BAB I

PENDAHULUAN

1.1. LATAR BELAKANG

Jadwal mata kuliah merupakan hal yang sangat penting bagi kelancaran proses belajar mengajar di Program studi Matematika FMIPA UNDIP. Sering terjadinya tumbukan, baik tumbukan yang terjadi pada mata kuliah yang diambil oleh mahasiswa maupun tumbukan yang terjadi pada dosen mengakibatkan tidak efektifnya proses belajar mengajar di Program studi matematika FMIPA UNDIP. Ditambah lagi terjadinya pergantian jadwal yang mengakibatkan bertambahnya tumbukan yang terjadi pada mahasiswa. Oleh karena itu di Program studi Matematika FMIPA UNDIP di butuhkan penjadwalan mata kuliah yang baik. Untuk membuat jadwal mata kuliah yang baik kita harus memperhatikan berbagai aspek yang mempengaruhi penjadwalan mata kuliah ini. Dari aspek mahasiswa, kita perlu perhatikan ada atau tidaknya tumbukan pada mata kuliah yang diambil oleh mahasiswa, selain dilihat dari aspek mahasiswa, kita juga harus melihat dari aspek dosen, yaitu kemungkinan kemungkinan dosen akan mengampu lebih dari satu mata kuliah yang ada, sebab ada kemungkinan jumlah mata kuliah dan jumlah dosen tidak sebanding, sehingga harus dipikirkan juga solusi agar dosen tidak mengampu dua mata kuliah berbeda pada hari dan jam yang sama. Selain itu, harus dipertimbangkan juga ketersediaan kelas sehingga kegiatan belajar dapat dilaksanakan. Di samping aspek-aspek di atas, dalam penyusunan jadwal mata kuliah ini pun terdapat sangat banyak kemungkinan yang selayaknya dicoba

untuk menemukan penjadwalan yang terbaik. Karena itu dibutuhkan metode optimasi yang dapat diterapkan untuk mengerjakan penjadwalan mata kuliah ini.

Masalah optimasi dapat diselesaikan menggunakan algoritma pencarian heuristik, tapi algoritma pencarian heuristik yang biasa seperti best first search digunakan untuk kasus yang sederhana. Untuk input dan persyaratan yang lebih rumit seperti pada kasus penjadwalan mata kuliah, algoritma pencarian heuristik sudah tidak dapat lagi digunakan dengan baik untuk mendapatkan solusi yang diinginkan. Dalam kasus penjadwalan mata kuliah ini diperlukan algoritma yang lebih baik yaitu algoritma yang dapat menyelesaikan masalah multi-kriteria dan multi-objektif. Salah satu algoritma yang dapat digunakan adalah algoritma genetika. Algoritma genetika adalah suatu algoritma pencarian yang berbasis pada mekanisme seleksi alam dan genetika. Algoritma genetika merupakan salah satu algoritma yang sangat tepat digunakan dalam menyelesaikan masalah optimasi kompleks, yang sulit dilakukan oleh metode konvensional. Banyak permasalahan optimalisasi yang telah diselesaikan dengan menggunakan algoritma genetika diantaranya adalah permasalahan task assignment pada sistem terdistribusi, travelling salesman problem, timetabling, transportasi, knapsack (Desiani,2006).

Dari latar belakang yang telah disebutkan di atas, maka dalam tugas akhir ini akan dicoba mengaplikasikan algoritma genetika untuk mengoptimalkan jadwal mata kuliah. Diharapkan dengan digunakannya algoritma genetik akan diperoleh optimasi penjadwalan yaitu terjadinya kombinasi terbaik untuk pasangan mata kuliah dan dosen pengajar secara keseluruhan, tidak ada

permasalahan tumbukan jadwal pada sisi mahasiswa, serta ketersediaan ruang yang cukup dan sesuai secara fasilitas untuk seluruh mata kuliah yang ada.

1.2. PERUMUSAN MASALAH

Berdasarkan latar belakang masalah, maka yang menjadi permasalahan adalah bagaimana mengaplikasikan algoritma genetika agar dapat digunakan untuk melakukan penjadwalan mata kuliah sehingga diperoleh jadwal mata kuliah dengan kombinasi terbaik untuk pasangan mata kuliah dan dosen pengajar secara keseluruhan, tidak ada permasalahan tumbukan jadwal pada sisi mahasiswa, serta ketersediaan ruang yang cukup untuk seluruh mata kuliah yang ada.

1.3. PEMBATASAN MASALAH

Adapun pembatasan masalah dalam penulisan tugas akhir ini adalah sebagai berikut :

1. Dalam penjadwalan mata kuliah ini diasumsikan bahwa:
    a. dosen boleh mengampu beberapa mata kuliah.
    b. dosen mampu mengajar pada setiap waktu yang ditentukan oleh jadwal.
    c. mahasiswa dapat mengambil mata kuliah angkatan sebelumnya maupun sesudahnya.
    d. adanya batas hari dalam satu minggu yaitu dari hari senin sampai hari jumat.
    e. adanya batas jam kuliah dalam satu hari yaitu dari jam 07.30 sampai jam 17.30 dengan 1 jam perkuliahan sama dengan 50 menit.

f. adanya batas jumlah mahasiswa dalam satu ruang kelas.

2. Dalam program studi Matematika FMIPA UNDIP, ruang perkuliahan juga dipergunakan oleh progam studi yang lain dan jurusan yang lain oleh karena itu dalam penjadwalan mata kuliah pada program studi Matematika FMIPA UNDIP ini diasumsikan ruang yang digunakan oleh program studi Matematika tidak dipergunakan oleh program studi lain maupun jurusan lain. Selain itu dalam program studi Matematika FMIPA UNDIP juga terdapat dosen yang juga mengampu mata kuliah pada program studi lain maupun jurusan yang lain oleh karena itu dalam penjadwalan mata kuliah pada program studi Matematika FMIPA UNDIP ini juga diasumsikan dosen tidak mengampu mata kuliah pada program studi lain maupun jurusan lain.
3. Aplikasi yang dibuat pada tugas akhir ini menggunakan program Microsoft visual c++ 2005.

1.4. TUJUAN PENULISAN

Tujuan penulisan tugas akhir ini adalah membuat suatu aplikasi algoritma genetika yang dapat digunakan untuk membentuk penjadwalan mata kuliah yang efektif, yaitu terjadinya kombinasi terbaik untuk pasangan mata kuliah dan dosen pengajar secara keseluruhan, tidak ada permasalahan tumbukan jadwal pada sisi mahasiswa, serta ketersediaan ruang yang cukup untuk seluruh mata kuliah yang ada.

