

LAPORAN TUGAS AKHIR PEMROGRAMAN LANJUT
“NAZI PROJECT : TANK”



Dosen Pengampu:

Dr. Reza Fuad Rachmadi, S.T., M.T.

Dr. Eko Mulyanto Yuniarno, S.T., M.T

Disusun Oleh :

I Gusti Ngurah Agung Hari Vijaya Kusuma

07211940000073

DEPARTEMEN TEKNIK KOMPUTER
FAKULTAS TEKNOLOGI ELEKTRO INFORMATIKA CERDAS
INSTITUT TEKNOLOGI SEPULUH NOPEMBER

DAFTAR ISI

DAFTAR ISI

ABSTRAK

BAB I: PENDAHULUAN

1.1 Latar Belakang

1.2 Tujuan

1.3 Permasalahan

BAB II: TIPE DATA DAN VARIABEL

2.1 Pemrograman C

2.1.1 Sejarah Pemrograman C

2.1.2 Sejarah Pemrograman C++

2.1.3 Pendalaman Pemrograman C

2.1.3 Struktur Bahasa C

2.2 Pemrograman Dengan Oop

2.3 Vektor

2.4 Gerak Objek

BAB III: GAME

3.1 Penjelasan Objek Yang Dibuat

3.2 Penjelasan Pembuatan Gambar Objek

3.3 Penjelasan Pergerakan Objek

3.4 Penjelasan Score

3.4 Penjelasan Akhir

BAB IV: LISTING PROGRAM

4.1 Penjelasan Class

4.2 Penjelasan Pergerakan Objek Dalam Class

4.3 Penjelasan Metode Tumbukan

4.4 Penjelasan Akhir

BAB V: SCREENSHOT PROGRAM

BAB VI: KESIMPULAN

ABSTRAK

Game Nazi Project : Tank merupakan game yang bergenre war dan history, dengan kondisi kemenangan dimana para playernya harus bertarung untuk menjadi tank yang terbaik dengan membuktikan ke hitler bahwa mereka bisa mendapatkan score yang lebih tinggi dari hitler. Game ini merupakan Tugas Akhir dari mata kuliah pemrogramman lanjut di semester 2. adapun tujuan dari pembuatan game ini ialah mengasah kompetensi mahasiswa dalam pemrograman berbasis C++. Dalam pembuatannya game ini menggunakan Visual Studio sebagai IDE.

BAB I

PENDAHULUAN

1.1. Latar Belakang

Di era ini perkembangan Teknologi kian meningkat, hal ini ditandai dengan melesatnya kemampuan komponen komputer dalam menjalankan beragam aktifitas. salah satu aktivitas tersebut ialah bermain game. Jika kita melihat kebelakang, komputer dulunya cenderung memiliki ukuran yang sangat besar, dibandingkan dengan sekarang yang bahkan kita bisa pegang dengan tangan. selain lebih kuat dan efisien. komputer ini juga dilengkapi dengan alat kelengkapan lain seperti, pemutar audio, camera, dll. hal ini tentu mendorong para kreator di industri game untuk menuangkan karyanya di komputer jenis baru tersebut.

Dengan Perkembangan teknologi yang semakin canggih tersebut, adapun kompetensi yang harus dimiliki seorang mahasiswa jika ingin menguasai perkembangan hal tersebut. Oleh karena itu Semua Mahasiswa Departemen Teknik Komputer Institut Teknologi Sepuluh Nopember, diberikan wadah dalam mewujudkan hal tersebut. Wadah tersebut ialah Pembuatan Game Berbasis C++.

Dalam mewujudkan Game, Saya memilih genre *War*(perang) dalam memuwujudkannya. Game perang ini saya ambil tema dari perang dunia ke 2, dengan berbasis teknologi perang yang ada pada jaman itu. Tank merupakan kendaraan tempur lapis baja yang sangat berguna di era perang dunia ke 2. Teknologi Tank inilah yang menjadi inspirasi dalam pembuatan game ini.

Judul yang saya angkat dalam game ini ialah NAZI PROJECT : TANK , sesuai dengan latar perang yang saya angkat tadi, game ini mensimulasikan tes subject dari Nazi yaitu Tank yang dimana Tank ini akan di tes oleh Hitler sendiri dalam kesediaannya saat menghadapi perang. Hitler ingin Tank tersebut bertempur satu sama lain, namun hal tersebut tidak akan mudah karena hitler juga ingin tank tersebut menghindari rudal- rudal yang ada. Untuk memenangkan permainan player wajib mengalahkan score yang dimiliki Hitler. Game ini saya kembangkan menggunakan Visual Studio sebagai IDE, dengan GNU GCC sebagai compiler, C++ sebagai bahasa pemrograman dan tambahan SFML.

1.2.Tujuan

Tujuan dari pembuatan game ini ialah, meningkatkan kompetensi saya sebagai peserta didik dalam penggunaan bahasa C++, Mengenal dan memahami PBO (Pemrograman Beroerientasi Object), dan peningkatan kualitas diri dalam penggunaan SFML sebagai wadah untuk memahami pengembangan Game kedepannya.

1.3. Permasalahan

Dalam pembuatan game ini, adapun masalah yang saya temui seperti, file lib dari SFML yang dalam penginstalannya tidak bisa ditaruh di file D, pada pc saya. seringnya terjadi lag pada SFML saat menggunakan Class, selama 3 minggu saya menggunakan class dalam mengerjakan objek maupun hal hal lain. Namun setelah memasukan lebih banyak asset file kedalam Class, Class cenderung membuat FPS pada game yang saya buat menjadi Drop. sehingga pada penyelesaian game ini saya memutuskan untuk tidak menggunakan class untuk meningkatkan FPS pada game yang saya buat ini.

BAB II

DASAR TEORI

2.1. Pemrograman C++

2.1.1. Sejarah Bahasa C

Bahasa C dikembangkan di Bell Telephone Laboratories pada tahun 1972 ditulis pertama kali oleh Dennis Ritchie, kemudian dikembangkan oleh Dennis Ritchie dan Brian W. Kernighan, bahasa ini merupakan bahasa pengembangan / turunan dari bahasa B yang ditulis oleh Ken Thompson pada tahun 1970 yang diturunkan oleh bahasa sebelumnya, yaitu BCL. Bahasa C, pada awalnya dirancang sebagai bahasa pemrograman yang dioperasikan pada sistem operasi UNIX. Bahasa C merupakan bahasa pemrograman tingkat menengah yaitu diantara bahasa tingkat rendah dan tingkat tinggi yang biasa disebut dengan Bahasa Tingkat Menengah.

Meskipun C dibuat untuk memprogram sistem dan jaringan komputer namun bahasa ini mempunyai banyak kemampuan salah satunya sering digunakan dalam mengembangkan software aplikasi misalnya Word Star, dBASE dan lain-lain. Bahasa C juga banyak dipakai oleh berbagai jenis platform sistem operasi dan arsitektur komputer. Bahasa C memiliki pengaruh yang besar pada perkembangan bahasa populer lainnya, terutama C++ yang merupakan ekstensi dari Bahasa C.

