Tugas Besar 1 (CPP)

1 Outcomes

Pada tugas besar C++ ini, anda diharapkan:

1. Mampu merancang dan mengimplementasikan kelas sesuai dengan kaidah yang sudah diajarkan di kelas.
2. Mampu memakai STL dan menerapkan kelas dengan hubungan inheritance & generik dengan baik dan benar.
3. Mampu membuat kelas yang menangani exception dengan baik.
4. Mampu membuat aplikasi dengan menggunakan ulang kelas sebaik-baiknya untuk sekumpulan aplikasi, dan dapat menunjukkan perbedaan satu versi dengan versi berikutnya. Kelak, pengalaman ini akan membantu mahasiswa dalam melakukan versioning dan braching saat membangun aplikasi yang lebih besar
5. Mampu menggunakan tools yang dipakai untuk pengembangan/pemodelan dan untuk dokumentasi *source code*
6. Mampu bekerja dalam tim dengan pembagian kerja yang baik,
7. Mampu mengkomunikasikan hasil kerja dalam bentuk laporan-laporan yang ditentukan.

2 Deskripsi Umum Tugas

Seseorang sedang memesan aplikasi untuk mengamati pergerakan makhluk pada jagat raya yang akan ditampilkan di layar, dan observasinya dapat direka. Untuk itu, Anda diminta untuk membuat sebuah aplikasi desktop dengan tampilan hanya berupa karakter saja, dikembangkan dalam bahasa C++, yang secara garis besar perilakunya dijelaskan sebagai berikut:

1. Saat aplikasi dijalankan, pada **bidang layar** akan muncul sekumpulan **makhluk-makhluk** yang merupakan turunan dari suatu makhluk abstrak. Makhluk abstrak tersebut dimunculkan dalam bentuk-bentuk geometris (referensi: kelas Figure dan semua turunannya yang dijelaskan pada konsep inheritance). Bidang layar adalah suatu area berbentuk segi empat yang terdiri dari **sel** dengan satuan ukuran adalah satu karakter, dan tentunya dapat dikonfigurasi sesuai dengan ukuran layar pengguna (diinisiasi nilainya saat aplikasi dihidupkan), dan tak mungkin berubah sampai aplikasi dihentikan atau berhenti. Untuk memudahkan persoalan, sebuah makhluk akan ditampilkan sebagai sebuah karakter saja.
2. Setiap saat (deltaT), setiap makhluk akan bergerak sesuai dengan **arah mata angin** yang menjadi ciri dirinya. Yang dimaksud dengan arah mata angin adalah [https://id.wikipedia.org/wiki/Mata_angin], yaitu: U (Utara), TL (Timur Laut), T (Timur), TG (Tenggara), S (Selatan), BD (Barat Daya), B (Barat), BL (Barat Laut). Arah pergerakan sebuah makhluk dapat berubah sesuai dengan keadaan yang dapat anda deskripsikan pada saat berada di bidang layar maupun pada tepian layar.
3. Setiap sel hanya dapat ditempati oleh maksimal N buah makhluk, dengan $N \geq 1$. Jika pada saat pergerakan ada lebih dari satu makhluk menempati sebuah sel dalam bidang layar, maka terjadilah **persaingan kehidupan** yang berakibat hanya tersisa N -1 makhluk. Algoritma persaingan kehidupan antar makhluk harus anda deskripsikan, dengan mengacu ke **kekuatan** setiap makhluk.
4. Program berhenti jika:
    a. Makhluk-makhluk yang muncul habis, yaitu tak ada lagi makhluk yang dapat ditampilkan
    b. Pengguna menekan suatu kunci yang mengakibatkan program berhenti.
5. Selain menghentikan program dengan penekanan suatu kunci, aplikasi juga menangani penekanan kunci-kunci keyboard rahasia (yang boleh anda usulkan), untuk:
    a. Menghentikan sementara aplikasi, dan "memotret" bidang serta menyimpannya dalam file. Tentu, setelah dihentikan, pengguna dapat melanjutkan pengamatan.

b. *Step-by step execution*, yaitu bahwa setelah display pada saat T, aplikasi menunggu penekanan kunci tertentu untuk ditampilkan posisi pada T+DT
c. Melahirkan sejumlah tertentu Makhluk (secara random), jika ternyata pada observasi diamati bahwa makhluk makin sedikit.

