

```
int main () {  
    int N, A, S, X, D = 0, B = 0, i = 0  
    scanf ("%d %d %d", &N, &A, &S);  
    do {  
        scanf ("%d", &X);  
        if (X < A) {  
            B++;  
        } else { D++; }  
        i++; printf ("%d %d %d", i, B, D);  
    } while (B != S || i != N)  
    printf ("%d %d %d", i, B, D);  
    if (B == S) {  
        printf ("session cancelled");  
    }  
    else { printf ("session valid"); }  
    return 0; }
```

## Copy 22

---

```
int main ( )
{
    int N, A, S, X, D = 0, B = 0, i = 0 scanf ( " %d %d %d ", & N, & A, & S ) ;
    do
    {
        scanf ( " %d ", & X ) ;
        if ( X < A )
        {
            B ++ ;
        }

        else
        {
            D ++ ;
        }

        i ++ ;
        printf ( " %d %d %d ", i, B, D ) ;
    }

    while ( B != S || i != N ) printf ( " %d %d %d ", i, B, D ) ;
    if ( B == S )
    {
        printf ( " session cancelled " ) ;
    }

    else
    {
        printf ( " session valid " ) ;
    }

    return 0 ;
}
```

### Analyse :

#### Algorithmique :

- Boucle `do..while (B != S || i != N)`. Condition `OU`. Tant que l'un est vrai (toujours vrai tant que `i!=N`). Il faut `ET (&&)` pour continuer, ou condition d'arrêt `(B == S || i == N)`. Ici `while` condition de continuation. Tant que `(B!=S ET i!=N)`. Le `OU` rend la boucle dangereuse (si `B` atteint `S` mais `i!=N`, continue ?).
- Syntaxe `B => S` (inversé `>=`).

**NOTE FINALE : 10 / 20**

### Feedback :

- **Appréciation globale : Moyen.** Erreur logique booléenne condition `while`.
-