);
71             }
72         }else if (aux1 == 2)
73         {
74             for (int i = 0; i < 3; ++i)
75             {
76                 if(fork() == 0){
77                     aux3 += pow(2,i);
78                 }
79                 if (aux3 > 4)
80                 {
81                     exit(0);
82                 }
83             }
84             // printf("%i:%i:%i:%i\n",index,aux1,aux2,aux3);
85             int aux4 = 0;
86             if (aux3 == 2)
87             {
88                 if (fork() == 0)
89                 {
90                     aux4++;
91                 }
92                 // printf("%i:%i:%i:%i:%i\n",index,aux1,aux2,aux3,aux4);
93             }else if (aux3 == 4)
94             {
95                 if (fork() == 0)
96                 {
97                     exit(0);
98                 }
99                 }else{
100                 // printf("%i:%i:%i:%i:%i\n",index,aux1,aux2,aux3,aux4);
101                 int aux5 = 0;
102                 for (int i = 0; i < 2; ++i)
103                 {
104                     if(fork() == 0){
105                         aux5 += pow(2,i);
106                     }
107                     if (aux5 > 2)
108                     {
109                         exit(0);
110                     }
111                 }

```

```

112         // printf("%i:%i:%i:%i:%i\n", index, aux1, aux2, aux3, aux4, aux5);
113     }
114 }
115 }
116 }
117 }
118 }
119 else if (index == 2)
120 {
121     if (fork() == 0)
122     {
123         // printf("%i:%i\n", index, ++aux1);
124         int aux2 = 0;
125         if (fork() == 0)
126         {
127             // printf("%i:%i:%i\n", index, aux1, ++aux2);
128             int aux3 = 0;
129             if (fork() == 0)
130             {
131                 exit(0);
132             }else{
133                 // printf("%i:%i:%i:%i\n", index, aux1, aux2, aux3);
134                 int aux4 = 0;
135                 if (fork() == 0)
136                 {
137                     // printf("%i:%i:%i:%i:%i\n", index, aux1, aux2, aux3, ++aux4);
138                     int aux5 = 0;
139                     if (fork() == 0)
140                     {
141                         aux5++;
142                     }
143                     // printf("%i:%i:%i:%i:%i:%i\n", index, aux1, aux2, aux3, aux4, aux5);
144                 }else{
145                     // printf("%i:%i:%i:%i:%i\n", index, aux1, aux2, aux3, aux4);
146                 }
147             }
148         }else{
149             // printf("%i:%i:%i\n", index, aux1, aux2);
150             int aux3 = 0;
151             if (fork() == 0)
152             {
153                 exit(0);
154             }else{
155                 // printf("%i:%i:%i:%i\n", index, aux1, aux2, aux3);
156                 int aux4 = 0;
157                 for (int i = 0; i < 2; ++i)
158                 {
159                     if(fork() == 0){
160                         aux4 += pow(2,i);
161                     }
162                     if (aux4 > 2)
163                     {
164                         exit(0);
165                     }
166                 }
167                 // printf("%i:%i:%i:%i:%i\n", index, aux1, aux2, aux3, aux4);
168                 if (aux4 == 0 || aux4 == 2)
169                 {
170                     int aux5 = 0;
171                     if (fork() == 0)
172                     {
173                         aux5++;
174                     }
175                     // printf("%i:%i:%i:%i:%i:%i\n", index, aux1, aux2, aux3, aux4, aux5);
176                 }
177             }
178         }
179     }else{
180         // printf("%i:%i\n", index, aux1);

```

```

181     }
182 }
183 }
184 return 1;
185 }

```

6. Árbol E)

```

1  #include <stdio.h>
2  #include <unistd.h>
3  #include <stdlib.h>
4
5  int main(void) {
6      int id_proc;
7
8      for (int i = 0; i < 2; i++)
9      {
10         if ((id_proc = fork()) == 0)
11         {
12             printf("Soy un proceso hijoA\n");
13             if (i == 1)
14             {
15                 for (int j = 0; j < 3; j++)
16                 {
17                     if ((id_proc = fork()) == 0)
18                     {
19                         printf("Soy un proceso hijoB\n");
20                         if (j == 0)
21                         {
22                             for (int k = 0; k < 3; k++)
23                             {
24                                 if ((id_proc = fork()) == 0)
25                                 {
26                                     printf("Soy un proceso hijoC\n");
27                                     if (k == 0)
28                                     {
29                                         for (int l = 0; l < 2; l++)
30                                         {
31                                             if ((id_proc = fork()) == 0)
32                                             {
33                                                 printf("Soy un proceso hijoD\n");
34                                                 if (l == 0)
35                                                 {
36                                                     if ((id_proc = fork()) == 0)
37                                                     {
38                                                         printf("Soy un proceso hijoE\n");
39                                                         for (int m = 0; m < 2; m++)
40                                                         {
41                                                             if ((id_proc = fork()) == 0)
42                                                             {
43                                                                 printf("Soy un proceso hijoF\n");
44                                                                 if (m == 0)
45                                                                 {
46                                                                     for (int n = 0; n < 3; n++)
47                                                                     {
48                                                                         if ((id_proc = fork()) == 0)
49                                                                         {
50                                                                             printf("Soy un proceso hijoG\n");
51                                                                             exit(0);
52                                                                         }
53                                                                     }
54                                                                 }
55                                                             else if (m == 1)
56                                                             {
57                                                                 exit(0);

```