C++ semula disebut sebagai "C dengan Kelas" (C With Classes) dan diciptakan untuk mempunyai fitur pemrograman berorientasi objek. Karena C++ berdasarkan dari C, maka kebanyakan kode C bisa dirakit di compiler C++ dengan mudah. Perbedaan kecil antara C dan C++ contohnya kata "new" dan "delete" yang terdapat di kode C tidak bisa dirakit di C++ karena kata-kata ini adalah kata yang hanya ada di C++. Pustaka C biasanya bisa diimpor ke pustaka C++, tapi karena kompilator C dan C++ memiliki "name mangling" yang berbeda, maka perlu dilakukan perubahan kecil di kode C.

2.1.2. Sejarah Bahasa C++

Pada tahun 1980 Bahasa C++ diciptakan oleh Bjarne Stroustrup, Awalnya prototype C++ muncul sebagai C yang diperancang dengan fasilitas kelas. Bahasa tersebut disebut C With Classes (C dengan kelas) dan diciptakan untuk memiliki fitur pemrograman berorientasi objek.

Pada tahun 1983-1984, C dengan kelas (C With Classes) disempurnakan dengan menambahkan fasilitas pembebanan lebih operator dan fungsi. yang awalnya C++ disebut "a better C" kemudian berganti nama pada tahun 1983 menjadi C++. Symbol ++ merupakan operator C untuk operasi penaikan, muncul untuk menunjukkan bahwa bahasa baru ini merupakan versi yang lebih canggih dari C. Borland International kemudian merilis compiler Borland C++ dan Turbo C++. Dua buah compiler ini bisa digunakan untuk mengkompilasi kode C++. Bedanya, Borland C++ selain mampu digunakan dibawah lingkungan DOS, juga bisa digunakan untuk pemrograman Windows. Selain Borland International, beberapa perusahaan lain juga merilis compiler C++, seperti GNU Compiler Collection, Topspeed C++ dan Zortech C++.

Untuk cerita lengkap tentang penemuan bahasa C++, seperti sebelumnya diceritakan diatas C++ adalah bahasa pemrograman yang diciptakan pada tahun 1980, ketika Bjarne Stroustrup melakukan pekerjaan untuk memperoleh gelar Ph.D. Saat itu Bjarne Stroustrup memiliki kesempatan untuk bekerja dengan bahasa Simula, yang seperti namanya bahasa tersebut merupakan bahasa untuk simulasi. Bahasa simula juga dianggap sebagai bahasa pertama untuk mendukung paradigma pemrograman berorientasi objek. Bjarne Stroustrup menemukan bahwa paradigma ini sangat berguna untuk pengembangan perangkat lunak, namun bahasa Simula terlalu lambat untuk penggunaan praktis.

Tak lama kemudian, ia mulai bekerja pada "C with classes" atau di dalam bahasa indonesia adalah C dengan Kelas, Petama Compiler C dengan Kelas disebut Cfront, yang berasal dari kompiler C yang disebut CPRE. Hal itu merupakan program yang dirancang untuk menerjemahkan C dengan Kelas ke C. perlu diperhatikan bahwa Cfront sebagian besar ditulis dalam C dengan Kelas, membuatnya menjadi compiler self hosting (compiler yang dapat mengkompilasi sendiri). Cfront kemudian ditinggalkan pada tahun 1993 setelah mengalami kesulitan dalam mengintegrasikan fitur baru di dalamnya. Meskipun demikian, Cfront membuat dampak besar pada implementasi kompiler masa depan terutama pada sistem operasi Unix.

Pada tahun 1983, terjadi perubahan nama dari C dengan Kelas menjadi C++. makna dari ++ dalam bahasa C karena ++ merupakan operator untuk increment variabel, yaitu proses penambahan pada nilai variabel sebanyak 1. Dengan demikian C++ berarti C+1, nilai 1 disini melambangkan dukungan terhadap pemrograman berorientasi objek. demngan demikian C++ merupakan bahasa C yang ditambah dengan kemampuan atau dukungan terhadap pemrograman berorientasi objek. Sebab semua yang dapat kita lakukan dalam Bahasa C pasti bisa dilakukan

didalam C++, namun hal tersebut tidak berlaku sebaliknya. Bebeapa fitur baru yang ditambahkan dalam C++ anantara lain fungsi virtual, fungsi overloading, reference dengan simbol "&", kata kunci "const", dan komentar pada satu baris tertentu menggunakan dua garis miring ke depan "///" (merupakan fitur yang diambil dari bahasa BCPL).

Pada tahun 1985, reference yang dikemukakan oleh Bjarne Stroustrup untuk bahasa berjudul "The C++ Programming Language" diterbitkan. Pada tahun yang sama, C++ digunakan sebagai produk komersial. Pada saat itu C++ belum secara resmi distandarkan. Kemudian C++ diperbarui lagi pada tahun 1989 untuk memasukkan protected dan static member, serta inheritance dari beberapa kelas.

Pada tahun 1990, The Annotated C++ Reference Manual dirilis. Pada tahun yang sama, Compiler Borland Turbo C++ dirilis sebagai produk komersial. Turbo C++ menambahkan banyak library tambahan yang memiliki dampak besar terhadap pengembangan C++. Meskipun rilis stabil terbaru dari Turbo C++ terjadi pada tahun 2006, namun compiler ini masih banyak digunakan.

Pada tahun 1998, Diterbitkan standar internasional pertama untuk C++ ISO / IEC 14882:1998, Yang secara informal dikenal sebagai C++98. dalam standar yang diterbitkan The Annotated C++ Reference Manual dikatakan memiliki pengaruh besar dalam pengembangan standar tersebut. Pada tahun 2003, komite standar C++ merespon beberapa masalah yang dilaporkan dengan standar mereka yang diterbitkan pada tahun 1998, dan merevisinya. kemudian Bahasa C++ yang direvisi dijuluki Bahasa C++03.

Pada tahun 2005, komite standar C++ merilis laporan teknis (dijuluki TR1) merinci berbagai fitur yang mereka rencanakan untuk menambah fitur yang ada di C++ standar terbaru. Standar baru itu secara informal dijuluki C++0x, standar terbaru tersebut diharap akan dirilis sebelum akhir dekade pertama. Namun, ironisnya, standar baru tersebut belum dirilis sampai pertengahan 2011.