1.5. METODE PENELITIAN

a. Bahan dan Alat Penelitian

Bahan penelitian adalah berupa karya-karya ilmiah yang tersaji dalam jurnal, prosiding seminar, bulletin, dan buku-buku yang berkaitan dengan permasalahan yang sedang diteliti. Sedangkan alat penelitian yang digunakan adalah berupa 1 unit komputer dengan spesifikasi prosesor intel pentium IV 3 GHz, memori 2 GB yang dilengkapi dengan perangkat lunak microsoft visual c++ 2005.

b. Pelaksanaan Penelitian

Secara umum dalam membangun Aplikasi penjadwalan mata kuliah ini dilakukan melalui tahap-tahap sebagai berikut:

1. Studi literatur algoritma genetika untuk menyelesaikan permasalahan penjadwalan mata kuliah.
2. Mengidentifikasi permasalahan atau kendala yang dihadapi dalam menyusun jadwal kuliah.
4. Mengimplementasikan rancangan algoritma genetika pada modul program (analisis coding program).
5. Membangun aplikasi penjadwalan berdasarkan hasil analisis.
6. Mengevaluasi kinerja program.

1.6. MANFAAT

Adanya tugas akhir yang dibuat ini, diharapkan dapat memberikan manfaat antara lain :

1. Bagi mahasiswa :
   a. Dapat menerapkan disiplin ilmu dan memanfaatkannya.
   b. Meningkatkan pemahaman tentang penggunaan algoritma genetika.
2. Bagi pihak terkait:
   a. Dapat mengoptimalkan penyusunan jadwal mata kuliah.

1.7. SISTEMATIKA PENULISAN

Untuk mempermudah proses pembahasan dalam pembuatan laporan tugas akhir ini, maka sistematika dibuat sebagai berikut:

BAB I PENDAHULUAN

Menjelaskan secara singkat tentang latar belakang, tujuan, manfaat, pembatasan masalah serta metode penelitian dan penulisan laporan tugas akhir ini.

BAB II DASAR TEORI

Menjelaskan tentang penjadwalan mata kuliah, algoritma genetika secara umum dan bahasa pemrograman visual C++.

BAB III APLIKASI ALGORITMA GENETIKA UNTUK PENJADWALAN MATA KULIAH

Berisi tentang representasi dan penyelesaian masalah penjadwalan mata kuliah dengan algoritma genetika dan implentasi program.

BAB IV HASIL PENJADWALAN MATA KULIAH DAN ANALISA

Berisi tentang hasil simulasi program dengan menggunakan data perkuliahan di Program studi Matematika FMIPA UNDIP dan analisa hasilnya.

BAB V PENUTUP

Berisi kesimpulan dan saran-saran.

BAB II

DASAR TEORI

1.1. PENJADWALAN MATA KULIAH

Menurut Ross permasalahan jadwal mata kuliah (*lecture timetable*) adalah permasalahan pengalokasian waktu dan tempat untuk suatu kegiatan perkuliahan, seminar, dan lain-lain untuk memenuhi beberapa kendala yang berhubungan dengan kapasitas dan lokasi dari ruang, waktu, dan hal lain yang berhubungan dengan perkuliahan.

Permasalahan penjadwalan mata kuliah didefinisikan secara formal dan matematis oleh Ross sebagai berikut :

- Diberikan suatu himpunan E = $\{e_1,e_2 \ldots e_v\}$ merupakan kegiatan, T = $\{t_1,t_2 \ldots t_s\}$ merupakan waktu, P = $\{p_1,p_2 \ldots p_m\}$ merupakan tempat dan A = $\{a_1,a_2 \ldots a_n\}$ merupakan 'agents' (orang yang kehadiranya diperlukan untuk kelangsungan kegiatan, dalam hal ini sebagai contoh adalah mahasiswa dan perkuliahan sebagai 'agents').
- Diimplementasikan dalam 4 tuple (e,t,p,a) sedemikian sehingga $e \in E, t \in T, p \in P, a \in A$
- Permasalahan penjadwalan mata kuliah sekarang didefinisikan sebagai permasalahan untuk mengalokasikan sekumpulan kegiatan pada suatu tempat dan waktu yang terbatas yang meminimalkan kendala-kendala yang ada.

Dalam proses penyelesaian masalah penjadwalan mata kuliah terdapat kendala-kendala yang harus dipenuhi atau tidak boleh dilanggar. Kendala tersebut

merupakan ukuran kualitas dari penjadwalan mata kuliah, sehingga suatu jadwal mata kuliah yang optimal dapat terbentuk. Kendala-kendala yang harus dipenuhi pada penjadwalan mata kuliah dalam tugas akhir ini adalah kendala yang terjadi pada penjadwalan mata kuliah di program studi matematika FMIPA UNDIP. Kendala-kendala tersebut adalah :

1. Dosen dapat mengajar lebih dari satu mata kuliah dan tidak boleh terjadi tumbukan pada dosen .
2. Satu mata kuliah dapat diampu oleh 2 orang dosen atau lebih. Terdapat mata kuliah tertentu yang menggunakan ruang laboratorium yang harus dijadwalakan pada ruang laboratorium.
3. Mahasiswa dapat mengambil mata kuliah angkatan sebelumnya maupun sesudahnya dan tidak boleh terjadi tumbukan pada mata kuliah yang sudah diambil.
4. Tersedianya ruang yang cukup untuk seluruh mata kuliah yang ada.

1.2. ALGORITMA GENETIKA

Algoritma genetika adalah suatu algoritma pencarian yang berbasis pada mekanisme seleksi alam dan genetika. Algoritma genetika merupakan salah satu algoritma yang sangat tepat digunakan dalam menyelesaikan masalah optimasi kompleks, yang sulit dilakukan oleh metode konvensional (Desiani,2006).

Algoritma genetika pertama kali dikembangkan oleh John Holland dari Universitas Michigan pada tahun 1975. John Holland menyatakan bahwa setiap masalah yang berbentuk adaptasi (alami maupun buatan) dapat diformulasikan

kedalam terminologi genetika. Algoritma genetika adalah simulasi dari proses evolusi darwin dan operasi genetika atas kromosom (Kusumadewi,2003). Perbandingan istilah algoritma genetika dan sistem alamiah dapat ditunjukan pada tabel 2.1.

Tabel 2.1. Perbandingan istilah pada sistem alamiah dan algoritma genetika.