```

58         }
59         exit(0);
60     }
61 }
62 exit(0);
63 }
64 }
65 else if (l == 1)
66 {
67     if ((id_proc = fork()) == 0)
68     {
69         printf("Soy un proceso hijoH\n");
70         for (int m = 0; m < 3; m++)
71         {
72             if ((id_proc = fork()) == 0)
73             {
74                 printf("Soy un proceso hijoI\n");
75                 if (m == 2)
76                 {
77                     for (int n = 0; n < 3; n++)
78                     {
79                         if ((id_proc = fork()) == 0)
80                         {
81                             printf("Soy un proceso hijoJ\n");
82                             exit(0);
83                         }
84                     }
85                 }
86                 else
87                 {
88                     exit(0);
89                 }
90                 exit(0);
91             }
92         }
93         exit(0);
94     }
95 }
96 exit(0);
97 }
98 }
99 }
100 else if (k == 1)
101 {
102     if ((id_proc = fork()) == 0)
103     {
104         printf("Soy un proceso hijoK\n");
105         if ((id_proc = fork()) == 0)
106         {
107             printf("Soy un proceso hijoL\n");
108             for (int l = 0; l < 4; l++)
109             {
110                 if ((id_proc = fork()) == 0)
111                 {
112                     printf("Soy un proceso hijoM\n");
113                     if (l == 3)
114                     {
115                         for (int m = 0; m < 4; m++)
116                         {
117                             if ((id_proc = fork()) == 0)
118                             {
119                                 printf("Soy un proceso hijoN\n");
120                                 exit(0);
121                             }
122                         }
123                     }
124                 }
125                 else
126                 {
127                     exit(0);
128                 }
129             }
130         }
131     }
132 }
133 }
134 }
135 }
136 }

```

```

127         }
128     }
129     exit(0);
130 }
131 else
132     exit(0);
133     exit(0);
134 }
135 else
136     exit(0);
137 }
138 else if (k == 2)
139 {
140     if ((id_proc = fork()) == 0)
141     {
142         printf("Soy un proceso hijoO\n");
143         exit(0);
144     }
145 }
146 exit(0);
147 }
148 }
149 }
150 else if (j == 1)
151 {
152     for (int k = 0; k < 2; k++)
153     {
154         if ((id_proc = fork()) == 0)
155         {
156             printf("Soy un proceso hijoP\n");
157             exit(0);
158         }
159     }
160 }
161 else if (j == 2)
162 {
163     if ((id_proc = fork()) == 0)
164     {
165         printf("Soy un proceso hijoQ\n");
166         for (int k = 0; k < 2; k++)
167         {
168             if ((id_proc = fork()) == 0)
169             {
170                 if (k == 0)
171                 {
172                     for (int l = 0; l < 2; l++)
173                     {
174                         if ((id_proc = fork()) == 0)
175                         {
176                             printf("Soy un proceso hijoR\n");
177                             if (l == 0)
178                             {
179                                 for (int m = 0; m < 2; m++)
180                                 {
181                                     if ((id_proc = fork()) == 0)
182                                     {
183                                         printf("Soy un proceso hijoS\n");
184                                         exit(0);
185                                     }
186                                 }
187                             }
188                             else if (l == 1)
189                             {
190                                 for (int m = 0; m < 2; m++)
191                                 {
192                                     if ((id_proc = fork()) == 0)
193                                     {
194                                         printf("Soy un proceso hijoT\n");
195                                         if (m == 0)

```



```
196         exit(0);
197     else if (m == 1)
198     {
199         for (int n = 0; n < 2; n++)
200         {
201             if ((id_proc = fork()) == 0)
202             {
203                 printf("Soy un proceso hijoU\n");
204                 exit(0);
205             }
206         }
207     }
208     exit(0);
209 }
210 }
211 }
212 exit(0);
213 }
214 }
215 }
216 else if (k == 1)
217 {
218     for (int l = 0; l < 3; l++)
219     {
220         if ((id_proc = fork()) == 0)
221         {
222             printf("Soy un proceso hijoV\n");
223             if (l == 0)
224             {
225                 exit(0);
226             }
227             else if (l == 1)
228             {
229                 for (int m = 0; m < 3; m++)
230                 {
231                     if ((id_proc = fork()) == 0)
232                     {
233                         printf("Soy un proceso hijoW\n");
234                         if (m == 0)
235                         {
236                             exit(0);
237                         }
238                         else if (m == 1)
239                         {
240                             for (int n = 0; n < 3; n++)
241                             {
242                                 if ((id_proc = fork()) == 0)
243                                 {
244                                     printf("Soy un proceso hijoX\n");
245                                     exit(0);
246                                 }
247                             }
248                         }
249                         else if (m == 2)
250                         {
251                             exit(0);
252                         }
253                     }
254                 }
255             }
256         }
257     }
258     else if (l == 2)
259     {
260         for (int m = 0; m < 2; m++)
261         {
262             if ((id_proc = fork()) == 0)
263             {
264                 printf("Soy un proceso hijoY\n");
265                 if (m == 0)
```

```
265         {
266             exit(0);
267         }
268         else if (m == 1)
269         {
270             for (int n = 0; n < 2; n++)
271             {
272                 if ((id_proc = fork()) == 0)
273                 {
274                     printf("Soy un proceso hijoZ\n");
275                     exit(0);
276                 }
277             }
278             exit(0);
279         }
280     }
281 }
282 }
283 exit(0);
284 }
285 }
286 }
287 exit(0);
288 }
289 }
290 exit(0);
291 }
292 }
293 exit(0);
294 }
295 }
296 }
297 }
298 else
299     exit(0);
300 }
301 exit(0);
302 }
```