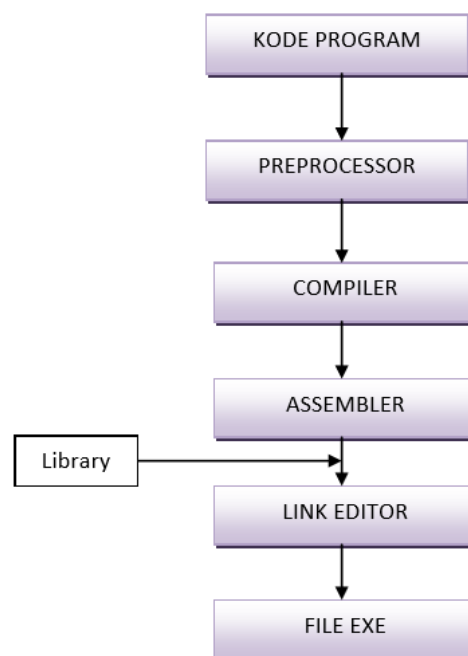
Pada pertengahan 2011, C++ dengan standar baru (dijuluki C++11) selesai dibuat (disetujui oleh ISO/IEC pada 12 Agustus 2011, diterbitkan sebagai 14882:11). Standar ini meningkatkan Library yang ada dalam C++, sehingga standar yang baru dikeluarkan membuat dampak (perubahan) yang besar pada standar C++.

2.2. Pengenalan C++

Untuk memulai bahasa C++ dapat menggunakan text editor apapun, sekalipun menggunakan notepad. Hal ini dimungkinkan karena sumber kode bahasa pemrograman ini disimpan pada format file plaintext, yang artinya isi dari file sumber kode berisi text apa adanya tanpa penyesuaian format binary. Namun untuk

Modul Algoritma dan Pemrograman 1 (C++) - Agung Sasongko | AMIK “BSI Pontianak” – Halaman 5

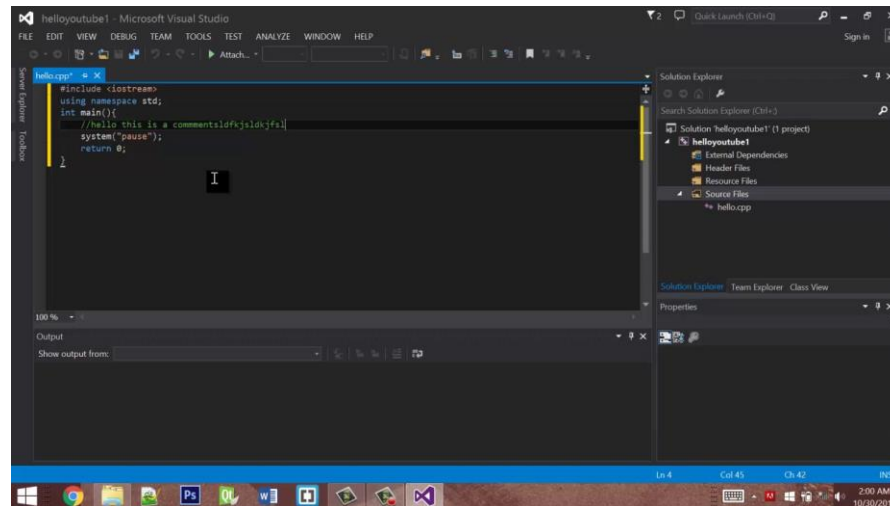
menjalankan perintah kode bahasa C++ diperlukan compiler, yaitu program yang menterjemahkan bahasa C++ dalam bentuk plaintext menjadi binary yang dapat dikenali oleh komputer. Berikut skema gambaran bagaimana kode program dari plaintext kemudian dapat menjadi dikenali sebagai instruksi kepada komputer.



Gambar 1.1. Urutan Kompilasi C++

Dari skema diatas dapat dilihat bahwa kode program yang telah ditulis disesuaikan dengan perintah-perintah menjadi preprocessor yang kemudian dilakukan penterjemahan oleh compiler

menjadi bahasa assembler. Pada bagian Assembler akan membuat file objek yang selanjutnya melihat perintah dengan adanya hubungan dengan library lain yang kemudian dibentuk file eksekusi (.executeable).



IDE (Interface Development Enviroment) diatas dapat dilihat beberapa bagian pada lingkungan pengembangan program.

1. Top Menu Menu dropdown yang terdapat semua menu yang dibutuhkan pada aplikasi IDE.
2. Menu Bar Bagian menu yang bersifat jalan pintas berbentuk gambar. Untuk membuat file baru, menjalankan program tersedia pada bagian ini.
3. Editor Bagian tempat menuliskan kode program.

2.3. Memulai Pengkodean Program

Untuk memulai pengkodean program pada bahasa C++, perlu diketahui terlebih dahulu struktur dasar penulisan instruksi.

Bentuk Dasar:

```
[PREPROCESSOR]

main() //entry poin
{
    [INSTRUKSI C++]
}
```

Contoh:

```
#include <iostream.h>

main() //entry poin
{
    cout<<"Hai... belajar C++ nih";
}
```

2.4. Type Data

Model data yang dikenal oleh C++ dapat dibagi menjadi dua (2) bagian, yaitu:

1. Data Huruf / Karakter

- Model data ini untuk segala jenis data yang tidak memiliki kepentingan untuk dilakukannya perhitungan matematis
- Contoh jenis data: nama, alamat, tempat tinggal, no telepon dan lain sebagainya
- Tipe data Huruf / karakter di C dikenal dengan istilah : char
- Pada dasarnya tipe data char ini hanya untuk menyimpan sebuah nilai huruf/karakter saja
- Format %c untuk karakter tunggal, dan %s untuk karakter jamak.

2. Data Angka

- Model data ini untuk segala jenis data yang memiliki kepentingan untuk dilakukannya operasi aritmatika
- Contoh: gaji, total harga, umur, panjang, lebar, nilai dan lain sebagainya
- Tipe data dari model data angka bilangan bulat adalah: 1) Integer di bahasa C : int (-32k s/d 32k). Format : %d 2) Short di bahasa C : short (-32k s/d 32k). Format : %d 3) Long di bahasa C : long (-2g s/d 2g). Format : %ld
- Tipe data dari model data angka bilangan pecahan adalah: 1) Float di bahasa C : float (3.4×10^{-38} s/d 3.4×10^{38}). Format : %f 2) Double di bahasa C : double (1.7×10^{-308}

s/d 1.7×10^{308}). Format : %f 3) Long Double di bahasa C : long double (3.4×10^{-4932} s/d 3.4×10^{4932}) Format : %f

- e. Unsigned digunakan pada pengenalan awal tipe data untuk menghilangkan nilai negatif dan dialihkan nilainya menjadi nilai positif

Contoh: jika int jangkauan range nilai yang dapat ditampung (-32k s/d 32k) maka bila unsigned int jangkauan range nilai menjadi (0k s/d 64k)

2.5. Perintah Masukkan (Input)

Perintah input adalah perintah masukkan yang harus digunakan apabila program membutuhkan nilai atau data masukkan dari pengguna program. Perintah input yang dapat digunakan pada bahasa pemrograman C++ adalah.

