Nel progetto Disparities ci sono 3 attività:

- StartActivity, nella quale si inserisce il valore massimo di disparità

-MainActivity, nella quale si svolge un ipotetico test visualizzando il variare dei valori di disparità

-ResultsActivity, nella quale si mostrano i risultati

L’algoritmo principale è presente in MainActivity e presenta una serie di metodi che vanno a modificare e a mostrare nelle varie TextView i vari valori di disparità calcolati.

N.B.: sono state prese delle TextView perché serviva inserire degli interi, non immagini.

Il metodo “test” è quello che viene chiamato ogni volta che si hanno dei nuovi valori di disparità da calcolare: al suo interno vi sono tutti i metodi che calcolano e mostrano i 6 valori di disparità (3 uguali a 0 e 3 diversi da 0).

Il metodo “updateDisparities” calcola i 3 valori di disparità che andranno usati nel test.

Il metodo “createDispForTest” unisce due vettori di dimensione 3 (uno contenente solo zeri e l’altro contenente le 3 disparità) in un unico vettore: sarà il vettore con il quale si andrà a lavorare.

N.B.: questo metodo fa si che la posizione dei 3 valori di disparità sia totalmente casuale: non in fila (es: o-1-2), non in posizione solo pari (0-2-4) o dispari (1-3-5),…

Il metodo “updateTextView” serve per modificare i 6 valori presenti nelle textView quando si passa da un test al successivo.

Il metodo “updateRequested” serve per incrementare il numero di volte che una disparità viene chiesta. Servirà per il risultato finale.

Il metodo “showSeenRequested” è di debug e serve per mostrare a video i valori presenti nei vettori requested e seen.

-Requested è un vettore contenente, per ogni valore di disparità (da 0 a MAXDISPARITY), il numero di volte che una disparità è stata chiesta (0 viene incrementato di 3 ogni volta che si genera un nuovo vettore di disparità).

-Seen è un vettore contenente, per ogni valore di disparità (da 0 a MAXDISPARITY), il numero di volte che una disparità è stata vista (cliccando sulla checkbox).

Il metodo “evaluateChecked” serve per aggiornare il vettore seen quando una disparità viene vista.

Il metodo “countSimils” serve per contare quanti 0 o quante disparità sono stati cliccati. Serve per valutare la correttezza nello svolgimento del test (non vengono cliccati troppi 0).

Il metodo “lastDisp” ritorna l’ultima disparità diversa da 0 vista. Questo metodo non tiene conto della correttezza del test: ad esempio se come valori di disparità ci sono 10-6-1 e vengono cliccati solo 10-1, il metodo ritornerà 1.

Il metodo “checkSequence” è il primo metodo usato per verificare la correttezza del test. Si prende il valore calcolato con il metodo “lastDisp”: se facendo scorrere il vettore di disparità si trova che un valore di disparità più grande di quello calcolato dal metodo non è stato cliccato si ritorna false, altrimenti true.

N.B.: questo sarà il primo controllo che verrà fatto quando verrà cliccato il bottone SUBMIT.

Il metodo “getMaxDisparity” ritorna il valore di disparità visto più piccolo.

Es: 0,10,6,0,0,1 e vengono visti 10,6, il metodo ritorna 6.

Il metodo “getMinDisparity” ritorna il valore di disparità non visto più grande.

Es: 0,10,6,0,0,1 e vengono visti 10,6, il metodo ritorna 1.

Il metodo “getDisparities” serve unicamente per prelevare i 3 valori di disparità dal vettore contenente le 3 disparità (una ripetizione).

Il metodo “countSameDisparities” ritorna un intero che indica quanti valori di disparità uguali ci sono nel vettore contenente le 3 disparità.

N.B.: questo servirà quando si arriverà al punto del test in cui ci saranno 2 o 3 immagini che avranno lo stesso valore di disparità e bisognerà verificare che vengano cliccate tutte.

Il metodo “limitsOneStep” serve per distinguere una prima fase del test, in cui ci sono 3 valori di disparità distinti, da una seconda fase, nella quale nel vettore di 3 disparità ci sono 2 o 3 disparità uguali.

Il metodo “checkSameDisparities” prende in input un valore di disparità e valuta se tutte le TextView contenente quel valore sono state cliccate: ritorna true in caso affermativo, false altrimenti.

N.B.: questo metodo verrà usato solo quando si risconteranno 2 o 3 valori uguali di disparità.

Il metodo “goToResults” fa passare all’attività successiva.

Il metodo “onClickSubmit” serve per valutare la correttezza del test e per decidere come continuare (o con un nuovo test, o andando ai risultati).

Per prima cosa valuta il metodo checkSequence:

-Se nel vettore 10,0,6,1,0,0 vengono visti 10,6, allora si prosegue con l’algoritmo

-Se nello stesso vettore si vedono 6,1, oppure 10,1, oppure 6, oppure 1 allora il metodo ritorna false e si propone un nuovo test con nuovo\_massimo = MAXDISPARITY e nuovo\_minimo = valore a metà del vettore di 3 disparità

Se checkSequence ritorna true, bisogna distinguere 2 casi:

-firstPhase, una fase in cui i 3 valori di disparità sono tutti diversi

-una seconda fase in cui si hanno 2 o 3 valori di disparità uguali.

