```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

```
princ_molt (...){
  if(pos==n){
    for(i=0; i<n; i++) printf("%d", sol[i]);
  printf("\n");
  return cnt+1;
  }
  for(i=0; i< VAL[pos].n_scelte; i++){
    sol[pos]: val[pos].scelte[i];
    princ_molt(pos+1, ...);
  }
  return cnt;
}
```

input

n=3 pos=0 cnt=0

VAL

pos→0 | scelte: 0 1 | n_scelte: 2
1 | scelte: 0 1 2 | n_scelte: 3
2 | scelte: 0 1 | n_scelte: 2

```
typedef struct {
  int *scelte;
  int n_scelte;
} Livello;
```

1

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

SOL

Call Stack: princ_molt 1

2

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

pos=0
cnt=0
i=0

SOL: A0

Call Stack: princ_molt 1

3

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

pos=1
cnt=0

Call Stack: princ_molt 2, princ_molt 1

4

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

pos=1
i=0
cnt=0

SOL: A0, A0 P0

Call Stack: princ_molt 2, princ_molt 1

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

pos=2
cnt=0

Call Stack: princ_molt 3, princ_molt 2, princ_molt 1

6

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

pos=2
i=0
cnt=0

SOL: A0, A0 P0, A0 P0 S0

Call Stack: princ_molt 3, princ_molt 2, princ_molt 1

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

```
princ_molt (...){
  if(pos==n){
      for(i=0; i<n; i++) printf("%d", sol[i]);
  printf("\n");
  return cnt+1;
  }
  for(i=0; i<VAL[pos].n_scelte; i++){
      sol[pos]: val[pos].scelte[i];
      princ_molt(pos+1, ...);
  }
  return cnt;
}
```

input

n=3 pos=0 cnt=0

VAL

pos→0 scelte [0 1] n_scelte 2

1 scelte [0 1 2] n_scelte 3

2 scelte [0 1] n_scelte 2

```
typedef struct {
    int *scelte;
    int n_scelte;
} Livello;
```

7

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Pos=3
Cnt=0

Call Stack: princ_molt 1 | princ_molt 2 | princ_molt 3

8

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Pos 3
Cnt = 0

Call Stack: princ_molt 1 | princ_molt 2 | princ_molt 3

9

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Pos=2

Call Stack: princ_molt 1 | princ_molt 2 | princ_molt 3

10

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Pos=2
cnt=1
i=1

Call Stack: princ_molt 1 | princ_molt 2 | princ_molt 3

SOL

A0 | A0 P0 | A0 P0 S0 | A0 P0 S1

14

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Pos=3
cnt=1

Call Stack: princ_molt 1 | princ_molt 2 | princ_molt 3

12

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Call Stack: princ_molt 1 | princ_molt 2 | princ_molt 3

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

```
princ_molt (...){
  if(pos==n){
      for(i=0; i<n; i++) printf("%d", sol[i]);
  printf("\n");
  return cnt+1;
  }
  for(i=0; i< VAL[pos].n_scelte; i++){
      sol[pos]: val[pos].scelte[i];
      princ_molt(pos+1, ...);
  }
  return cnt;
}
```

input

n=3 pos=0 cnt=0

VAL

pos→0: scelte [0 1], n_scelte 2

1: scelte [0 1 2], n_scelte 3

2: scelte [0 1], n_scelte 2

```
typedef struct {
  int *scelte;
  int n_scelte;
} Livello;
```

13

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Pos=2
Cnt=2

Call Stack: princ_molt 3 / princ_molt 2 / princ_molt 1

14

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

i=2

i=2
Pos=2
Cnt=2

Call Stack: princ_molt 3 / princ_molt 2 / princ_molt 1

15

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

→ tolgo dallo stack princ_molt 3

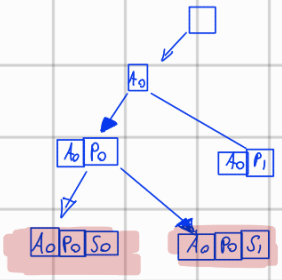
Call Stack: princ_molt 2 / princ_molt 1

16

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Pos=1
i=0
Cnt=2

Call Stack: princ_molt 2 / princ_molt 1



17

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Pos=2
Cnt=2

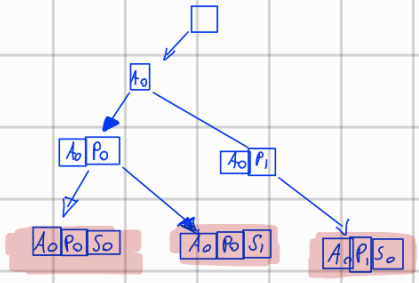
Call Stack: princ_molt 3 / princ_molt 2 / princ_molt 1

18

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Pos=2
i=0
Cnt=2

Call Stack: princ_molt 3 / princ_molt 2 / princ_molt 1