1. scanf()

Modul Algoritma dan Pemrograman 1 (C++) - Agung Sasongko | AMIK “BSI Pontianak” – Halaman 8

Perintah masukkan sesuai format yang ditentukan. Perintah ini membutuhkan processor `stdio.h`. Contoh penggunaan : `scanf(“%s”, &nama);`

2. gets()

Perintah masukan huruf yang lebih dari satu. Perintah ini memungkinkan pengguna memasukkan text dengan spasi. Berbeda dengan perintah `scanf` yang bila diberikan spasi data harus dibaca lebih dari satu variable. Perintah `gets` membutuhkan include `stdio.h`.

Contoh penggunaan: `char nama[20]; gets(nama);`

3. getch()

Perintah masukkan untuk satu karakter. Nilai karakter yang dibaca menjadi nilai balik pada pemanggilan fungsi ini. Biasanya perintah `getch` digunakan untuk menahan tampilan program C++ yang berbasis command line yang berjalan diatas Windows. Karena pada, bila tidak ada instruksi yang harus dijalankan lagi, maka program akan keluar dari memori. Perintah `getch` membutuhkan include `stdio.h`. Contoh penggunaan biasanya diletakkan pada akhir baris perintah program sebelum tanda penutup badan program.

4. cout

Perintah masukkan hampir mirip seperti perintah scanf, namun tidak perlu format tipe data penerima. Perintah ini membutuhkan include iostream.h dan hanya ada pada C++ tidak dikenal pada bahasa C yang biasa. Contoh penggunaan: `char nama[20];`
`cout<<nama;`

2.6. Perintah Keluaran (Output)

Perintah output atau keluaran merupakan perintah yang digunakan untuk menghasilkan text yang tampil di layar. Biasanya tampilan ini untuk menampilkan instruksional cara penggunaan program, notifikasi, meminta masukan data maupun hasil pengolahan data. Perintah-perintah keluaran yang dikenal yaitu:

1. printf

Perintah ini adalah perintah keluaran yang menggunakan format tipe data. Perintah keluaran ini memungkinkan gabungan antara string konstan dengan string variable. Untuk menggunakan printf harus menggunakan preprocessor stdio.h. Contoh: `printf("Hello saya belajar c++");`

2. puts

Perintah ini merupakan perintah keluaran yang tidak dapat digabungkan antara string konstan dengan string variable, yang berarti harus menggunakan salah satunya. Membutuhkan preprocessor stdio.h. Contoh: `puts("IGustiNgurahAgung Ganteng")`

3. putchar

Perintah keluaran untuk sebuah nilai karakter saja. Membutuhkan preprocessor stdio.h. Contoh: `putchar(„B“); putchar(„S“); putchar(„I“);`

4. cout

Perintah keluaran ini mirip scanf, yaitu perintah keluaran yang dapat mengkombinasikan string konstan dengan string variable. Namun untuk menggunakan perintah cout harus menggunakan preprocessor iostream.h

2.7. Operator Aritmatika

Operator aritmatika adalah operator yang digunakan untuk operasi perhitungan pada variable yang dikenal pada bahasa pemrograman. Operator yang dikenal adalah

Tabel 2.1. Daftar Operator Aritmatika

No	Operator	Symbol	Kegunaan
1.	Pengurangan	-	Mengurangi nilai
2.	Penjumlahan	+	Menambahkan nilai
3.	Perkalian	*	Mengalikan nilai
4.	Pembagian	/	Membagi nilai
5.	Modulo	%	Mendapatkan sisa pembagian. Contoh: 5 % 3 menghasilkan 2. 10 % 2 menghasilkan 0 11 % 4 menghasilkan 3

2.8. Kondisi

Kondisi pada algoritma hanya mengenal keadaan benar atau salah, true atau false, 1 atau 0 atau yang juga dikenal dengan istilah boolean. Operator yang menghasilkan keadaan nilai Boolean yaitu operator pembandingan.

No	Operator	Symbol	Contoh
1.	Lebih besar	>	10 > 2 menghasilkan nilai benar 10 > 3+5 menghasilkan nilai benar 10 > 10 menghasilkan nilai salah 10 > 5+5 menghasilkan nilai salah
2.	Lebih kecil	<	10 < 2 menghasilkan nilai salah 10 < 100 menghasilkan nilai benar
3.	Lebih besar sama dengan	>=	10 >= 10 menghasilkan nilai benar 10 >= 9 menghasilkan nilai salah
4.	Lebih kecil sama dengan	<=	10 <= 10 menghasilkan nilai benar 10 <= 9 menghasilkan nilai salah
5.	sama dengan	==	29 == 29 menghasilkan nilai benar 29 == 30 menghasilkan nilai benar
6.	Tidak sama dengan	!=	29 != 30 menghasilkan nilai benar 29 != 29 menghasilkan nilai salah

2.9. Operator Logika

Operator logika adalah operator yang membandingkan dua kondisi untuk menghasilkan nilai benar atau salah, 1 atau 0. Biasanya operator logika disandingkan dengan operator pembandingan

No.	Operator	Symbol	Keterangan
1	AND	&&	1 && 0 menghasilkan 0 0 && 1 menghasilkan 0 0 && 0 menghasilkan 0 1 && 1 menghasilkan 1
2	OR		1 && 0 menghasilkan 1 0 && 1 menghasilkan 1 0 && 0 menghasilkan 0 1 && 1 menghasilkan 1

2.10. Kontrol Percabangan IF

Kontrol percabangan merupakan perintah pada algoritma yang akan menentukan kode program mana yang akan dijalankan dari berbagai kode program yang dibuat pada rangkaian scenario alir program. Pada dasarnya setiap perintah percabangan menggunakan suatu kondisi sebagai penentu apakah bagian kode program akan dijalankan atau tidak. Struktur dasar

```
If(kondisi)
{
    Pernyataan 1;
}

If(kondisi)
{
    Pernyataan 1;
}else{
    Pernyataan 2
}
```

2.11. Kontrol Percabangan NESTED IF

Kontrol percabangan memungkinkan pengendalian percabangan didalamnya atau dengan istilah percabangan didalam percabangan (nested loop).

Struktur dasar:

```

if (kondisi)
{
    if(kondisi2)
    {
        Pernyataan;
    }
}

```

2.12. Kontrol Percabangan SWITCH

Kontrol percabangan switch merupakan bentuk percabangan yang melihat nilai variable secara konstan.

Struktur dasar:

```

switch(variable)
{
    case nilai_konstan_1:
        pernyataan;
        break;
    case nilai_konstan_2:
        pernyataan;
    case nilai_konstan_n:
        pernyataan;
    default:
        pernyataan;
}

```

Keterangan:

1. Variable = variable yang akan di uji nilainya sebagai penentu pernyataan yang akan dijalankan

2. Nilai_konstan = nilai yang menjadi penentu dengan variable yang akan diuji. Bila nilainya sama maka akan menjalankan pernyataan pada baris perintah dibawah case ini hnya.
3. Break = perintah untuk menahan perintah agar tidak menjalankan perintah yang ada dibawah. Bila break tidak dibuat akan menjadikan suatu case yang terpilih menjalankan case yang ada dibawahnya.