Se entriamo nella sezione firstPhase si possono distinguere 3 casi:

-caso di test fatto correttamente (si vedono uno o nessuno 0 e almeno una disparità):

->nuovo test con nuovo\_Massimo = getMaxDisparity e nuovo\_minimo = getMinDisparity

Esempio: 10,0,0,6,1,0 e si vedono 10,6, oppure 10,0,6, allora il test è corretto e il nuovo test proposto avrà nuovo\_massimo = 6 e nuovo\_minimo = 1

-caso di nessuna disparità vista (indipendentemente dal numero di 0 visti):

-> è come se si fosse fatto uno skip (metodo “onClickSkip”)

-altrimenti il test è fatto scorrettamente (caso in cui vengano visti più di uno 0)

->nuovo test con nuovo\_massimo = MAXDISPARITY e nuovo\_minimo = valore a metà del vettore di 3 disparità

Esempio: 10,0,6,1,0,0 e si cliccano 10,0,0 -> nuovo test con 10 e 6

Se entriamo nella seconda fase bisogna distinguere se nel vettore contenente le 3 disparità ci sono 2 o 3 valori uguali:

-Se ci sono 2 valori uguali vuol dire che questi 2 valori saranno compresi tra MAXDISPARITY e 2 (inclusi), mentre il terzo valore sarà più piccolo di un’unità rispetto ai due valori precedenti (es: 8,8,7, oppure 2,2,1, oppure se MAXDISPARITY=10 si avrà 10,10,9, …).

-Se ci sono 3 valori uguali questi saranno tutti 1

Se siamo nel caso di 2 disparità uguali: per prima cosa si verifica che entrambe le TextView contenenti lo stesso valore di disparità siano state cliccate. Poi si valutano 3 possibili casi:

-Se il test è fatto correttamente (entrambe e sole le 2 disparità cliccate e nessuno o uno zero viene cliccato), allora si va al risultato.

->Esempio: 7,7,0,6,0,0 e si vedono 7,7, oppure 7,7,0 si va al risultato

-Se il test è fatto correttamente (si vedono tutte e 3 le disparità e nessuno o uno vero viene cliccato), allora si propone un nuovo test con nuovo\_massimo = il minimo valore di disparità e nuovo\_minimo = 1.

->Esempio: 7,7,0,6,0,0 e si vedono 7,7,6, oppure 7,7,6,0, allora nuovo test con massimo=6 e minimo = 1

-altrimenti (più di uno zero o non vengono cliccate entrambe le immagini con la stessa disparità) si propone un nuovo test con nuovo\_massimo = MAXDISPARITY e nuovo\_minimo = valore a metà del vettore di 3 disparità (che coincide con il valore massimo di quel vettore)

->Esempio 7,7,0,6,0,0 e vengono viste 7, oppure 7,7,0,0, oppure 7,6,… allora si propone un nuovo test con nuovo\_massimo = MAXDISPARITY e nuovo\_minimo = 7.

Se siamo nel caso di 3 disparità uguali: per prima cosa si verifica che tutte e 3 le TextView contenenti lo stesso valore di disparità siano state cliccate.

Questa sezione è l’unica che richiede che il test venga effettuato più volte di seguito con gli stessi valori di disparità (tre 0 e tre 1). Prendo come vettore di riferimento per questa sezione 1,0,0,1,1,0.

Vengono forniti 3 tentativi e viene richiesto che 2 di questi 3 tentativi siano svolti correttamente per verificare il valore 1 di disparità.

-Se non vengono viste tutte e tre le disparità ad 1 oppure vengono selezionate troppe immagini vuote si distinguono 2 casi:

-Se ho ancora tentativi si diminuisce di 1 il numero di tentativi e si propone un nuovo test con tre 0 e tre 1

-Se non ho più tentativi riparto da capo con il test con nuovo\_massimo = MAXDISARITY e nuovo\_minimo = 1

-Altrimenti vuol dire che sono state viste tutte e tre le disparità ad 1 ed il numero di 0 visti è 0 o 1 e si valuta quanti di questi test sono stati fatti correttamente

-Se sono stati fatti 2 test corretti su 3 allora si va al risultato

-Se il numero di test corretti è 0 o 1 allora si propone un nuovo test con tre 0 e tre 1, ma con un tentativo in meno ed un test corretto in più.

->Questa è una sorta di verifica ripetuta per il valore di disparità 1: è stata fatta perché il valore di disparità 1 viene calcolato dall’algoritmo in 2 soli casi:

-Nel vettore iniziale, in cui DISPARITA\_MINIMA = 1

-Quando si esegue il test e si vedono tutti i valori di disparità. Esempi:

-se dal vettore iniziale 10,6,0,0,0,1 si vedono 10,6,1,0 oppure 10,6,1 si passa subito alla sezione di tre 0 e tre 1

-se si arriva ad un certo punto del test in cui si ha questo vettore 5,5,4,0,0,0, e si vedono 5,5,4 oppure 5,5,4,0 allora l’algoritmo propone un nuovo test con nuovo\_massimo=4 e nuovo\_minimo=1. Da questo punto si può generare ad esempio un vettore di questo tipo: 4,0,0,3,0,1 e se si vedono 4,3,1,0 oppure 4,3,1 allora l’algoritmo propone un vettore composto da tre 0 e tre 1

Tutti gli altri valori di disparità compresi tra MAXDISPARITY e 2 inclusi, invece, possono essere calcolati dall’algoritmo in qualsiasi momento in quanto il test sarebbe ipoteticamente infinito

Il metodo “onClickSkip” viene considerato come se il test è stato fatto scorrettamente. Le opzioni sono 2:

-Se il vettore contenente le 3 disparità contiene MAXDISPARITY, MAXDISPARITY, MAXDISPARITY-1 e si clicca SKIP, allora si va al risultato. Questo vuol dire che non è stata vista nessuna disparità (a meno di click fortunati).

-Se invece siamo in qualsiasi altro punto del test, si propone un nuovo test con nuovi massimo e minimo: nuovo\_massimo = MAXDISPARITY e nuovo\_minimo = valore a metà del vettore di 3 disparità