BAB II

LANDASAN TEORI

2.1 Kepribadian

Kepribadian menurut Allport (Barrick & Ryan, 2003) didefinisikan sebagai suatu organisasi yang dinamik dalam diri individu yang menghubungkan antara individu dengan lingkungannya dan hal tersebut menentukan penyesuaian diri individu secara unik terhadap lingkungan. Definisi ini menekankan pada atribut eksternal seperti peran individu dalam lingkungan sosial, penampilan individu, dan reaksi individu terhadap orang lain. Feist dan Feist mendefinisikan kepribadian sebagai sebuah pola yang relatif menetap, disposisi atau karakteristik didalam individu yang memberikan beberapa ukuran yang konsisten tentang perilaku (Feist & Feist, 1998).

Menurut Larsen & Buss kepribadian merupakan sekumpulan sifat psikologis dan mekanisme di dalam individu yang diorganisasikan, relatif bertahan yang mempengaruhi interaksi dan adaptasi individu di dalam lingkungan, meliputi lingkungan fisik dan lingkungan sosial (Larsen, Buss, & David, 2002). Dari berbagai definisi tersebut dapat disimpulkan bahwa kepribadian adalah sebuah karakteristik di dalam diri individu yang relatif menetap, bertahan, yang mempengaruhi penyesuaian diri individu terhadap lingkungan (Mastuti, 2005).

2.2 Tes Kepribadian Dengan Metode DISC

Saifuddin Azwar menyebutkan bahwa pengertian tes adalah suatu alat yang sudah distandarisasikan untuk mengukur salah satu sifat, kecakapan atau tingkah laku (Azwar, 1987). Oleh karena itu dapat disimpulkan bahwa tes kepribadian adalah suatu alat yang mengukur sifat dan tingkah laku manusia untuk menyesuaikan diri dengan lingkungannya. Tes kepribadian memiliki banyak metode, salah satunya metode *Dominance – Influence – Steadiness – Compliance* atau biasa disebut DISC, dalam bahasa Indonesia dikenal dengan Dominan – Intim – Stabil – Cermat.

Istilah DISC pertama kali ditemukan dan diperkenalkan oleh William Moulton Marston (1893 - 1947). DISC juga berhubungan erat dengan teori Dr. Albert Mehrabian dari UCLA yang menyatakan bahwa komunikasi adalah komposisi dari 55% gerakan tubuh, 38% nada suara, 7% perkataan.

Secara garis besar sifat-sifat keempat tipe DISC adalah sebagai berikut (Shin, 2013) :

a. Tipe D : tipe yang ‘menguasai’, baik dalam percakapan, tugas, pengambilan keputusan dan segalanya. Biasanya berkemauan keras dan tahu apa yang ingin dicapai. Berani mengambil resiko dan melanggar peraturan yang telah ditetapkan. Tidak takut dengan konflik, malah menjadi semakin bersemangat ketika ada konflik.
b. Tipe I : tipe yang secara naturalnya senang berbicara, mempengaruhi orang lain, dan cenderung membesar-besarkan supaya orang lain terpengaruh. Tipe yang mudah mengekspresikan perasaannya, bisa tertawa, bisa juga

mengangis. Kadang adalah penghibur dan pendengar yang baik. Tidak suka yang terstruktur, inginnya mengalir dan spontan.

c. Tipe S : tipe yang secara natural tidak suka mengatur orang lain dan tidak suka memaksakan kehendaknya, menjauhi konflik dan bahkan cenderung mengikuti kehendak orang lain. Tipe yang mudah diprediksi, suka melakukan sesuatu secara berulang-ulang/rutin dan tidak suka mengubah tatanan yang ada. Banyak berpikir sebelum berbicara namun seringnya akhirnya tidak mengungkapkan isi hatinya dan berharap orang lain akan mengerti apa yang dia inginkan/pikirkan.
d. Tipe C : tipe yang tidak suka menonjolkan diri di antara orang banyak, tetapi fokus pada pekerjaan. Sangat mengikuti sistem dan peraturan yang berlaku. Biasanya memiliki memori dan daya ingat yang canggih, sehingga bisa mengingat banyak hal walaupun kecil. Tipe yang sangat disiplin dan perfeksionis, sangat menjaga barang-barang atau milik mereka. Tipe yang penuh kekhawatiran, penuh pertimbangan dan biasanya suka mempersiapkan sesuatu jauh-jauh hari sebelum kejadian.