2.13. Perulangan Perulangan

merupakan perintah pemrograman untuk mengulang alir program sebanyak kondisi yang ditentukan. Perintah ini akan sangat sering digunakan pada program yang mengharuskan pembacaan data yang jumlahnya banyak, menampilkan data, pencarian, pengurutan data, penggambaran dan lain sebagainya.

Struktur dasar:

```
for( kondisi awal; kondisi berhenti; peubah kondisi)
{
    pernyataan;
}
```

2.14. Kontrol Perulangan WHILE

Struktur WHILE biasanya digunakan untuk perulangan yang jumlah perulangannya belum diketahui. Namun hal seperti ini tidap menutup kemungkinan juga dapat ditangani oleh perulangan FOR.

Struktur Dasar:

```
while(kondisi)
{
    Pernyataan;
}
```

Perintah while akan selalu mengulang pernyataan didalamnya apabila kondisi selalu berstatus true

2.15. Kontrol Perulangan DO WHILE

Kontrol perulangan do..while merupakan bentuk perulangan yang melakukan pengecekan kondisi setelah melakukan badan perintah perulangan.

Struktur dasar.

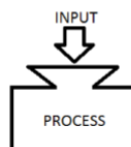
```
do{  
    pernyataan;  
}while(kondisi);
```

1. Program contoh do while

```
#include <iostream.h>  
#include <conio.h>  
  
main()  
{  
    int ulang=1;  
  
    do  
    {  
        cout<<"tekan 3 untuk keluar : ";  
        cin>>ulang;  
    }while (ulang!=3);  
    getch();  
}
```

2.16. Procedure / Sub Routine

Procedure atau subroutine merupakan blok rangkaian kode program yang mengerjakan sesuatu sebagai bagian pembentuk suatu program yang tidak memberikan hasil nilai kembalian pada pemanggilnya. Gambaran Subroutine dapat ditunjukkan sebagai berikut:



Gambar 5.1. Ilustrasi Procedure / Subroutine

Struktur dasar:

```
void namasubroutine([parameter])  
{  
    [Perintah-perintah / blok kode program];  
}
```

Biasanya procedure dibuat untuk memudahkan pengelolaan suatu program. Karena program yang dibuat secara kompleks memiliki banyak baris perintah, dengan adanya procedure maka akan membagi blok-blok perintah menjadi bagian-bagian tersendiri.

2.17. Fungsi

Fungsi merupakan blok kode program yang mengerjakan sesuatu untuk menghasilkan suatu nilai yang akan diterima secara assignment pada pemanggilan nya. Ilustrasi fungsi dapat ditunjukkan sebagai berikut:



2.18. Fungsi String

Fungsi string yang telah tersedia pada bahasa C yaitu sebagai berikut:

1. strlen

Fungsi untuk mengetahui jumlah teks pada data karakter. Parameter nya adalah variable yang mengandung teks, nilai baliknya adalah angka bernilai panjang teksnya.

Fungsi ini membutuhkan preprocessor string.h, namun juga tersedia pada iostream.h.

2. strcpy

Fungsi ini digunakan untuk menyalin nilai text pada suatu variable ke variable lain.

Fungsi ini membutuhkan preprocessor string.h, namun juga tersedia pada iostream.h.

3. strcat

Fungsi ini digunakan untuk menggabungkan nilai text suatu variable maupun string konstan dengan nilai text lainnya. Seperti fungsi-fungsi string lainnya, fungsi ini tersedia pada string.h, namun juga tersedia pada iostream.h

4. strcmp

Fungsi ini digunakan untuk membandingkan nilai text. Kesamaan text akan memberikan nilai balik 0, namun bila ada perbedaan maka fungsi ini akan menghasilkan jarak karakter yang berbeda.

5.strupr

Fungsi ini digunakan untuk mengganti nilai text menjadi huruf capital. Preprocessor string.h maupun iostream.h.

6. strlwr

Fungsi ini digunakan untuk merubah nilai teks menjadi huruf kecil. Preprocessor string.h maupun iostream.h.

7. itoa

Fungsi ini digunakan untuk merubah nilai tipe data angka bilangan bulat (int, long) menjadi tipe data karakter (char). Parameter fungsi ini adalah (variable angka, variable karakter, radix). Preprocessor stdlib.h

8. atoi

Fungsi atoi adalah kebalikan dari itoa. Preprocessor stdlib.h

2.19. Fungsi Matematika

Fungsi matematika yang telah tersedia pada bahasa C++ berada pada preprocessor math.h.diantaranya:

1. pow

Fungsi untuk menghitung nilai pangkat.

2. sqrt

Fungsi ini untuk menghitung akar kuadrat.

3. sin

2.20. Array Satu Dimensi

Array adalah tipe data yang terdiri dari sejumlah komponen elemen yang berjenis sama.

```
tipeData namaVariabel[jumlahElemen];
```

Contoh penerapan:

```
char x[8];
```

Deklarasi diatas akan menghasilkan 8 elemen dengan bentuk sebagai berikut:



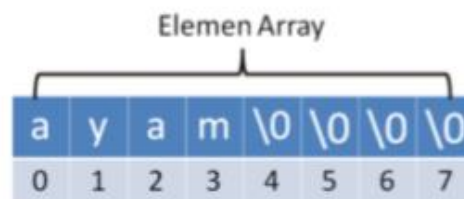
Gambar 8.1 Ilustrasi elemen array

Contoh deklarasi pembuatan variable array satu dimensi:

```
char nama[20];  
double daftarGaji[100];  
float y[10];  
long dfthrg[100];
```

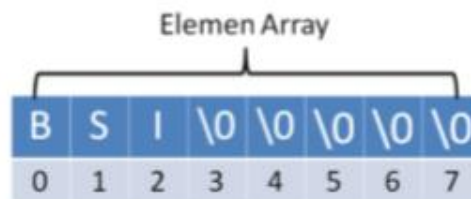
1. Pengisian array teks cara pertama

```
char x[8] = ('a','y','a','m');
```



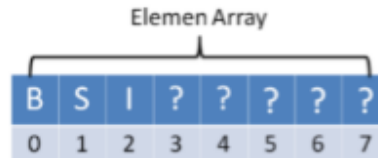
2. Pengisian array teks cara kedua

```
char x[8] = "BSI"
```



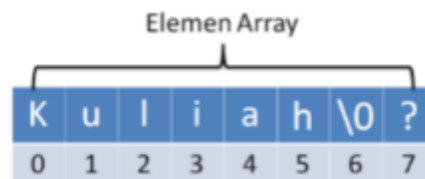
3. Pengisian array teks per elemen

```
char x[8];  
x[0] = 'B';  
x[1] = 'S';  
x[2] = 'I';
```



4. Pengisian array teks menggunakan strcpy

```
char x[8];  
strcpy(x, "Kuliah");
```



5. Pengisian array angka

```
int x[8] = {2, 5, 7, 9, 10, 14};
```

Elemen Array

2	5	7	9	10	14	0	0
0	1	2	3	4	5	6	7

6. Pengisian array angka per-elemen

```
int x[5];  
x[0] = 9;  
x[4] = 10;  
  
float t[3];  
t[1] = 9.5;  
t[2] = 2.12;
```

9	?	?	?	10
0	1	2	3	4

?	9.5	2.12
0	1	2

2.21. Array Multi Dimensi

Array multidimensi adalah sekumpulan data yang dapat dianalogikan sebagai matrix, yang terdiri dari baris dan kolom.