Sistem Alamiah	Algoritma Genetik
Kromosom	String
Gen	Fitur, Karakter, atau detector
Allel	Nilai fitur
Locus	Posisi String
Genotip	Struktur
Fenotip	Set parameter, solusi alternatif, struktur yang di-decode
Epitasis	Non linieritas

Beberapa definisi penting dalam algoritma genetika, yaitu :

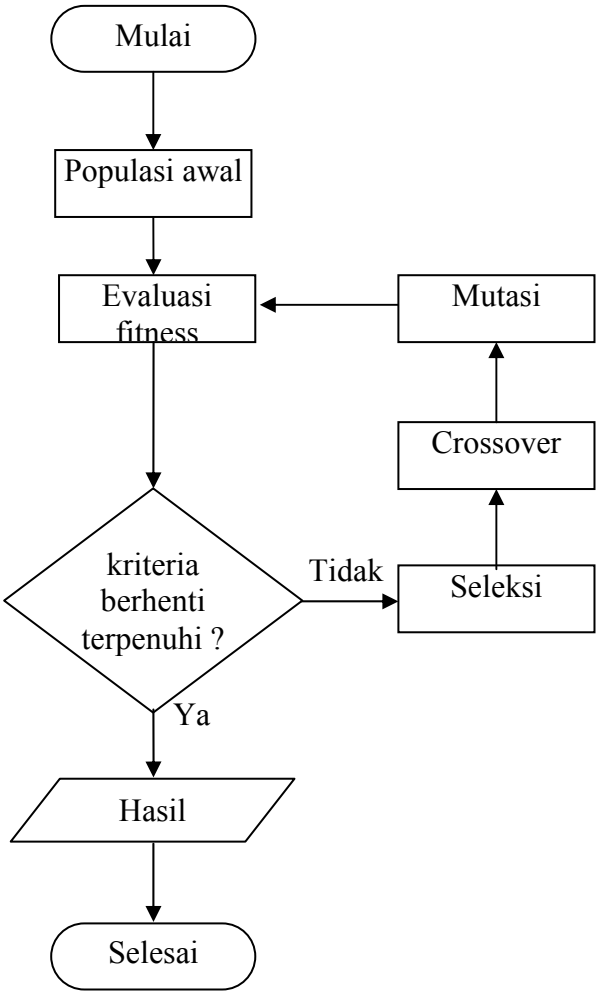
1. Genotype (Gen) adalah sebuah nilai yang menyatakan satuan dasar yang membentuk suatu arti tertentu dalam satu kesatuan gen yang dinamakan kromosom. Dalam algoritma genetika, gen ini bisa berupa nilai biner, float, integer maupun karakter.
2. Allele adalah nilai dari gen.
3. Kromosom adalah gabungan gen-gen yang membentuk nilai tertentu.

4. Individu menyatakan satu nilai atau keadaan yang menyatakan salah satu solusi yang mungkin dari permasalahan yang diangkat
5. Populasi merupakan sekumpulan individu yang akan diproses bersama dalam satu siklus proses evolusi.
6. Generasi menyatakan satu-satuan siklus proses evolusi.
7. Nilai Fitness menyatakan seberapa baik nilai dari suatu individu atau solusi yang didapatkan.

Ciri-ciri permasalahan yang dapat dikerjakan dengan menggunakan algoritma genetika adalah (Basuki,2003):

- Mempunyai fungsi tujuan optimalisasi non linear dengan banyak kendala yang juga non linear.
- Mempunyai kemungkinan solusi yang jumlahnya tak berhingga.
- Membutuhkan solusi “real-time” dalam arti solusi bisa didapatkan dengan cepat sehingga dapat diimplementasikan untuk permasalahan yang mempunyai perubahan yang cepat seperti optimasi pada pembebanan kanal pada komunikasi seluller.
- Mempunyai multi-objective dan multi-criteria, sehingga diperlukan solusi yang dapat secara bijak diterima oleh semua pihak.

Algoritma genetika secara umum dapat diilustrasikan dalam diagram alir yang dapat dilihat pada gambar 2.1.



Gambar 2.1. Diagram alir algoritma genetika

Keterangan gambar 2.1:

1. Populasi awal

   Proses ini merupakan proses yang digunakan untuk membangkitkan populasi awal secara random sehingga didapatkan solusi awal.

2. Evaluasi fitness

   Proses ini merupakan proses untuk mengevaluasi setiap populasi dengan menghitung nilai fitness setiap kromosom dan mengevaluasinya sampai terpenuhi kriteria berhenti.

3. Seleksi

Proses seleksi merupakan proses untuk menentukan individu-individu mana saja yang akan dipilih untuk dilakukan crossover. Keterangan selengkapnya dibahas pada sub bab 2.2.1 pada halaman 18.

4. Crossover

Proses crossover ini merupakan proses untuk menambah keanekaragaman string dalam satu populasi. Keterangan selengkapnya dibahas pada sub bab 2.2.2 pada halaman 23.

5. Mutasi

Mutasi merupakan proses mengubah nilai dari satu atau beberapa gen dalam suatu kromosom. Keterangan selengkapnya dibahas pada sub bab 2.2.3 pada halaman 27.

6. Kriteria berhenti

kriteria berhenti merupakan kriteria yang digunakan untuk menghentikan proses algoritma genetika.

7. Hasil

Hasil merupakan solusi optimum yang didapat algoritma genetika.

Struktur umum dari suatu algoritma genetika dapat didefinisikan dengan langkah-langkah sebagai berikut(Thiang,2001):

1. Membangkitkan populasi awal

Populasi awal ini dibangkitkan secara random sehingga didapatkan solusi awal. Populasi itu sendiri terdiri dari sejumlah kromosom yang merepresentasikan solusi yang diinginkan.

2. Membentuk generasi baru

Dalam membentuk generasi baru digunakan tiga operator yaitu operator reproduksi/seleksi, crossover dan mutasi. Proses ini dilakukan berulang-ulang sehingga didapatkan jumlah kromosom yang cukup untuk membentuk generasi baru dimana generasi baru ini merupakan representasi dari solusi baru.

3. Evaluasi solusi

Proses ini akan mengevaluasi setiap populasi dengan menghitung nilai fitness setiap kromosom dan mengevaluasinya sampai terpenuhi kriteria berhenti. Bila kriteria berhenti belum terpenuhi maka akan dibentuk lagi generasi baru dengan mengulangi langkah 2. Beberapa kriteria berhenti yang sering digunakan antara lain:

· Berhenti pada generasi tertentu.
· Berhenti setelah dalam beberapa generasi berturut-turut didapatkan nilai fitness tertinggi tidak berubah.
· Berhenti bila dalam n generasi berikut tidak didapatkan nilai fitness yang lebih tinggi.