Cara menggunakan tes kepribadian dengan metode DISC adalah sebagai berikut (Shin, 2013) :

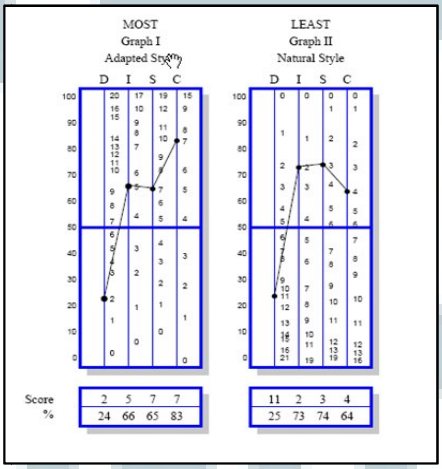
a. Peserta tes akan diberikan 4 lembar berisi soal dan kunci kode jawaban.
b. Ada 24 soal yang diberikan, dan setiap soal memiliki 4 pilihan. Peserta diharuskan memilih satu yang menurut peserta adalah sifat yang sesuai dengan peserta (*Most*), dan memilih satu yang bukan sifat peserta (*Least*), dari 4 pilihan yang ada.

	M	L	Tick only ONE most (M) and ONE least (L)
01	❒	❒	I am gentle and kind with others; not harsh or rough
	❒	❒	I can easily persuade others to my point of view
	❒	❒	I am humble – and not too proud or assertive
	❒	❒	I am creative and original – I do things in new ways
02	❒	❒	I like receiving attention from others
	❒	❒	I work well with others to achieve a goal
	❒	❒	I stand up for what I think is right
	❒	❒	I am pleasant, agreeable, and like pleasing others
03	❒	❒	I easily give in to other people's ideas
	❒	❒	I like to take risks and try new experiences
	❒	❒	I am loyal and true to my friends
	❒	❒	I am charming and like to attract attention

Gambar 2.1 Contoh Soal DISC
Sumber (http://www.staffhub.com/pages/disc/DISC%20Questionaire%20-%20STAFFHUB.pdf, 2010)

c. Setelah peserta mengisi semua soal, peserta menghitung jumlah D, I, S, dan C yang disesuaikan dengan kunci kode jawaban.

d. Peserta kemudian mencocokkan kunci kode jawaban dengan tabel yang tersedia untuk mengetahui karakter dari peserta.



Gambar 2.2 Gambar tabel DISC
Sumber (Advanced Hiring System, 2011)

2.3 Algoritma

Sejarah kata Algoritma berasal dari nama seorang ahli matematika bangsa Arab yaitu Abu Ja'far Muhammad ibnu Musa al-Khuwarizmi. Al-Khuwarizmi dibaca oleh orang Barat menjadi *Algorism*. Perubahan kata *algorism* menjadi *algorithm* karena kata *algorism* sering dikelirukan dengan *arithmetic*, sehingga akhiran *–sm* berubah menjadi *–thm*. Lambat laun kata algoritma dipakai sebagai metode perhitungan(komputasi) secara umum, sehingga kehilangan makna aslinya. Dalam bahasa Indonesia kata *algorithm* diserap menjadi algoritma (Saniman & Fathoni, 2008).

Definisi dari algoritma adalah langkah-langkah yang logis dan terstruktur untuk menyelesaikan suatu permasalahan. Yang dimaksud logis dalam hal ini adalah bahwa langkah yang diambil masuk akal dan dapat dinilai benar atau salahnya. Sedangkan terstruktur memiliki pengertian bahwa langkah tersebut memiliki susunan sesuai urutan kejadian atau waktu (Wahyudi, 2007).

