Whitebox Testing Scenario

Dokumen ini berisi proses pengujian *whitebox* terhadap modul pengacakan tantangan dan nama target. Pemilihan modul ini ditentukan berdasarkan fungsi utama dari aplikasi yang bertujuan untuk menentukan tantangan secara acak. Pada pengujian ini algoritma disajikan dalam bentuk pseudocode dan dibentuk *branch* untuk dilakukan proses *basis path testing*. Masukan utama dari algoritma ini adalah pilihan pemain terhadap tipe tantangan, "Jujur" atau "Berani". Keluaran dari algoritma ini adalah menampilkan tantangan dan objek dari tantangan. Tidak akan ada masukan bernilai "null" karena masukan bergantung pada aksi penekanan tombol pada aktivitas sebelumnya. Keluaran juga tidak akan bernilai "null" karena tantangan sudah diinisalisasi dalam bentuk array begitu juga dengan target.

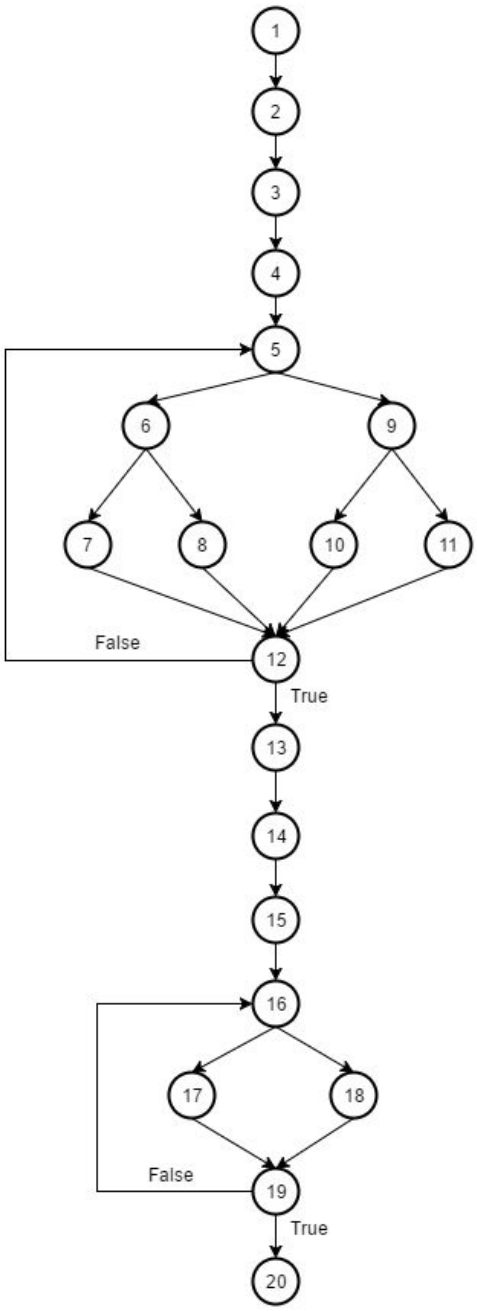
1. Pseudocode

```
i     <- 0
task  <- getTask
truth <- [array pertanyaan jujur]
dare  <- [array tantangan berani]
while (i < dare.length) {
    switch (task) {
        case "truth":
            text1 <- truth[i]
            if (i == truth.length - 1) {
                i <- 0
            } else {
                i++
            }
        case "dare" :
            text1 <- dare[i]
            if (i == dare.length - 1) {
                i <- 0
            } else {
                i++
            }
    }
    if interrupted (button clicked), break from loop.
}
print text1.

j      <- 0
player <- [array pemain sisa]
while (j < player.length) {
    text2 <- player[j]
    if (j == player.length - 1) {
        j <- 0
    } else {
        j++
    }
    if interrupted (button clicked), break from loop.
}
print text2.
```

2. *Branch*

Berikut adalah branch yang dibuat berdasarkan pseudocode pada poin sebelumnya. Dari branch ini kemudian ditentukan kemungkinan jalur yang dapat dilewati.

Code	Branch
i<-0 task<-getTask truth<-[array pertanyaan jujur] dare<-[array tantangan berani] while switch (task) case "truth" text1<-truth[i] if i<-0 else i++ case "dare" text1<-dare[i] if i<-0 else i++ interrupted print text1 j<-0 target<-[array pemain sisa] while text2<-target[j] if j<-0 else j++ interrutped print text2	

3. *Cyclomatic Complexity and Path Testing*

$$V(G) = E-N+2(P)$$

$$V(G) = 25-20+2(1)$$

$$V(G) = 7$$

Berdasarkan perhitungan *cyclomatic complexity* dengan jumlah *edge* = 25 dan *node* = 20, terdapat 7 jalur minimum semua *nodes* terlewati. 7 jalur ini menunjukkan bahwa dengan melakukan pengujian terhadap minimum 7 jalur, program akan menampilkan hasil yang sesuai.

Terdapat dua skenario yang akan diujikan.

- Pemain memilih JUJUR

  Jalur:

  1. 1-2-3-4-5-6-7-12-13-14-15-16-17-19-20
  2. 1-2-3-4-5-6-7-12-13-14-15-16-18-19-20
  3. 1-2-3-4-5-6-8-12-13-14-15-16-17-19-20
  4. 1-2-3-4-5-6-8-12-13-14-15-16-18-19-20

- Pemain memilih BERANI

  Jalur:

  1. 1-2-3-4-5-9-10-12-13-14-15-16-17-19-20
  2. 1-2-3-4-5-9-10-12-13-14-15-16-18-19-20
  3. 1-2-3-4-5-9-11-12-13-14-15-16-17-19-20
  4. 1-2-3-4-5-9-11-12-13-14-15-16-18-19-20

Pada pengujian *branch*, terdapat 8 jalur yang dapat dilewati untuk dapat menampilkan hasil. 8 jalur ini menunjukkan bahwa agar dapat dipastikan tidak terjadi *error* seluruh jalur tersebut harus dilakukan pengujian.