Komponen-komponen utama algoritma genetika (Kusumadewi,2003):

1. Teknik pengkodean

Pengkodean adalah suatu teknik untuk menyatakan populasi awal sebagai calon solusi suatu masalah ke dalam suatu kromosom sebagai suatu kunci pokok persoalan ketika menggunakan algoritma genetika (Desiani,2006).

Teknik pengkodean ini meliputi pengkodean gen dan kromosom. Gen merupakan bagian dari kromosom. Satu gen bisanya akan mewakili satu variabel.

Gen dapat direpresentasikan dalam bentuk string bit, pohon, array bilangan real, daftar aturan, elemen permutasi, elemen program, atau representasi lainya yang dapat diimplementasikan untuk operator genetika.

2. Prosedur inisialisasi

Ukuran populasi tergantung pada masalah yang akan dipecahkan dan jenis operator genetika yang akan diimplementasikan. Setelah ukuran populasi ditentukan, kemudian harus dilakukan inisialisasi terhadap kromosom yang terdapat pada populasi tersebut. Inisialisasi kromosom dilakukan secara acak, namun demikian harus tetap memperhatikan domain solusi dan kendala permasalahan yang ada.

3. Evaluasi fitness

Evaluasi fitness merupakan dasar untuk proses seleksi. Langkah-langkahnya yaitu string dikonversi ke parameter fungsi, fungsi objektifnya dievaluasi, kemudian mengubah fungsi objektif tersebut ke dalam fungsi fitness. Dimana untuk maksimasi problem, fitness sama dengan fungsi objektifnya. Output dari fungsi fitness dipergunakan sebagai dasar untuk menseleksi individu pada generasi berikutnya (Syamsuddin,2004).

Untuk permasalahan minimalisasi, nilai fitness adalah inversi dari nilai minimal yang diharapkan. Proses inversi dapat dilakukan dengan (Basuki,2003):

fitness = A – f(x) atau $fitness = \frac{A}{f(x) + \varepsilon}$

keterangan :

A : konstanta yang ditentukan x : individu (kromosom)

ε : bilangan kecil yang ditentukan untuk menghindari pembagi nol atau f(x) = 0

4. Seleksi

Seleksi ini bertujuan untuk memberikan kesempatan reproduksi yang lebih besar bagi anggota populasi yang paling baik.

Ada beberapa metode seleksi dari induk, antara lain:

a. Rank-based fitness assigment
b. Roulette wheel selection
c. Stochastic universal sampling
d. Truncation selection
e. Tournament selection

5. Operator genetika

Operator genetika terdiri dari crossover dan mutasi. Crossover (perkawinan silang) bertujuan menambah keanekaragaman string dalam satu populasi. Mutasi merupakan proses mengubah nilai dari satu atau beberapa gen dalam suatu kromosom.

6. Penentuan parameter kontrol algoritma genetika

Kontrol parameter genetika diperlukan untuk mengendalikan operator-operator seleksi. Pemilihan parameter genetika menentukan penampilan kinerja algoritma genetika dalam memecahkan masalah. Ada dua parameter dasar dari algoritma genetika, yaitu probabilitas crossover (pc) dan probabilitas mutasi (pm).

Probabilitas crossover menyatakan seberapa sering proses crossover akan terjadi antara dua kromosom orang tua. Jika tidak terjadi crossover, satu orang tua dipilih secara random dengan probabilitas yang sama dan di duplikasi menjadi anak. Jika terjadi crossover, keturunan dibuat dari bagian-bagian kromosom orang

tua. Jika probabilitas crossover 100% maka keseluruhan keturunan dibuat dengan crossover. Jika probabilitas crossover 0% maka generasi baru dibuat dari salinan kromosom-kromosom dari populasi lama yang belum tentu menghasilkan populasi yang sama dengan populasi sebelumnya karena adanya penekanan selektif. Probabiltas mutasi menyatakan seberapa sering bagian-bagian kromosom akan dimutasikan. Jika tidak ada mutasi, keturunan diambil/disalin langsung setelah crossover tanpa perubahan. Jika mutasi dilakukan, bagian-bagian kromosom diubah. Jika probabilitas mutasi 100%, keseluruhan kromosom diubah. Jika probabilitas mutasi 0%, kromosom tidak ada yang diubah. Probabilitas mutasi dalam algoritma genetika seharusnya diberi nilai yang kecil. Umumnya, probabilitas mutasi diset untuk mendapatkan rata-rata satu mutasi per kromosom, yaitu angka/allele = 1/(panjang kromosom). Kemudian, hasil yang sudah pernah dicoba menunjukan bahwa angka probabilitas terbaik adalah antara 0,5% sampai 1%. Hal ini karena tujuan mutasi adalah menjaga perbedaan kromosom dalam populasi, untuk menghindari konvergensi prematur. Parameter lain yang juga ikut menentukan efisiensi kinerja algoritma genetika adalah ukuran populasi (popsize), yaitu banyaknya kromosom dalam satu populasi. Jika terlalu sedikit kromosom dalam populasi, algoritma genetika mempunyai kemungkinan sedikit untuk melakukan crossover dan hanya sebagian kecil dari ruang pencarian yang dieksplorasi. Sebaliknya, jika terlalu banyak jumlah kromosom ,algoritma genetika cenderung menjadi lambat dalam menemukan solusi (Desiani,2006).

Ada beberapa rekomendasi yang bisa digunakan dalam kontrol algoritma genetika, antara lain:

- Untuk permasalahan yang memiliki kawasan solusi cukup besar, De Jong merekomendasikan untuk nilai parameter kontrol:

  (popsize;pc;pm) = (50; 0,6; 0,001).

- Bila rata-rata fitnes setiap generasi digunakan sebagai indikator, maka Grefenstette merekomendasikan :

  (popsize;pc;pm) = (30; 0,95; 0,01).

- Bila fitness dari individu terbaik dipantau pada setiap generasi, maka usulannya adalah:

  (popsize;pc;pm) = (80; 0,45; 0,01).