Dalam memecahkan suatu permasalahan, algoritma dapat dituliskan dalam beberapa cara, yaitu sebagai berikut (Wahyudi, 2007) :

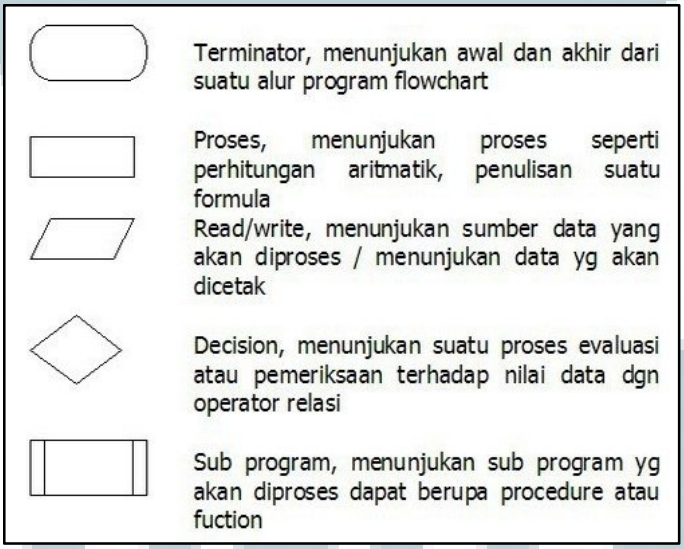
a. Menggunakan bahasa sehari-hari, seperti contoh berikut, cara memecahkan masalah memindahkan kecap dalam botol dengan sirup dalam botol adalah dengan cara pertama, pindahkan sirup ke botol lain, kemudian pindahkan kecap ke dalam botol sirup, setelah itu pindahkan sirup ke botol kecap.
b. Menggunakan *Pseudo-code*, yaitu dituliskan mendekati perintah bahasa pemrograman yang akan digunakan sebagai alat implementasi program.

```
function TARJAN(Node* node)
    node.visited ← true
    node.index ← indexCounter
    s.push(node)
    for all successor in node.successors do
        if !node.visited then TARJAN(successor)
        end if
        node.lowlink ← MIN(node.lowlink, successor.lowlink)
    end for
    if node.lowlink == node.index then
        repeat
            successor ← stack.pop()
        until successor == node
    end if
end function
```

Gambar 2.3 Contoh Pseudocode

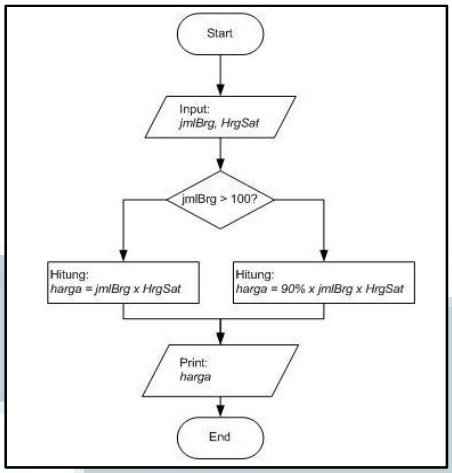
Sumber (Moose, 2012)

c. Menggunakan *Flowchart,* yaitu ditulis dengan simbol-simbol yang mewakili urutan atau uraian kejadian pemecahan masalah.



Gambar 2.4 Simbol flowchart

Sumber (Nuraini, 2013)



Gambar 2.5 Contoh Flowchart

Sumber (Nuraini, 2013)

2.4 Pencarian(Searching)

Pencarian(*Searching*) merupakan suatu pekerjaan yang sering dikerjakan dalam kehidupan sehari-hari. Tempat pencarian data dengan cara menelusuri data-data tersebut. Tempat pencarian data dapat berupa *array* dalam *memory* (pencarian *internal*), maupun pada *file external storage* (pencarian *external*). Ada kalanya pencarian dilakukan dengan tujuan hanya untuk mengetahui apakah data tersebut ada dalam sekumpulan data atau tidak, atau mungkin di lain waktu posisi dari data yang dicari tersebut dibutuhkan untuk keperluan tertentu, atau jika kemunculan data lebih dari satu kali maka semua posisi dan frekuensi kemunculannya ingin ditampilkan (Ningtyas, 2013).

Pencarian paling sederhana dapat digambarkan sebagai berikut, misalkan suatu barisan data A[1] … A[n], maka yang menjadi masalah adalah apakah X (sembarang data dengan tipe data sama dengan tipe data yang ada dalam barisan A dan biasanya di-*input* atau sudah diketahui terlebih dahulu) ada di antara A[1]

… A[n] atau apakah X ada di dalam atau tidak dengan hasil *Boolean* benar / salah atau sukses / gagal (Ningtyas, 2013).

