Spesifikasi Tugas Kecil
IF2210 Pemrograman Berorientasi Objek
ArkavQuarium

Pendahuluan

Insaniquarium adalah sebuah *game* di mana kita dapat memelihara ikan yang akan menghasilkan uang yang dapat dikumpulkan. Ada berbagai jenis ikan yang dapat dipelihara di antaranya adalah *guppy*, piranha, dan sebagainya. Ikan-ikan tersebut harus diberi makan dan dirawat agar tidak mati. Selain ikan yang dapat dipelihara ada juga *pet* yang dapat membantu dalam memelihara ikan atau mengumpulkan uang seperti siput, kerang, dll. Kita juga harus melindungi ikan di dalam akuarium dari serangan alien yang dapat muncul sewaktu-waktu.

Gambar 1. Permainan *Insaniquarium*

Pada tugas kecil kali ini anda diminta untuk **membuat desain kelas** sebuah simulasi permainan Insaniquarium bernama **ArkavQuarium** menggunakan pengetahuan yang sudah didapat pada mata kuliah Pemrograman Berorientasi Objek.

Deskripsi ArkavQuarium

Pada program ini, terdapat sebuah akuarium yang berisi ikan dan entitas-entitas lainnya.

1. Akuarium
    - Akuarium dapat dianggap sebagai sebuah matriks 2D (sumbu X dan Y) dengan ukuran yang sama dengan ukuran layar.
    - Posisi dinyatakan dalam notasi (x, y), di mana x merupakan posisi pada sumbu X/horizontal dan y merupakan posisi pada sumbu Y/vertikal.
    - Posisi ujung kiri-atas akuarium memiliki koordinat (0,0). Semakin ke bawah, maka nilai y semakin besar. Semakin ke kanan, nilai x semakin membesar.
    - Setiap entitas ikan/makanan ikan/koin/piranha yang berada di akuarium harus dicatat posisinya.
    - Ketika dilakukan inisialisasi, muncul beberapa guppy, satu piranha, dan satu siput. Posisi setiap entitas harus ditentukan secara acak.
2. Guppy
    - Guppy muncul secara berkala secara *random*.
    - Guppy dimunculkan pada posisi *random* dan dalam kondisi tahap pertumbuhan ke-1.
    - Guppy memiliki **tiga tahap pertumbuhan (1-3).**
    - Guppy dapat pindah dari tahap pertumbuhan lebih kecil ke tahap pertumbuhan lebih besar **setelah memakan sejumlah makanan**.
    - Guppy hanya dapat memakan makanan yang ada di dekatnya (makanan ada di radius tertentu dari guppy).
    - **Setiap C waktu, guppy akan mengeluarkan koin**. Nilai koin harus berbeda pada setiap tahapan guppy, dan guppy yang lebih besar harus mengeluarkan koin dengan nilai lebih besar.
    - Jika tahap pertumbuhan guppy sudah tahap ke-3, maka ikan tidak dapat tumbuh ke tahap lebih tinggi.
    - Guppy dapat bergerak ke arah apapun (360 derajat). Gerakan guppy memiliki kecepatan tetap tanpa memperhatikan arah gerak guppy (guppy tidak boleh bergerak lebih cepat ketika bergerak diagonal, atau sebaliknya).
    - Guppy tidak boleh bergerak keluar dari akuarium.
    - Tampilan guppy harus mengikuti arah gerak pada sumbu horizontal. Jika guppy sedang bergerak ke kanan (atau kanan atas, kanan bawah, dst..), maka tampilan guppy adalah melihat ke kanan, dan sebaliknya. Gerakan pada sumbu vertikal tidak mempengaruhi tampilan.
    - Guppy dapat berjalan-jalan **dalam keadaan kenyang dengan arah bebas selama T waktu**. Pada periode ini, guppy tidak mungkin memakan makanan.
    - Setelah T waktu, guppy akan membutuhkan makanan dan harus dipenuhi **dalam waktu M**. Jika guppy ingin makan dan ada makanan guppy di dalam akuarium, dia akan bergerak ke arah **makanan terdekat**. Jika makanan guppy yang