- Ukuran populasi sebaiknya tidak lebih kecil dari 30, untuk sembarang jenis permasalahan

2.2.1 Seleksi

Seleksi akan menentukan individu-individu mana saja yang akan dipilih untuk dilakukan rekombinasi dan bagaimana anak terbentuk dari individu-individu terpilih tersebut. Langkah pertama yang dilakukan dalam seleksi ini adalah pencarian nilai fitness. Masing–masing individu dalam suatu wadah seleksi akan menerima probabilitas reproduksi yang tergantung pada nilai objektif dirinya sendiri terhadap nilai objektif dari semua individu dalam wadah seleksi tersebut. Nilai fitness inilah yang nantinya akan digunakan pada tahap-tahap seleksi berikutnya.

Ada beberapa definisi yang biasanya digunakan untuk melakukan perbandingan terhadap beberapa metode yang akan digunakan, antara lain:

- Selective pressure : probabilitas dari individu terbaik yang akan diseleksi dibandingkan dengan rata-rata probabilitas dari semua individu yang diseleksi.
- Bias : perbedaan absolut antara fitness dari suatu individu dan probabilitas reproduksi yang diharapkan.
- Spread : range nilai kemungkinan untuk sejumlah anak dari suatu individu.
- Loss of diversity : proporsi dari individu-individu dalam suatu populasi yang tidak terseleksi selama fase seleksi.
- Selection intensity : nilai fitness rata-rata yang diharapkan dalam suatu populasi setelah dilakukan seleksi.
- Selection variance : variasi yang diharapkan dari distribusi fitness dalam populasi setelah dilakukan seleksi.

Beberapa jenis seleksi yang umum dipakai adalah:

1. Rank-based Fitness

Pada rank-based fitness, populasi diurutkan menurut nilai objektifnya. Nilai fitness tiap-tiap individu hanya tergantung pada posisi individu tersebut dalam urutan, dan tidak dipengaruhi oleh nilai objektifnya.

Misalkan N adalah jumlah individu dalam suatu populasi. Pos adalah posisi individu dalam populasi tersebut (posisi terendah suatu individu adalah pos=1 , dan posisi tertingginya adalah pos=N). sedangkan sp adalah selective pressure. Nilai fitness dari suatu individu dapat dihitung sebagai:

- Linier ranking:

Fitness(pos) = 2- SP + 2 (sp -1)(pos-1)/(N-1)

Nilai spε[1,2].

- Non-linier rangking:

Fitness(pos) = Nind*X^(pos-1)/sum(X^(i-1)); i=1..N

Sedangkan x dihitung sebagai akar polinominal:

(sp-1) * X ^ (N-1) + sp * X ^ (N-2)+…+sp*X+sp = 0

Nilai spε[1,N-2].

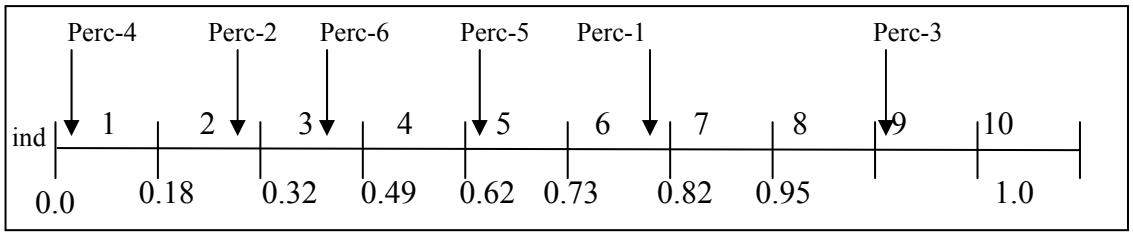
2. Seleksi Roda Roulette (Roulette Wheel Selection)

Metode seleksi roda roulette ini merupakan metode yang paling sederhana, dan sering juga dikenal dengan nama stochastic sampling with replacement. Pada metode ini, individu-individu dipetakan dalam suatu segmen garis secara berurutan sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran fitnessnya. Sebuah bilangan random dibangkitkan dan individu yang memiliki segmen dalam kawasan bilangan random tersebut akan terseleksi. Proses ini akan diulang hingga diperoleh sejumlah individu yang diharapkan .

Pada tabel 2.2 menunjukan probabilitas seleksi dari 11 individu. Individu pertama memiliki fitness terbesar, dengan demikian dia juga memiliki interval terbesar. Sedangkan individu ke 10 memiliki fitness terkecil kedua. Individu ke 11 memiliki nilai fitness terkecil (=0), interval terkecil sehingga tidak memiliki kesempatan untuk melakukan reproduksi.

Tabel 2.2 Probabilitas seleksi dan nilai fitness

Individu ke-	1	2	3	4	5	6	7	8	9	10	11
Nilai Fitness	2.0	1.8	1.6	1.4	1.2	1.0	0.8	0.6	0.4	0.2	0.0
Probabilitas seleksi	0.18	0.16	0.15	0.13	0.11	0.09	0.07	0.06	0.03	0.02	0.0



Gambar 2.2 Seleksi roda roulette

Setelah dilakukan seleksi, maka individu-individu yang terpilih adalah:

1 2 3 5 6 9

3. Stochastic universal sampling

Pada metode ini, individu-individu dipetakan dalam suatu segmen garis secara berurutan sedemikian hingga tiap-tiap segmen individu memiliki ukuran yang sama dengan ukuran fitnessnya seperti halnya pada seleksi roda roulette. Kemdian diberikan sejumlah pointer sebanyak individu yang ingin diseleksi pada garis tersebut. Andaikan N adalah jumlah individu yang akan diseleksi, maka jarak antar pointer adalah 1/N, dan posisi pointer pertama diberikan secara acak pada range [1,1/N].

4. Seleksi dengan pemotongan (Truncation selection)

Seleksi ini biasanya digunakan oleh populasi yang jumlahnya sangat besar. Pada metode ini, individu-individu diurutkan berdasarkan nilai fitnessnya. Hanya individu-individu yang terbaik saja yang akan diseleksi sebagai induk. Parameter yang digunakan dalam metode ini adalah suatu nilai ambang trunc yang

mengindikasikan ukuran populasi yang akan diseleksi sebagai induk yang berkisar antara 50%-10%. Individu-individu yang ada di bawah nilai ambang ini tidak akan menghsilkan keturunan.

5. Seleksi dengan turnamen (Turnament Selection)

Pada metode seleksi dengan turnamen ini, akan ditetapkan suatu nilai tour untuk individu-individu yang dipilih secara random dari suatu populasi. Individu-individu yang terbaik dalam kelompok ini akan diseleksi sebagai induk. Parameter yang digunakan pada metode ini adalah ukuran tour yang bernilai 2 sampai N (jumlah individu dalam suatu populasi).