Dalam ilmu komputer terdapat bermacam – macam algoritma untuk metode pencarian (*searching*). Secara garis besar, metode pencarian data dapat dibagi menjadi 2 bagian yaitu (Manurung, 2013) :

a. Metode pencarian data tanpa penempatan data, seperti
    i. Metode pencarian linier (*Linear* / *Sequential Search*).
    ii. Metode pencarian Biner (*Binary Search*).
    iii. Metode pencarian Interpolasi (*Interpolation Search*).
b. Metode pencarian data dengan penempatan data, seperti
    i. Metode pencarian Langsung (*Direct Search*).
    ii. Metode pencarian Relatif (*Hash Search*).

Masing – masing algoritma pencarian memiliki syarat dan cara serta waktu pelaksanaan yang berbeda – beda. Pemilihan atas metode pencarian dilakukan berdasarkan keadaan dan keinginan pengguna metode yang biasanya tergantung pada jumlah data, jenis data dan struktur data yang digunakan (Ningtyas, 2013).

2.5 Algoritma Interpolation Search

Diantara banyaknya algoritma yang ada untuk mencari di tabel yang terurut, metode yang sangat menarik, terutama untuk mencari di sebuah *external storage*, adalah algoritma *Interpolation Search*, yang dijelaskan pertama kali oleh W. W. Peterson dalam jurnalnya yang berjudul *Addressing for random-access storage* (Peterson, 1957). Tidak seperti pecarian biner, yang meneliti entri di tengah tabel setiap saat, pencarian interpolasi meneliti dugaan lokasi dari nilai

yang dicari. Dengan demikian, pencarian interpolasi memanfaatkan pengetahuan mengenai distribusi nilai – nilai dalam tabel untuk menentukan entri yang diteliti. Contoh ada tabel yang sudah terurut yang berisikan 10 nilai yang secara acak terdiri dari 0 sampai 1.

0.08, 0.12, 0.34, 0.46, 0.47, 0.49, 0.53, 0.60, 0.65, 0.83

Nilai yang dicari adalah 0.60. Dugaan awal nilai 0.60 ada di 60% dari tabel, dengan kata lain berada di urutan ke 6. Secara matematika, dapat ditulis $\left[10 \times \frac{0.60-0}{1-0}\right] = 6$. Ini merupakan langkah interpolasi. Entri ke 6 adalah 0.49, sedangkan 0.49 < 0.60. Oleh karena itu diperlukan pencarian lebih lanjut, dengan mencari nilai tersebut di bagian tabel sisanya :

0.53, 0.60, 0.65, 0.83

Seperti cara sebelumnya dengan menggunakan interpolasi maka didapatkan proses matematika sebagai berikut $\left[4 \times \frac{0.60-0.49}{1-0.49}\right] = 1$, dan data 1 setelah 0.49 adalah 0.53, sedangkan 0.53 < 0.60. Lakukan pencarian lagi di bagian tabel sisanya :

0.60, 0.65, 0.83

Proses interpolasi dilakukan lagi sehingga mendapat persamaan $\left[3 \times \frac{0.60-0.53}{1-0.53}\right] = 1$, dan 1 data setelah 0.53 adalah 0.60, maka *index* dengan nilai 0.60 tersebut berhasil didapatkan (Marsaglia, 1993).

Dengan penjelasan di atas, maka dapat disimpulkan bahwa pencarian Interpolasi mencari data dengan cara menebak (*guess*) posisi data yang dicari dengan menggunakan rumus tertentu. Pencarian Interpolasi hanya dapat dilakukan pada barisan bilangan yang telah diurutkan baik secara menaik (*ascending*)

maupun menurun (*descending*). Posisi yang diduga sebagai posisi data yang dicari (X) tersebut dapat dihitung dengan menggunakan rumus berikut. Sumber (Suprapta, 2010)

$$posisi = \frac{kunci - k[min]}{k[max] - k[min]} \times (\max - min) + min$$

…Rumus 2.1 Interpolation Search

Keterangan :

Posisi = posisi yang diduga, pembulatan dari hasil perhitungan.