Batasan implementasi secara umum:

1. Setiap kelas (kecuali main program dan kelas dengan hubungan *inheritance* yang kuat), harus dapat dikembangkan dan ditest secara terpisah, dan harus dilengkapi dengan driver.
2. Setiap kelas harus jelas siapa pengembangnya, dan dituliskan dalam header file sebagai komentar
3. Setiap kelas harus "*self documented*", artinya terdokumentasi dengan baik, dapat dimengerti, dan dokumentasinya dapat dibangkitkan otomatis dengan *tools*.

Pemesan aplikasi menghendaki beberapa versi aplikasi, yang deskripsinya diberikan sebagai berikut:

No. Versi	Deskripsi Khusus Produk	Konstraint Implementasi
S-1a	Perilaku pada butir 5 tidak perlu diimplementasikan	Himpunan makhluk harus direpresentasi sebagai **list of Makhluk. List harus dikelola sendiri**, anda boleh memilih representasi kontigu atau berkait. Makhluk dan turunannya harus dirancang sebagai sebuah kumpulan kelas yang berhubungan inheritance, dengan sbuah root class sebuah kelas abstract
S-2a	Perilaku pada butir 5 tidak perlu diimplementasikan	Himpunan makhluk harus direpresentasi dengan **STL**, dengan memilih struktur yang tepat. Untuk ini, anda harus mengambil semua kelas riil dari taksonomi kelas dengan hubungan inheritance, dan anda ~~tidak~~ boleh menerapkan atau tidak menerapkan hubungan inheritance. ~~Tidak ada inheritance dalam versi ini~~
S-1b	Pengembangan dari S-1a, yaitu dengan mengimplementasi perilaku pada butir 5	Idem S-1a (optional). Jika berhasil dibuat, maka cukup diserahkan S-1b saja
S-2b	Pengembangan dari S-2a, yaitu dengan mengimplementasi perilaku pada butir 5	Idem S-2a (optional). Jika dibuat maka cukup diserahkan S-2b saja
H-1	Ada konkurensi, artinya setiap makhluk adalah thread dan ini adalah program konkuren paling sederhana sebab tidak ada komunikasi/sinkronisasi antar thread	Anda boleh memilih salah satu representasi internal yang paling optimal untuk versi H ini, dan menangani semua penekanan kunci yang ditangani sesuai dengan versi yang anda pilih (**belum atau sudah** mengimplementasi butir 5)

Catatan:
Dengan demikian, setoran wajib adalah 3 versi aplikasi, setiap aplikasi harus digambarkan secara lengkap diagram kelasnya. Jika anda membuat S-1b dan S-2b, maka versi S-1a dan versi S-2a harus tetap disimpan aplikasinya, untuk dapat dicek, karena pengembangan versi *b hanya melakukan modifikasi pada kelas main saja. Secara intuitive, anda belajar mengenai membuat branching dalam proses versioning, yang kelak akan anda pelajari teorinya.

3 Tugas Anda

Tugas anda adalah:

1. Membentuk kelompok yang terdiri dari 4 mahasiswa dengan syarat versi S-1a + S-2a dan S-1b + S-2b dikerjakan oleh mahasiswa berbeda

2. Memahami persoalan di atas, dan membuat sketsa solusi yang memungkinkan untuk membagi pekerjaan, sehingga setiap anggota kelompok dapat membangun kelas secara independen dan menjadi nilai dirinya.
3. Merancang kelas-kelas yang sedapat mungkin dapat digunakan ulang untuk semua versi. Nama kelas, atribut dan method harus bermakna sesuai dengan konteks di atas, dan anda harus membuat spesifikasi yang jelas dalam bentuk komentar dalam program, yang akan diekstraksi dokumentasinya dengan Doxygen.
4. Membuat semua versi sesuai dengan deskripsi di atas, dan mendemokan ke asisten untuk dinilai, sesuai dengan script demo yang anda rancang. Kelengkapan script demo akan menjadi salah satu kriteria penilaian.