6. $(\mu + \lambda)$ selection

Metode ini merupakan prosedur deterministik yang memilih kromosom terbaik dari orang tua dan keturunan. Metode ini biasanya digunakan pada masalah optimasi kombinatorial.

7. Steady-state reproduction

Pada metode ini sejumlah fitness parents yang terburuk digantikan dengan sejumlah individu baru (offspring).

8. Ranking and scaling

Ide dasar metode ini adalah mengurutkan berdasarkan ranking fitness-nya, kemudian menetapkan probabilitas seleksi tiap kromosom berdasarkan urutan ranking-nya.

9. Sharing

Teknik sharing diperkenalkan oleh Goldberg dan Richardson untuk optimasi dengan fungsi multi model. Teknik ini digunakan untuk menjaga

keanekaragaman populasi. Fungsi sharing adalah sebuah cara untuk menentukan degradasi fitness individu dikarenakan jaraknya dengan tetangga. Dengan adanya degradasi, probabilitas reproduksi individu pada puncak keramaian ditahan, individu lain akan memperoleh keturunan.

2.2.2 Crossover

Crossover (perkawinan silang) bertujuan menambah keanekaragaman string dalam satu populasi. Beberapa jenis crossover tersebut adalah

1. Crossover diskret

Proses crossover dilakukan dengan menukar nilai variabel antar kromosom induk. Misalkan ada 2 individu dengan 3 variabel, yaitu:

Induk 1 :	12	25	5
Induk 2 ;	123	4	34

Untuk tiap-tiap variabel induk yang menyumbangkan variabelnya ke anak dipilih secara random dengan probabilitas yang sama.

Sample 1 :	2	2	1
Sample 2 :	1	2	1

Kromosom baru yang terbentuk :

Anak 1 :	123	4	5
Anak 2 :	12	4	5

2. Crossover intermediate (menengah)

Crossover menengah merupakan metode crossover yang hanya dapat digunakan untuk variabel real. Nilai variabel anak dipilih di sekitar dan antara nilai-nilai variabel induk. anak dihasilkan menurut aturan sebagai berikut:

Anak = induk 1 + alpha (induk 2 – induk 1)

Dengan alpha adalah faktor skala yang dipilih secara random pada interval [-d,1+d], biasanya d = 0,25. tiap-tiap variabel pada anak merupakan hasil crossover variabel-variabel menurut aturan diatas dengan nilai alpha dipilih ulang untuk tiap variabel.

Misalkan ada 2 individu dengan 3 variabel, yaitu:

Induk 1 :	12	25	5
Induk 2 ;	123	4	34

Misalkan nilai alpha yang terpilih adalah;

Sampel 1 :	0,5	1,1	-0,1
Sample 2 :	0,1	0,8	0,5

Kromosom baru yang terbentuk:

Anak 1 :	67,5	1,9	2,1
Anak 2 :	23,1	8,2	19,5

3. Crossover garis

Pada dasarnya crossover garis ini sama dengan crossover menengah, hanya saja nilai alpha untuk semua variabel sama. Misalkan ada 2 individu dengan 3 variabel, yaitu:

Induk 1 :	12	25	5

Induk 2 ; 123 4 34

Alpha yang dipilih adalah;

Sample 1 : 0,5

Sample 2 : 0,1

Kromosom baru yang terbentuk :

Anak 1 : 67,5 14,5 19,5

Anak 2 : 23,1 22,9 7,9

4. Crossover satu titik

Proses crossover dilakukan dengan memisahkan suatu string menjadi dua bagian dan selanjutnya salah satu bagian dipertukarkan dengan salah satu bagian dari string yang lain yang telah dipisahkan dengan cara yang sama.

Misalkan ada 2 kromosom dengan panjang 12 :

Induk 1 : 0 1 1 1 0 | **0 1 0 1 1 1 0**

Induk 2 : **1 1 0 1 0** | 0 0 0 1 1 0 1

Posisi yang dipilih : 5

Kromosom baru yang terbetuk:

Anak 1 : 0 1 1 1 0 | 0 0 0 1 1 0 1

Anak 2 : **1 1 0 1 0 | 0 1 0 1 1 1 0**

5. Crossover banyak titik

Proses crossover ini dilakukan dengan memisahkan suatu string menjadi beberapa bagian dan selanjutnya dipertukarkan dengan bagian dari string yang lain yang telah dipisahkan dengan cara yang sama sesuai dengan urutanya.

Misalkan ada 2 kromosom dengan panjang 12 :

Induk 1 : **0 1 | 1 1 0 0 | 1 0 1 1 | 1 0**

Induk 2 : 1 1 | 0 1 0 0 | 0 0 1 1 | 0 1

Posisi yang dipilih : 5

Kromosom baru yang terbetuk:

Anak 1 : **0 1** | 0 1 0 0 | **1 0 1 1** | 0 1

Anak 2 : 1 1 | **1 1 0 0** | 0 0 1 1 | **1 0**

6. Crossover seragam

Kromosom seragam menghasilkan kromosom keturunan dengan menyalin bit-bit secara acak dari kedua orang tuanya.

Misalkan ada 2 kromosom dengan panjang 12

Induk 1 : **0** 1 1 1 **0 0** 1 0 1 **1 1 0**

Induk 2 : 1 **1 0 1** 0 0 **0 0 1** 1 0 1

Kromosom baru yang terbentuk:

Anak 1 : **0 1 0 1 0 0 0 0 1 1 1 0**

Anak 2 : 1 1 1 1 0 0 1 0 1 1 0 1

7. Crossover dengan permutasi.

Pada penyilangan permutasi ini kromosom-kromosom anak diperoleh dengan cara memilih sub barisan suatu tour dari satu induk dengan tetap menjaga urutan dan posisi sejumlah gen yang mungkin terhadap induk yang lainnya.

Contoh crossover dengan permutasi:

Misal

Induk 1 : (1 2 3 | 4 5 6 7 | 8 9)

Induk 2 : (4 5 3 | 1 8 7 6 | 9 2)

Anak 1 : (x x x | 1 8 7 6 | x x)

Anak 2 : (x x x | 4 5 6 7 | x x)

Dari sini kita memperoleh pemetaan:

1-4,8-5,7-6,6-7

Kemudian copy sisa gen di induk 1 ke anak 1 dengan menggunakan pemetaan yang sudah ada.