```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

```
princ_molt (...){
  if(pos==n){
      for(i=0; i<n; i++) printf("%d", sol[i]);
  printf("\n");
  return cnt+1;
  }
  for(i=0; i< VAL[pos].n_scelte; i++){
      sol[pos]: val[pos].scelte[i];
      princ_molt(pos+1, ...);
  }
  return cnt;
}
```

input

n=3 pos=0 cnt=0

VAL

pos→0 | scelte: 0 1 | n_scelte: 2
1 | scelte: 0 1 2 | n_scelte: 3
2 | scelte: 0 1 | n_scelte: 2

```
typedef struct {
  int *scelte;
  int n_scelte;
} Livello;
```

(19)

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

pos 3

Call Stack: princ_molt 4 | princ_molt 3 | princ_molt 2 | princ_molt 1

(20)

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;   // cancella ultima chiamata nello stack
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Call Stack: princ_molt 3 | princ_molt 2 | princ_molt 1

(21)

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

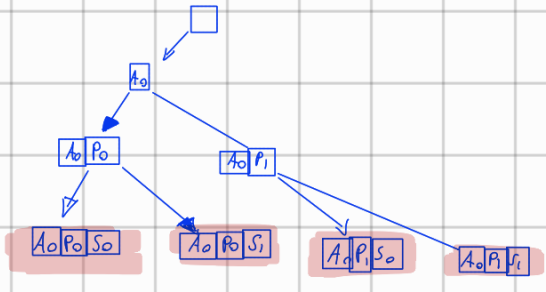
Call Stack: princ_molt 3 | princ_molt 2 | princ_molt 1

(22)

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {   // i=1
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

pos = 2
i = 1
cnt = 3

Call Stack: princ_molt 3 | princ_molt 2 | princ_molt 1



(23)

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

pos = 3

Call Stack: princ_molt 4 | princ_molt 3 | princ_molt 2 | princ_molt 1

(24)

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;   // cancella ultima chiamata
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

Call Stack: princ_molt 3 | princ_molt 2 | princ_molt 1

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

```
princ_molt (...){
  if(pos==n){
      for(i=0; i<n; i++) printf("%d ", sol[i]);
  printf("\n");
  return cnt+1;
  }
  for(i=0; i< VAL[pos].n_scelte; i++){
      sol[pos]: val[pos].scelte[i];
      princ_molt(pos+1, ... );
  }
  return cnt;
}
```

input

n=3 pos=0 cnt=0

VAL

pos→0

1

2

```
typedef struct {
  int *scelte;
  int n_scelte;
} Livello;
```

25

Call Stack: princ_molt 3 / princ_molt 2 / princ_molt 1

26

pos=2
i=2
cnt=4

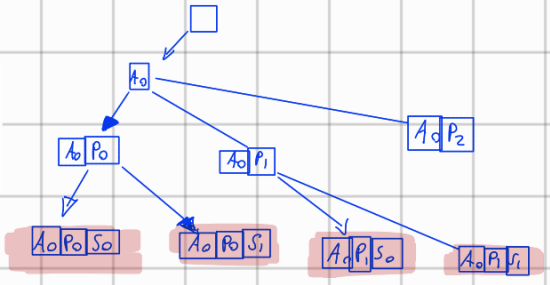
Call Stack: ~~princ_molt 3~~ / princ_molt 2 / princ_molt 1