Struktur dasar:

```
int data[4][3];
```

Variabel diatas dapat diilustrasikan berupa elemen sebagai berikut:

	0	1	2
0	?	?	?
1	?	?	?
2	?	?	?
3	?	?	?

Contoh deklarasi variable array multidimensi:

```
int    data[4][3];  
char   dft[5][20];  
float  x[100][200];
```

2.22. Struct

Struct adalah statement yang berguna untuk mengelompokkan sejumlah data dengan tipe yang berlainan.

Struktur dasar:

```
struct nama_struktur
{
    tipe_data var_anggota_struktur;
};
```

Contoh pendeklarasian struktur:

```
struct Mahasiswa
{
    char  nim[9];
    char  nama[20];
    int   thnmasuk;
};
```

2.23. Pointer

Merupakan variabel yang dapat digunakan menunjuk ke alamat memori suatu variable.

Penggunaan pointer biasanya untuk mengendalikan suatu nilai variable pada blok kode program.

Struktur dasar:

```
tipeData *namaVariabel;
```

Contoh pembuatan variable pointer:

```
char *nama;
int *a;
float *x;
```

2.24. Pointer Sebagai Pengendali Variabel

Agar pointer dapat mengendalikan variable, pointer harus memiliki informasi alamat dari variable yang akan dikendalikan nilainya.

1. Reference (&) Reference adalah symbol yang digunakan untuk mendapatkan alamat dari suatu variable. Berikut percobaan yang membuktikan bahwa reference memberikan informasi alamat memori variable.

2.25. Pointer Pengendali variable Array

Bentuk reference pada variable array tidak menggunakan symbol (&). Namun cukup menuliskan nama variable array tanpa symbol bracket ([]) sebagai alamat elemen.

2.26. Pointer Sebagai Penyimpan Data

Selain digunakan untuk mengendalikan variable, pointer dapat juga untuk mengalokasikan memori untuk penyimpanan tanpa menunjuk ke variable lain.

Struktur dasar:

```
tipeData* namaVariabel = new tipeData;
```

2.2 Pemrograman dengan OOP

Pemrograman berorientasi objek ([Inggris](#): *object-oriented programming* disingkat OOP) merupakan [paradigma pemrograman](#) berdasarkan konsep "objek", yang dapat berisi [data](#), dalam bentuk *field* atau dikenal juga sebagai atribut; serta kode, dalam bentuk fungsi/prosedur atau dikenal juga sebagai *method*. Semua data dan fungsi di dalam paradigma ini dibungkus dalam *kelas-kelas* atau *objek-objek*. Bandingkan dengan logika [pemrograman terstruktur](#). Setiap objek dapat menerima [pesan](#), memproses data, dan mengirim pesan ke objek lainnya,

Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam [teknik peranti lunak](#) skala besar. Lebih jauh lagi, pendukung OOP mengklaim bahwa OOP lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya, dan pendekatan OOP lebih mudah dikembangkan dan dirawat.

Konsep dasar

- a. [Kelas](#) — kumpulan atas definisi data dan fungsi-fungsi dalam suatu unit untuk suatu tujuan tertentu. Sebagai contoh 'class of dog' adalah suatu unit yang terdiri atas definisi-definisi data dan fungsi-fungsi yang menunjuk pada berbagai macam perilaku/turunan dari anjing. Sebuah class adalah dasar dari modularitas dan struktur dalam pemrograman berorientasi object. *Sebuah class secara tipikal sebaiknya dapat dikenali oleh seorang non-*

programmer sekalipun terkait dengan domain permasalahan yang ada, dan kode yang terdapat dalam sebuah class sebaiknya (relatif) bersifat mandiri dan independen (sebagaimana kode tersebut digunakan jika tidak menggunakan OOP). Dengan modularitas, struktur dari sebuah program akan terkait dengan aspek-aspek dalam masalah yang akan diselesaikan melalui program tersebut. Cara seperti ini akan menyederhanakan pemetaan dari masalah ke sebuah program ataupun sebaliknya.

- b. **Objek** - membungkus data dan fungsi bersama menjadi suatu unit dalam sebuah **program komputer**; **objek** merupakan dasar dari **modularitas** dan **struktur** dalam sebuah program komputer berorientasi objek.
- c. **Abstraksi** - Kemampuan sebuah program untuk melewati aspek informasi yang diproses olehnya, yaitu kemampuan untuk memfokus pada inti. Setiap objek dalam sistem melayani sebagai model dari "pelaku" abstrak yang dapat melakukan kerja, laporan dan perubahan keadaannya, dan berkomunikasi dengan objek lainnya dalam sistem, tanpa mengungkapkan bagaimana kelebihan ini diterapkan. Proses, fungsi atau metode dapat juga dibuat abstrak, dan beberapa teknik digunakan untuk mengembangkan sebuah pengabstrakan.
- d. **Enkapsulasi** - Memastikan pengguna sebuah objek tidak dapat mengganti keadaan dalam dari sebuah objek dengan cara yang tidak layak; hanya metode dalam objek tersebut yang diberi izin untuk mengakses keadaannya. Setiap objek mengakses **interface** yang menyebutkan bagaimana objek lainnya dapat berinteraksi dengannya. Objek lainnya tidak akan mengetahui dan tergantung kepada representasi dalam objek tersebut.
- e. **Polimorfisme** melalui pengiriman pesan. Tidak bergantung kepada pemanggilan subrutin, bahasa orientasi objek dapat mengirim pesan; metode tertentu yang berhubungan dengan sebuah pengiriman pesan tergantung kepada objek tertentu di mana pesa tersebut dikirim. Contohnya, bila sebuah burung menerima pesan "gerak cepat", dia akan menggerakkan sayapnya dan terbang. Bila seekor singa menerima pesan yang sama, dia akan menggerakkan kakinya dan berlari. Keduanya menjawab sebuah pesan yang sama, namun yang sesuai dengan kemampuan hewan tersebut. Ini disebut polimorfisme karena sebuah variabel tunggal dalam program dapat memegang berbagai jenis objek yang berbeda selagi program berjalan, dan teks program yang sama dapat memanggil beberapa metode yang

berbeda di saat yang berbeda dalam pemanggilan yang sama. Hal ini berlawanan dengan [bahasa fungsional](#) yang mencapai polimorfisme melalui penggunaan fungsi kelas-pertama.