Anak 1 : (1-4 2 3 | 1 8 7 6 | 8-5 9)

Anak 1 : (4 2 3 | 1 8 7 6 | 5 9)

Lakukan hal yang sama untuk anak 2

Anak 2 : (4-1 5-8 3 | 4 5 6 7 | 9 2)

Anak 2 : (1 8 3 | 4 5 6 7 | 9 2)

2.2.3 Mutasi

Mutasi merupakan proses mengubah nilai dari satu atau beberapa gen dalam suatu kromosom. Mutasi ini berperan untuk menggantikan gen yang hilang dari populasi akibat proses seleksi yang memungkinkan munculnya kembali gen yang tidak muncul pada inisialisasi populasi. Beberapa cara operasi mutasi diterapkan dalam algoritma genetika menurut jenis pengkodean terhadap phenotype, antara lain:

1. Mutasi dalam pengkodean biner

Mutasi pada pengkodean biner merupakan operasi yang sangat sederhana. Proses yang dilakukan adalah menginversi nilai bit pada posisi tertentu yang

dipilih secara acak (atau dengan menggunakan skema tertentu) pada kromosom, yang disebut inversi bit.

Contoh mutasi pada pengkodean biner

Kromosom sebelum mutasi : 1 0 0 1 0 **1** 1 1

Kromosom sesdah mutasi : 1 0 0 1 0 **0** 1 1

2. Mutasi dalam pengkodean permutasi

Proses mutasi yang dilakukan dalam pengkodean biner dengan mengubah langsung bit-bit pada pada kromosom tidak dapat dilakukan pada pengkodean permutasi karena konsistensi urutan permutasi harus diperhatikan. Salah satu cara yang dapat dilakukan adalah dengan memilih dua posisi (locus) dari kromosom dan kemudian nilainya saling dipertukarkan.

Contoh mutasi dalam pengkodean permutasi

Kromosom sebelum mutasi : 1 2 **3** 4 6 5 8 **7** 9

Kromosom sesudah mutasi : 1 2 **7** 4 6 5 8 **3** 9

3. Mutasi dalam pengkodean nilai

Proses mutasi dalam pengkodean nilai dapat dilakukan dengan berbagai cara, salah satunya yaitu dengan memilih sembarang posisi gen pada kromosom, nilai yang ada tersebut kemudian ditambahkan atau dikurangkan dengan suatu nilai kecil tertentu yang diambil secara acak.

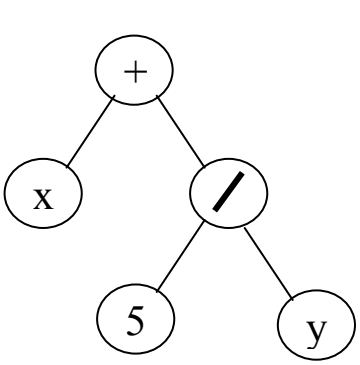
Contoh mutasi dalam pengkodean nilai riil dengan nilai yang ditambahkan atau dikurangkan adalah 0,1

Kromosom sebelum mutasi : 1,43 **1,09** 4,51 **9,11** 6,94

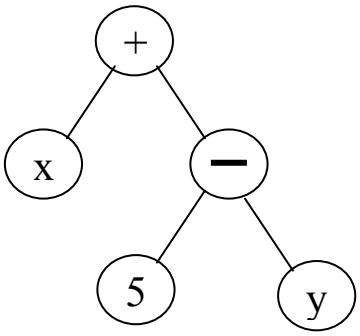
Kromosom sesudah mutasi : 1,43 **1,19** 4,51 **9,01** 6,94

4. Mutasi dalam pengkodean pohon

Mutasi dalam pengkodean pohon dapat dilakukan antara lain dengan cara mengubah operator (+, -, *, /) atau nilai yang terkandung dalam suatu verteks pohon yang dipilih. Atau dapat juga dilakukan dengan memilih dua verteks dari pohon dan saling mempertukarkan operator atau nilainya. Contoh mutasi dalam pengkodean pohon seperti pada gambar 2.3.



Sebelum mutasi



Sesudah mutasi

Gambar 2.3. Contoh gen sebelum dan sesudah mutasi dengan pengkodean pohon

1.3. BAHASA PEMROGRAMAN VISUAL C++

2.3.1. Sejarah Bahasa C++

Bahasa C++ pertama kali dikembangkan oleh Bjarne Stoustrup di Bell Lab pada awal tahun 1980-an. Bahasa C++ diturunkan dari bahasa sebelumnya yaitu bahasa C. Untuk mendukung fitur-fitur pada C++, dibangun efisiensi dan system support untuk pemrograman tinggkat rendah (low level coding). Pada C++ ditambahkan konsep pemrograman berorientasi objek.

2.3.2. Pemrograman dengan Visual C++

Visual C++ merupakan pemrograman windows yang menggunakan struktur bahasa C++. Pemrograman windows merupakan pembuatan perangkat lunak yang dapat dijalankan didalam system operasi windows. Visual C++ memiliki IDE (Integrated Development Environment) yang berfungsi untuk membuat, mengkompilasi menghubungkan / menggabungkan (linking), dan menguji program. Secara visual IDE tersusun menjadi 4 bagian (gambar 2.4.), yaitu

1. Menu dan toolbar

   menu digunakan untuk mengakses berbagai pilihan tidak hanya untuk mengatur konfigurasi IDE tetapi juga untuk project secara keseluruhan. Misalnya untuk mengatur tampilan IDE dan untuk mengkompilasi program. Selain dengan menggunakan menu, untuk melakukan hal yang sama juga dapat dilakukan melalui toolbar. Jadi toolbar adalah jalan pintas (short cut) untuk mengakses menu.

2. Window untuk project view (project workspace windows)

   window untuk project view berfungsi sebagai pintu gerbang untuk dapat mengakses segala aspek yang terdapat didalam suatu project. Project view memiliki tiga buah view untuk dapat melihat project dari tiga macam aspek sesuai dengan namanya, yaitu class view, resource view, dan file view. Calss view dipakai untuk melihat berbagai macam class yang terdapat didalam project. File view dipakai untuk melihat berbagai macam file program.