return cnt; → cancella princ_molt 3

27

pos=1
i=2 (n_scelte=3)
cnt=4

Call Stack: princ_molt 2 / princ_molt 1



28

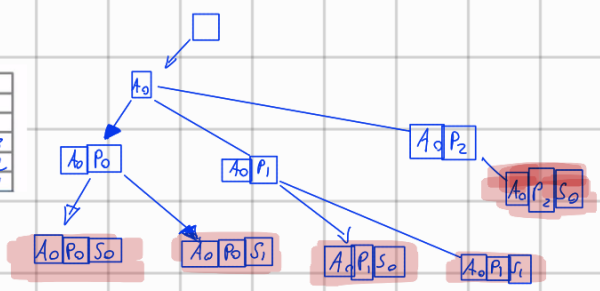
pos=2
cnt=4

Call Stack: princ_molt 3 / princ_molt 2 / princ_molt 1

29

pos=2
cnt=4
i=0

Call Stack: princ_molt 3 / princ_molt 2 / princ_molt 1



30

pos=3
cnt=4

Call Stack: princ_molt 4 / princ_molt 3 / princ_molt 2 / princ_molt 1

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

```
princ_molt (...){
  if(pos==n){
      for(i=0; i<n; i++) printf("%d", sol[i]);
  printf("\n");
  return cnt+1;
  }
  for(i=0; i< VAL[pos].n_scelte; i++){
      sol[pos]: val[pos].scelte[i];
      princ_molt(pos+1, ...);
  }
  return cnt;
```

input

n=3 pos=0 cnt=0

VAL

pos→0: scelte 0 1 | n_scelte 2

1: scelte 0 1 2 | n_scelte 3

2: scelte 0 1 | n_scelte 2

```
typedef struct {
  int *scelte;
  int n_scelte;
} Livello;
```

31

pos 3

cnt 5

Call Stack: princ_molt 4 (cancellata), princ_molt 3, princ_molt 2, princ_molt 1

32

Call Stack: princ_molt 3, princ_molt 2, princ_molt 1

33

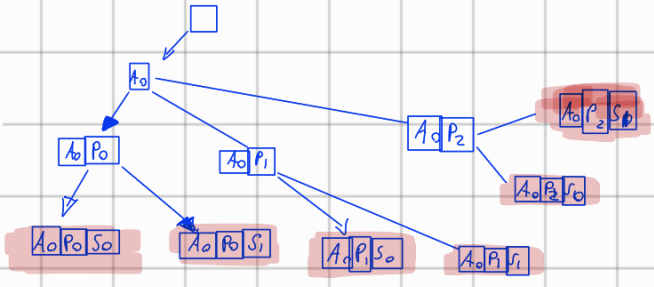
i = 1

cnt = 5

pos = 2

=3)

Call Stack: princ_molt 3, princ_molt 2, princ_molt 1



34

Pos 3

cnt = 5

Call Stack: princ_molt 4, princ_molt 3, princ_molt 2, princ_molt 1

35

Cnt 6

Pos 3

Call Stack: princ_molt 4 (cancellata), princ_molt 3, princ_molt 2, princ_molt 1

36

pos = 2

cnt 6

Call Stack: princ_molt 3, princ_molt 2, princ_molt 1

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

```
princ_molt (...){
  if(pos==n){
      for(i=0; i<n; i++) printf("%d ", sol[i]);
  printf("\n");
  return cnt +1;
  }
  for(i=0; i< VAL[pos].n_scelte; i++){
      sol[pos]: val[pos].scelte[i];
      princ_molt(pos+1, ... );
  }
  return cnt;
```

input

n=3 pos=0 cnt=0

VAL

pos→0: scelte 0 1 | n_scelte 2
1: scelte 0 1 2 | n_scelte 3
2: scelte 0 1 | n_scelte 2

```
typedef struct {
    int *scelte;
    int n_scelte;
} Livello;
```

37

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

i=2 esco dal for

chiamo princ_molt 3

Pos 2
Cnt 6

Call Stack: princ_molt 3 (crossed out), princ_molt 2, princ_molt 1

vedi n. 26

38

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

i=3 esco dal for

chiamo princ_molt 2

Pos 1
Cnt 6

Call Stack: princ_molt 2 (crossed out), princ_molt 1

vedi n. 2

39

```c
int princ_molt(int pos, Livello *val, int *sol,int n, int cnt) {
  int i;
  if (pos >= n) {
    for (i = 0; i < n; i++)
      printf("%d ", sol[i]);
    printf("\n");
    return cnt+1;
  }
  for (i = 0; i < val[pos].num_scelte; i++) {
    sol[pos] = val[pos].scelte[i];
    cnt = princ_molt(pos+1, val, sol, n, cnt);
  }
  return cnt;
}
```

i=1 → continuo con

Pos 0
Cnt 6

Call Stack: princ_molt 0