- f. Dengan menggunakan OOP maka dalam melakukan pemecahan suatu masalah kita tidak melihat bagaimana cara menyelesaikan suatu masalah tersebut (terstruktur) tetapi objek-objek apa yang dapat melakukan pemecahan masalah tersebut. Sebagai contoh anggap kita memiliki sebuah departemen yang memiliki manager, sekretaris, petugas administrasi data dan lainnya. Misal manager tersebut ingin memperoleh data dari bag administrasi maka manager tersebut tidak harus mengambilnya langsung tetapi dapat menyuruh petugas bag administrasi untuk mengambilnya. Pada kasus tersebut seorang manager tidak harus mengetahui bagaimana cara mengambil data tersebut tetapi manager bisa mendapatkan data tersebut melalui objek petugas administrasi. Jadi untuk menyelesaikan suatu masalah dengan kolaborasi antar objek-objek yang ada karena setiap objek memiliki deskripsi tugasnya sendiri.

[Bahasa pemrograman](#) yang mendukung OOP antara lain:

1. [Visual Foxpro](#)
2. [Java](#)
3. [C++](#)
4. [Pascal \(bahasa pemrograman\)](#)
5. [SIMULA](#)
6. [Smalltalk](#)
7. [Ruby](#)
8. [Python](#)
9. [PHP](#)
10. [C#](#)
11. [Delphi](#)
12. [Eiffel](#)
13. [Perl](#)
14. [Adobe Flash AS 3.0](#)

2.3 Vektor

Vector adalah suatu class template yang merupakan bagian dari STL (*Standard Template Library*) dan dapat digunakan untuk menggantikan array. Sama dengan array, vector juga menyimpan elemen-elemen secara bersebelahan dan elemen tersebut dapat diakses sesuai subscript/index.

Untuk menggunakan vector pada C++ kita perlu menyertakan header `<vector>`.

Bentuk pendeklarasian vector:

```
vector <type> nama_variabel (jumlah_elemen);
```

**jumlah elemen bersifat opsional*

Vektor sama dengan array dinamis dengan kemampuan untuk mengubah ukurannya sendiri secara otomatis ketika elemen dimasukkan atau dihapus, dengan penyimpanannya ditangani secara otomatis oleh wadah. Elemen vektor ditempatkan dalam penyimpanan yang berdekatan sehingga mereka dapat diakses dan dilalui menggunakan iterator. Dalam vektor, data dimasukkan di bagian akhir. Memasukkan pada akhirnya membutuhkan waktu yang berbeda, karena terkadang ada kebutuhan untuk memperpanjang array. Menghapus elemen terakhir hanya membutuhkan waktu konstan karena tidak ada perubahan ukuran yang terjadi. Memasukkan dan menghapus di awal atau di tengah adalah linear dalam waktu.

2.4 Gerak Objek

Tujuan utama dari pemrograman C++ ialah untuk menambahkan orientasi objek pada bahasa pemrograman C dan kelas-kelas yang dijadikan sebagai fitur dari C++ yang mendukung pemrograman berorientasi objek dan sering juga dikenal sebagai *user-defined type*.

Pengenalan Class

Class merupakan *blueprint* (cetak biru) untuk menciptakan suatu *instance* dari objek dimana terdiri dari sekumpulan objek dengan kemiripan data / *properties* / *attributes*, fungsi / *behavior* / *method* dan relasi ke objek lain. Pemrograman C++ memungkinkan pembuatan *class* lebih dari 1. Ketika data dan fungsi yang terkait disimpan di dalam sebuah *class* mampu membantu mem-visualisasikan permasalahan yang kompleks dengan efisien dan efektif.

Contoh: *ClassMahasiswa*, *ClassDosen*, *Class Flowers*.

Class
data 1 data 2 ... data n
fungsi 1 fungsi 2 ... fungsi n

Berdasarkan Gambar diatas, data dan fungsi yang berada di dalam sebuah *class* disebut sebagai anggota dari suatu *class*. Data pada suatu *class* digunakan untuk memegang informasi yang ada pada *class* tersebut, sedangkan fungsi digunakan sebagai *behavior* dari *class* tersebut.

Untuk pembuatan sebuah *class*, dimulai dengan kata kunci ***class*** dan diikuti dengan **nama** kelasnya, dibuka dengan {, isi dari *class*, ditutup dengan};. Berikut adalah sintaks pembuatan *class*:

```
Class class_name
{
    Data Members;
    Methods;
};
```

Contoh:

```
class Box{
public:
    double length; // Panjang box
    double width;  // Lebar box
    double height; // Tinggi box
};
```

Seperti yang telah disebutkan di atas, pendefinisian suatu kelas dimulai dengan kata kunci ***class*** dan diikuti dengan nama kelas **Box** dalam kasus ini. Isi dari suatu kelas ditandai dengan { dan diakhiri dengan } diikuti;. Kata kunci **public** pada contoh di atas menentukan cara

pengaksesan anggota kelas. Anggota kelas **public** dapat diakses di kelas manapun. Terdapat beberapa jenis pengaksesan anggota kelas lainnya (access specifier) yang akan dibahas pada bab *Encapsulation*.

Member Class

Seperti yang telah disinggung sebelumnya bahwa data dan fungsi di dalam suatu *class* disebut dengan anggota suatu *class*. Perhatikan contoh berikut ini:

```
classBox{  
    public:  
    double length; // Length of a box  
    double breadth; // Breadth of a box  
    double height; // Height of a box  
  
    void print()  
    {  
        cout<<"Printing Box Object"<<endl;  
    }  
};
```

Berdasarkan contoh di atas, **length**, **breadth** dan **height** merupakan data member dari *class* **Box**; sedangkan **print** merupakan *function member* (member fungsi) dari *class* **Box**.

Pengenalan Objek

Jika *class* menyediakan *blueprint* untuk membuat objek, maka, secara dasarnya, objek dibentuk dari suatu *class*. Pada intinya, objek adalah suatu kumpulan yang memiliki atribut dan metode yang sama (*instance* dari *class*). Dalam konteks variabel, suatu *class* dapat dianggap sebagai tipe data, dan objek sebagai variabelnya.

Contoh: Dari *class* *Flowers* dapat dihasilkan objek *Rose*, *Orchid*, *SunFlower*, dsb.

Sintaks untuk membuat sebuah objek ialah:

```
class_name variable name;
```

Berdasarkan contoh kelas *Box* di atas, maka objeknya ialah:

```
Box obj1, obj2;
```

Berdasarkan contoh di atas, terdapat dua buah objek yang berasal dari kelas Box, yaitu obj1 dan obj2.