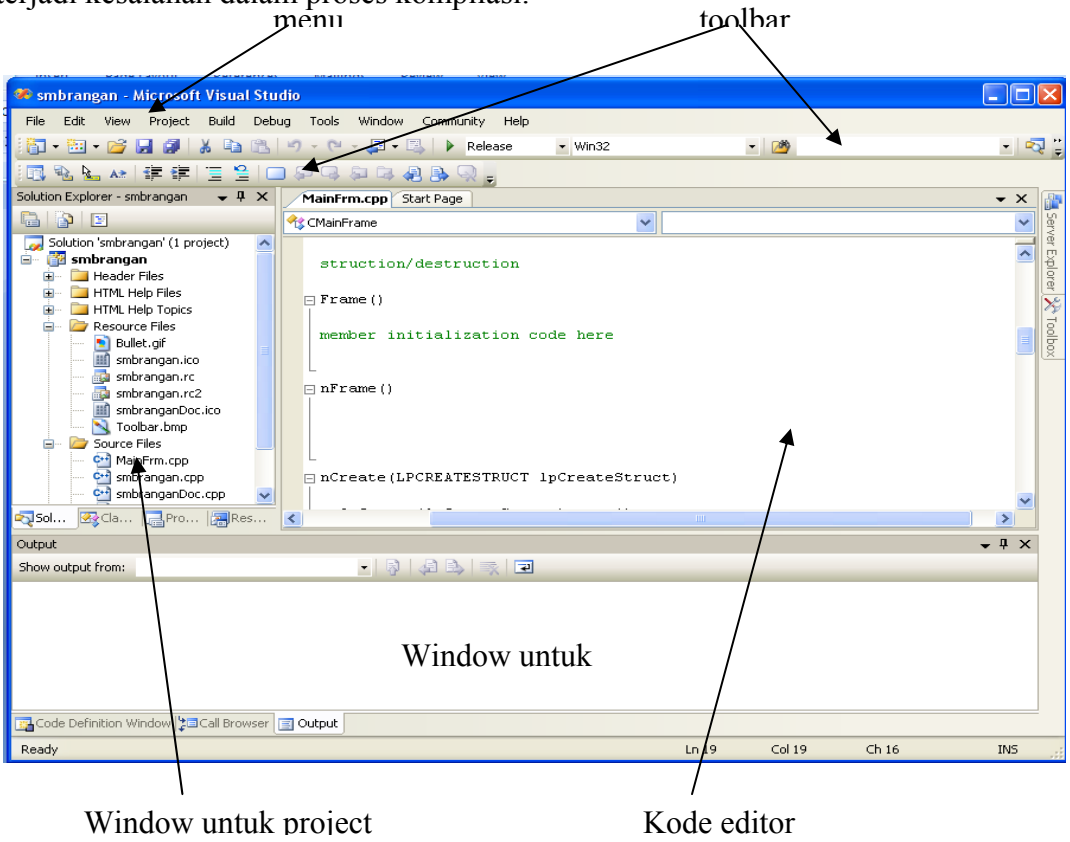
Sedangkan resource view dipakai untuk melihat tampilan visual suatu interface dari aplikasi yang sedang dibangun, misalnya kotak dialog dan icon.

3. Kode/ text editor (editor window)

   kode/ text editor (editor window) berfungsi sebagai tempat menulis dan mengedit berbagai macam kode program yang terdapat pada berbagai macam class dan juga digunakan untuk mendesain dan mengedit Graphical User Interface (GUI) dari suatu aplikasi, kotak dialog, icon dan lain-lain.

4. Window untuk debug (output window)

   window untuk debug berfungsi untuk menampilkan informasi-informasi penting selama proses pembuatan suatu aplikasi atau dalam pengujian suatu bagian program. Window ini akan menampilkan pesan-pesan kesalahan jika terjadi kesalahan dalam proses kompilasi.



Gambar 2.4. Komponen IDE secara visual

2.3.2.1. Pendeklarasian Variabel dan Konstanta

Pada C++ semua variabel yang akan digunakan dalam program harus dideklarasikan terlebih dahulu. Deklarasi sebuah variabel memiliki bentuk umum:

Type Nama_variabel

Type **:** menentukan tipe variabel

Nama_variabel **:** menentukan nama variabel yang digunakan dalam program. Jika ada lebih dari satu variabel dengan tipe sama maka bisa dipisahkan dengan tanda koma.

Dalam mendefinisikan konstanta pada C++ terdapat 3 cara yang dapat digunakan. pertama konstanta didefinisikan dengan menggunakan preprosesor directive #define. Contoh penggunaan preprosesor directive #define yaitu

#define pi 3.14

Teknik kedua adalah dengan menggunakan keyword const saat mendeklasikan variabel. Contoh penggunaan keyword const:

const float pi 3.14

Teknik ketiga adalah enumeration. Enumeration mendefinisikan tipe baru dan batasan nilai hanya pada kumpulan nilai yang ditentukan oleh programmer. Sebagai contoh :

enum Color{RED,BLUE,GREEN}

2.3.2.2. Pernyataan If

pernyataan if digunakan untuk menjalankan blok – blok kode program jika tes kondisi yang digunakan bernilai benar. Jika tes kondisi bernilai benar, maka

blok-blok kode dijalankan. Jika tidak, maka blok-blok kode tersebut dilewati. Bentuk umum dari pernyataan if:

```
if (kondisi)
{ statement }
 Atau
if (kondisi)
{ Statement 1}
else
{ Statement 2 }
Atau
if (kondisi)
{ Statement 1}
else if
{ Statement 2 }
else
{ Statement 3}
```

2.3.2.3. Pernyataan Switch

Switch digunakan untuk menggantikan penggunaan if yang berurutan atau penggunaan if dalam if yang banyak. Bentuk umum dari pernyataan switch:

```
switch (variabel){
case ekspresi 1:
        Statement
        break;
```

```
case ekspresi 2:
        Statement
        break;
…
default:
        Statement
}
```

2.3.2.4. Pernyataan While

Perulangan while digunakan untuk mengeksekusi blok kode selama suatu kondisi bernilai benar. Jika kondisi bernilai salah dari awal maka blok kode tidak akan dieksekusi. Sintaks perulangan dengan while:

```
while (kondisi yang diuji terpenuhi)
{
blok kode
}
```

2.3.2.5. Pernyataan Do

Perulangan do mengeksekusi blok kode selama suatu kondisi terpenuhi. Perulangan do melakukan tes kondisi pada akhir perulangan. Oleh karena itu blok kode dieksekusi palng tidak satu kali. Sintaks dari perulangan do:

```
do
{
blok kode
} while(kondisi yang diuji)
```

2.3.2.6. Pernyataan For

Perulangan for digunakan untuk mengeksekusi sebuah blok kode untuk sejumlah perulangan yang sudah tertentu jumlah perulangannya. Sintaks umumnya:

```
for(initialization; test condition; action)
{
blok kode
}
```