sedang didekati hilang (dimakan guppy lain atau jatuh ke bawah) dan tidak ada makanan lain, maka guppy bergerak dengan arah bebas.
    - Jika dalam waktu M guppy tidak menemukan makanan, guppy mati.
3. Makanan Ikan
    - Makanan ikan muncul secara acak di bagian atas akuarium.
    - Makanan ikan selalu bergerak ke bawah.
    - Kecepatan gerak makanan ikan harus lebih lambat dari kecepatan ikan.
    - Ketika makanan ikan sampai di dasar, **makanan tersebut hilang**.
4. Koin
    - Koin memiliki suatu nilai.
    - Koin selalu bergerak ke bawah.
    - Ketika koin sampai di dasar, koin tersebut diam.
5. Piranha
    - Piranha pada dasarnya sama seperti guppy (periode makan, cara bergerak, dst.). Perbedaannya akan dijelaskan pada poin-poin selanjutnya.
    - Piranha **tidak memakan makanan ikan**. Makanan piranha adalah guppy dengan tahap apapun.
    - Koin yang dihasilkan piranha tidak dihasilkan secara periodik, tetapi **langsung muncul** setelah dia makan. Nilai koin adalah **harga ikan * (tahap guppy yang dimakan + 1)**.
6. Siput
    - Siput hanya ada di dasar akuarium.
    - Siput bergerak ke arah koin terdekat yang ada di dasar.
    - Jika tidak ada koin yang di dasar, siput bergerak ke arah koin yang paling dekat ke dasar.
    - Jika tidak ada koin, siput diam.
    - Siput mengambil koin yang ada di dekatnya (koin ada di radius tertentu dari siput).

Spesifikasi

Berikut adalah spesifikasi dari tugas kecil ini.

1. Tugas ini **dikerjakan berkelompok** sebesar 4 orang.
2. Buatlah desain kelas untuk implementasi seluruh deskripsi ArkavQuarium yang telah dijelaskan sebelumnya dengan **pendekatan berorientasi objek** (classes dan objects).
3. Desain kelas Anda dibuat pada header C++ (file .h atau .hpp).
4. Lengkapi setiap kelas dengan method dan atribute yang tepat, dengan komentar yang menjelaskan kegunaan setiap kelas, method, dan atribut.
5. Setiap kelas harus ada di file terpisah.
6. Desain kelas Anda harus mencantumkan diagram kelas. Gunakanlah alat untuk mengenerate diagram kelas tersebut (contoh: Doxygen).
7. Buatlah sebuah file Readme yang berisi deskripsi singkat setiap kelas.

8. Desain kelas akan digunakan di tugas besar selanjutnya. Usahakan buat desain kelas yang tepat pada tugas kecil ini, karena **setiap perubahan desain kelas pada tugas besar harus disertai dengan alasan perubahan**.
9. Desain yang Anda buat harus memiliki elemen pemrograman berorientasi objek di bawah ini.
    a. Inheritance (“Is-a” relationship)
    b. Method override pada inheritance
    c. Aggregation (“Has-a” relationship)
    d. Polymorphism
    e. Generic
    f. Abstract class
    g. Function/method overloading
    h. Interface (diimplementasikan sebagai abstract class tanpa implementasi)
    i. Multiple inheritance (disarankan satu class dan satu atau lebih interface)
10. Manfaatkan elemen-elemen pemrograman berorientasi objek tersebut untuk membuat program yang baik, seperti:
    a. Tidak terdapat kode duplikat
    b. Memiliki struktur kelas yang mudah dipahami
    c. Implementasi kelas tidak terlalu kompleks
    d. Mengikuti prinsip SOLID.
11. Selain entitas-entitas yang sudah disebutkan sebelumnya, program yang Anda **diwajibkan mengimplementasikan kelas *linked list*** yang dapat menyimpan **tipe *generic*.**
12. Jika linked list ini digunakan di minimal dua tempat dengan tipe yang berbeda, maka linked list memenuhi spek desain generic.
13. Berikut adalah method *linked list* **minimal** (boleh Anda tambah sesuai kebutuhan) yang harus dibuat.
    a. find(T element) → int
       Method ini mengembalikan indeks di mana elemen berada pada *linked list*, dan -1 jika tidak ada.
    b. isEmpty()
       Method ini mengembalikan nilai True jika *linked list* kosong, dan False jika sebaliknya.
    c. add(T element) → void
       Method ini menambahkan elemen sebagai elemen paling belakang pada LinkedList
    d. remove(T element) → void
       Method ini membuang elemen dengan identitas demikian
    e. get(int index) → T
       Method ini mengembalikan elemen dengan tipe T pada indeks ke-i.

Deliverable dan Milestone

Deliverable yang harus dikirimkan adalah:

1. Folder *src* yang berisi:
    - Seluruh file header yang dibuat (boleh dimasukkan ke subfolder untuk memperjelas pembagian kelas)
2. Folder doc yang berisi:
    - File diagram kelas
    - Readme.txt sesuai spesifikasi tugas

Semua deliverable tersebut di-zip dengan format nama:

TucilOOP_<NIM1>_<NIM2>_<NIM3>_<NIM4>.zip

Deliverable dikumpulkan selambat-lambatnya pada hari Kamis, 15 Maret 2018 pukul 23.55 melalui *uploader* di Olympia.