Domande dell’orale di Programmazione

* Compilare il codice.
* C’era un pezzo di codice con dei calcoli poco chiari fatti con un vettore 2D (SFML) e le funzioni std::cos e std::sin. Hanno chiesto di riscrivere il codice facendo operazioni con gli angoli invece che con i vettori. **Soluzione:** convertire il vettore in un angolo, trovare il nuovo angolo, sommarli e poi creare un nuovo vettore utilizzando il nuovo angolo. (N.B: soluzione utile nel nostro specifico caso, non è una cosa generale.)
* Cosa è l’invariante di classe? Come avete fatto a fare in modo che fosse rispettata? Dove va verificato che venga rispettata? **Risposta:** l’invariante di classe è una relazione che sussiste in ogni momento fra gli elementi di una classe e che giustifica l’esistenza stessa della classe. La correttezza dell’invariante di classe va verificata sia all’interno del codice con opportune verifiche all’interno dei metodi della classe, sia con i test.
* Il programma ad un certo punto fa due iterazioni attraverso lo stesso vettore per contare due tipi di elementi diversi. Come si sarebbe potuto evitare di fare due iterazioni? **Soluzione:** utilizzare std::accumulate e contare il risultato all’interno di una struct che contiene il numero di entrambi i tipi di elementi.
* Da dov’è venuta l’idea per il progetto? Come mai avete scelto queste soluzioni per implementarlo?
* Argomento a piacere da esporre.

1. Eseguire il programma
2. Generatore di numeri casuali: come risolvere il problema della generazione fuori range? (normalizzare il risultato nel range: no, crea un bias. Scartare il risultato e generarne uno nuovo)
3. Double: spiegarli
4. Divisione della memoria e a cosa servono i vari pezzi
5. Dove viene allocata la memoria del programma creando una variabile? (lo stack). Con new? (lo heap)
6. Contenuto della memoria all’avvio (stack allocato per la funzione main, si allocano le variabili globali, la memoria di iostream, eventuali variabili static)
7. C’è qualcosa nello heap se non abbiamo allocato memoria dinamica? (no)
8. Il compilatore NON alloca lo stack frame, viene fatto al runtime
9. Come il programma alloca la memoria nello heap: allocatore. Come lo implementeresti? (solo ragionamento teorico). Come individuare le celle di memoria vuote?
10. Come funziona l’allocazione di memoria di vector?
11. Discussione di ciò che hai scritto nella relazione
12. Visione del codice: questi due pezzi sono molti simili, si potrebbero accorpare?
13. Discussione del codice senza fare modifiche (le chiede solo battilana)

2)

Eseguire il programma

Mettere gli assert: devi verificare le condizioni iniziali

Errore: fare un hpp per tanti cpp (dividere header e cpp)

Correggere pass by value con pass by const reference

Togliere la parete inutile (il biliardo non ha parete posteriore)

Pass by const reference

Shared PTR: come li scriveresti? (scriverli)

Ordine di distruzione

3)

Avete messo boid type come intero: cosa poteva essere? (enum)

Perché avete usato Int16\_t?

Sostituire for con gli indici con i vettori/algoritmi (rifarlo sul momento)

Quando i boids sono vicini avete drop di framerate: questo perché fate copie di vettori con centinaia di elementi ogni secondo

Perché avete usato una tuple? Come la potreste modificare? (una struct)

Come avreste potuto implementare in modo migliore la molla delle pareti? Se nel frame successivo vai a sbattere il punto dove va a sbattere (intersezione tra due rette). Si calcola il delta t per arrivare al punto, nel restante tempo si ribalta la velocità (tipo riflessione)

Alternativa: effetto pacman

1. Cosa vi aspettate che succeda se ci sono troppi boids? (riferito al fatto di aver usato Int16 per mettere un cap)
2. Riscrivere il codice usando vettori di classe base invece che di derivata
3. Slicing
4. Derivazione delle classi
5. Se uso solo l’interfaccia della classe base, ho bisogno di un vettore di puntatori a oggetti della derivata?
6. Le funzioni chiamabili solo dalle derivate di una classe dove vanno messe? Protected

4)

Perché avete messo un limite massimo alla velocità? Come potrebbe risolvere?

Allocazione dinamica della memoria

In root si usa new ma non si fanno le delete? Chi le fa? Come implementeresti la gestione della memoria su root?

Definizione di static, come si usa, come si invocano le funzioni static

New e delete: cosa succede se perdo il puntatore all’oggetto? (non lo posso più eliminare e ho un memory leak)

Che valore hanno le celle di memoria quando il sistema operativo te lo assegna (azzerate). Che valore hanno se fai new di un int, delete e poi di nuovo new? Potresti beccare il valore precedente

5) differenza tra pre e post incremento