Kunci = nilai yang dicari (X)

K = array / tabel

K[min] = data dalam tabel yang minimum

K[max] = data dalam tabel yang maksimum

Min = jumlah data minimum / batas bawah dari tabel

Max = jumlah data maksimum / batas atas tabel

Apabila data pada posisi tersebut sama dengan nilai data yang dicari (X) maka pencarian dihentikan dan dinyatakan sukses. Sedangkan apabila nilai data pada posisi tersebut tidak sama dengan nilai data yang dicari (X), maka (Manurung, 2013):

a. Untuk data yang diurutkan secara menaik (*ascending*), apabila nilai data yang dicari (X) lebih kecil dibandingkan dengan nilai data pada posisi yang diduga, maka pecarian dilanjutkan dengan mengubah batas atas kawasan pencarian. Sedangkan apabila nilai data yang dicari (X) lebih besar dibandingkan dengan nilai data pada posisi yang diduga, maka pencarian dilanjutkan dengan mengubah batas bawah kawasan pencarian.

b. Untuk data yang diurutkan secara menurun (*descending*), apabila nilai data yang dicari (X) lebih besar dibandingkan dengan nilai data pada posisi yang diduga, maka pencarian dilakukan dengan mengubah batas atas kawasan pencarian. Sedangkan apabila nilai data yang dicari (X) lebih kecil dibandingkan dengan nlai data pada posisi yang diduga, maka pencarian dilanjutkan dengan mengubah batas bawah kawasan pencarian.

Proses pencarian akan dihentikan dan dinyatakan gagal apabila terjadi kondisi – kondisi berikut : nilai posisi lebih kecil dari nilai batas bawah, nilai posisi lebih besar dari nilai batas berturut – turut sama.

Algoritma pencarian dapat dibagi menjadi 2 macam, yaitu : algoritma untuk data terurut menaik (*ascending*) dan algoritma untuk data terurut menurun (*descending*). Metode pencarian Interpolasi untuk data terurut secara menaik (*ascending*) dalam bahasa pemrograman *Basic* adalah sebagai berikut (Manurung, 2013) :

```
Min = 1; Max = n; Ketemu = false; CekPos = 0;
While (Min <= Max) And Not (Ketemu)
        Temp = Min + ((X – K[min]) / (K[Max] – K[Min])) * (Max - Min)
        Posisi = INT(Temp)
        If (Temp – Posisi > 0) Then Posisi = Posisi + 1
        If (CekPos = Posisi) Then Min = Max + 1
        Else If (X = K[Posisi]) Then Ketemu = True
        Else If (X < K[Posisi]) Then Max = Posisi – 1
        Else If (X > K[Posisi]) Then Min = Posisi + 1
        Else     CekPos = Posisi    End If
```

```
Wend
If (Ketemu) Then Print X, “ditemukan”
Else Print X, “tidak ditemukan”
```

Metode pencarian Interpolasi untuk data terurut secara menurun (*descending*) dalam bahasa pemrograman *Basic* adalah sebagai berikut (Manurung, 2013) :

```
Min = 1; Max = n; Ketemu = false; CekPos = 0;
While (Min <= Max) And Not (Ketemu)
    Temp = Min + ((X – K[min]) / (K[Max] – K[Min])) * (Max - Min)
    Posisi = INT(Min + Temp)
    If (Temp – Posisi > 0) Then Posisi = Posisi + 1
    If (CekPos = Posisi) Then Min = Max + 1
    Else If (X = K[Posisi]) Then Ketemu = True
    Else If (X > K[Posisi]) Then Max = Posisi – 1
    Else If (X < K[Posisi]) Then Min = Posisi + 1
    Else    CekPos = Posisi    End If
Wend
If (Ketemu) Then Print X, “ditemukan”
Else Print X, “tidak ditemukan”
```

Fungsi dari variable *Temp* pada *source code program* Interpolasi di atas adalah sebagai tempat penyimpanan nilai asli dari perhitungan, sedangkan variable *Posisi* digunakan untuk menyimpan nilai perhitungan hasil pembulatan. Jika selisih dari *Temp* dan *Posisi* lebih besar dari nol, berarti hasil perhitungan bukan berupa bilangan bulat, maka nilai *Posisi* harus dibulatkan ke atas (dalam hal ini, nilai *Posisi* ditambah satu) (Manurung, 2013).