4 Jadwal Pengerjaan Tugas

Week	Tanggal/Perioda	Deskripsi Aktivitas	Deliverable
W06	22-Feb-2016	Rilis Tugas	
	24-Feb-2016	Klarifikasi dan pengerjaan Tugas di kelas	Daftar klarifikasi
	25-Feb-2016	Mengerjakan tugas di lab di Lab	Sketsa solusi (hardcopy) Pembagian tugas (hardcopy)
W07	29-Feb-2016	ReDesign kelas, Unit implementation	
	03-Mar-2016	Coding Unit	Upload unit disertai driver. 1 kelas per mahasiswa
W08	07-11 Mar 2016	Bekerja mandiri, mentoring berdasarkan perjanjian. Kamis tidak perlu bekerja di lab	
W09	14-18-Mar-2016	Minggu UTS: pending tugas	
W10	22-Mar-2016	Pengumpulan semua deliverables (hard copy, upload softcopy)	Pembagian Tugas dan Log activity Dokumentasi Teknis PL Poster Spesifikasi Produk (hanya 1 versi yang dipilih)
	24-Mar-2016	Demo Penilaian Tugas	Evaluasi diri

5 Kerangka deliverables

1. *Source code*: sesuai dengan batasan yang disebutkan (*unit test*, *well explianed*, *well documented*).
2. Pembagian Tugas dan Log Activity
    a. Peran setiap orang terhadap kelas baik di tingkat unit maupun di tingkat aplikasi (developer, tester)
    b. Log activity: individu maupun kelompok, harus jelas siapa mengerjakan apa, hasilnya apa pada perioda apa
3. Poster hanya untuk satu versi yang dipilih
    a. Satu lembar berisi deskripsi aplikasi dan keunggulan teknisnya (untuk marketing)
4. Kerangka Dokumentasi teknis
    a. Deskripsi umum aplikasi dan variannya
    b. Daftar rancangan Seluruh kelas dan penggunaan ulangnya, dalam bentuk tabel sbb:

(kelas yang mirip harus diberi nama mirip)

Kelas	Type kelas	S-1a	S-2a	S-1b	S-2b	H-1
Kelas-1	ADT	ADT1	ADT-1 dengan modif minor	-	-	ADT-1 sama dengan S-2a
Kelas-2	Mesin					
Kelas-N	main					

c. Diagram Paket dan diagram kelas dengan tools, minimal hanya untuk dua aplikasi.
d. Source code documentation dengan Doxygen, hanya untuk salah satu aplikasi saja yang anda pilih.

6 Kerangka Program Utama

Program utama adalah program yang menghidupkan sekumpulan makhluk, dan menunggu ketukan kunci. File program utama walaupun bukan kelas, dianggap sebagai sebuah kelas.

```
Inisialiasi           // hidupkan makhluk-makhluk
repeat
  // tentukan posisi awal
  repeat              // tunggu sampai user tekan kunci atau mahluk habis
                      // jika habis maka stop:= true
    // display makhluk
    // gerakkan
  until  GetKunci() atau MahlukHabis;
  If GetKunci() then
     // proses analisa kasus tergantung kunci yang ditekan
  End-if
until stop;
Terminasi        // clean up
```

Operasi File

Untuk operasi baca/tulis file, silahkan mempelajari contoh di:
http://www.cplusplus.com/doc/tutorial/files/
http://www.tutorialspoint.com/cplusplus/cpp_files_streams.htm

Sedangkan jika anda ingin memakai format JSON:
http://marc.helbling.fr/2015/02/writing-json-c++

Threading with CPP

http://en.cppreference.com/w/cpp/thread
http://www.cplusplus.com/reference/thread/thread/
http://www.bogotobogo.com/cplusplus/C11/1_C11_creating_thread.php