Mengakses Data Member dan Function Member

Datamember dan *memberfunction* (anggota data dan fungsi) dapat diakses dengan menggunakan operator (.). Secara umum, sintaks untuk mengakses *datamember* ataupun *functionmember* ialah:

```
object_name.data_member;
```

Berikut merupakan contoh keseluruhan penggalan kode yang telah dibahas:

```
#include<iostream>
usingnamespacestd;

classBox{
public:
    double length; // Panjang box
    double width; // Lebar box
    double height; // Tinggi box
}

int main()
{
    Box obj1, obj2; // Deklarasi 2 objek obj1 dan obj2 untuk kelas Box
    double volume = 0.0; // Menyimpan volume dari Box

    // spesifikasi box obj1
    obj1.height = 4.0;
    obj1.length = 6.0;
    obj1.width = 3.0;

    // spesifikasi box obj2
    obj2.height = 10.0;
    obj2.length = 12.0;
    obj2.width = 12.0;

    // volume box obj1
    volume = obj1.height * obj1.length * obj1.width;
    cout<<"Volume of Box1 : "<< volume <<endl;

    // volume box obj2
    volume = obj2.height * obj2.length * obj2.width;
    cout<<"Volume of Box2 : "<< volume <<endl;
}
```

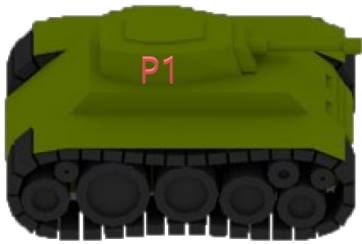
Hasilnya ialah:

Volume of Box1 : 72 Volume of Box2 : 1440
--

BAB III

GAME

3.1. Objek yang dibuat



Tank Player 1
620x349p



Tank Player 2
620x349p



Rudal Hitler
100x100p



Loot Drop
100x100p



Background German
1600x900p



Tutorial Gerakan
1600x900p



Tutorial Scoring
1600x900p



Ending
1600x900p



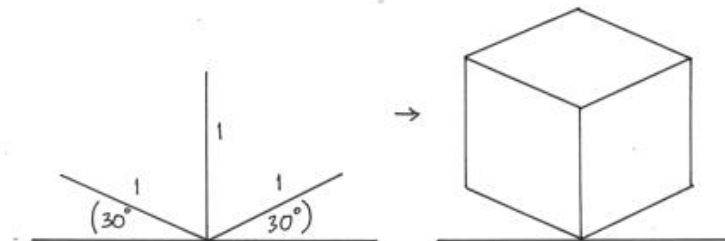
Main Menu
1600x900p

3.2. Penjelasan Pembuatan Gambar Objek

Dalam Pembuatan Game yang berlatar belakang Jerman ini saya menggunakan Perspektif *Isometric* / isometrik yaitu jenis proyeksi aksonometri berpenampilan tiga dimensi atau piktorial dengan besaran sudut masing-masing 120, dan perbandingan masing-masing ukuran tinggi, panjang, dan dalam yaitu 1:1:1. Besar sudut sumbu 120 dapat digunakan alternatif dibuat sudut 30 terhadap horisontal (baik sudut kanan maupun kiri)

Gambar 3.1.1.

Isometrik

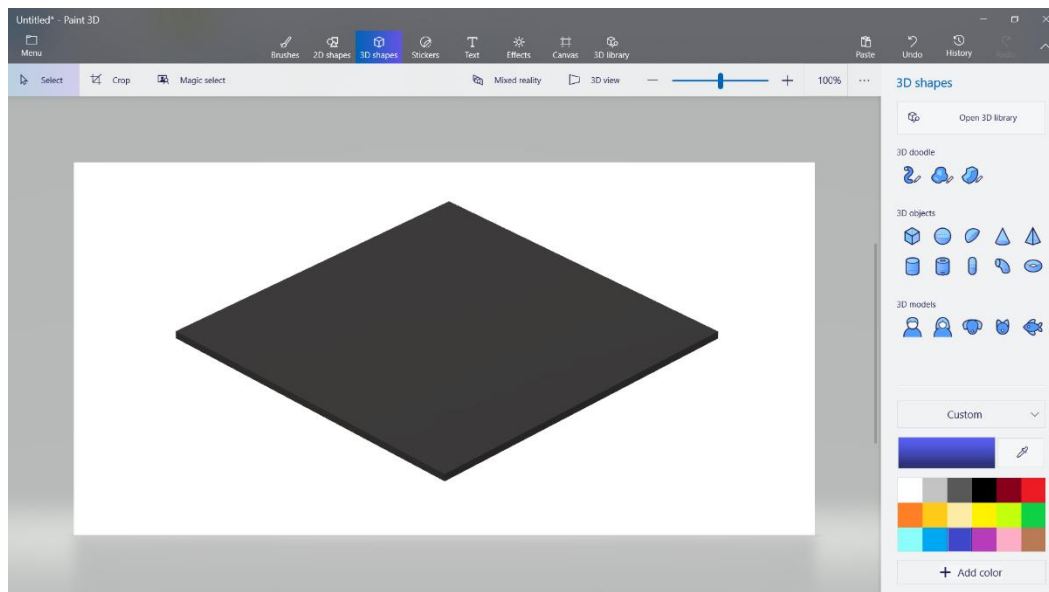


Perseptif

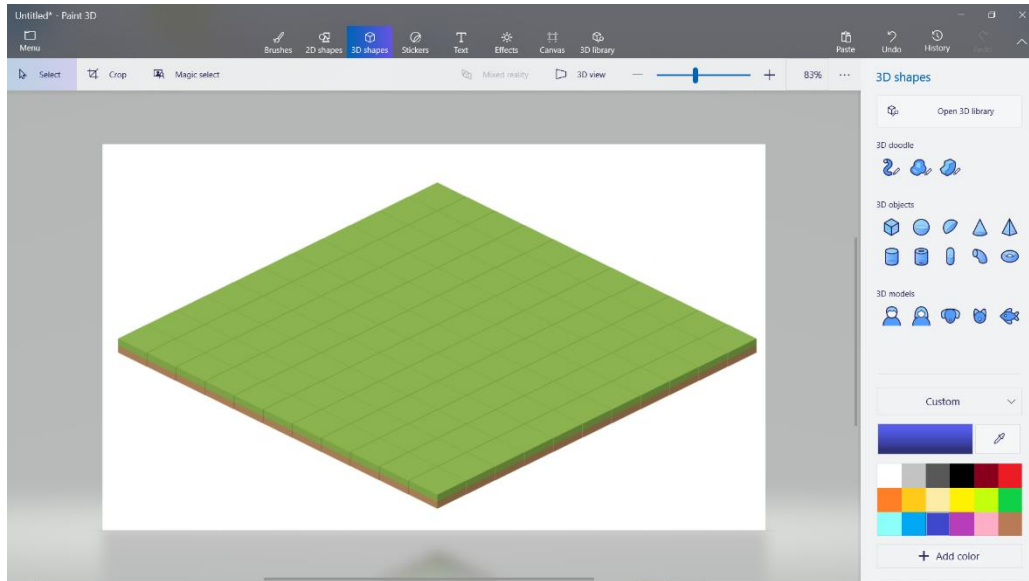
Dengan Perspektif ini saya dapat memanipulasi dunia 2D menjadi lebih 3D. dengan texture isometrik. dalam pengembangan ini saya pun menggunakan aplikasi Editing seperti *COREL DRAW, P 3D* untuk menggambar texture. texture yang saya gambar terlihat seperti ini:



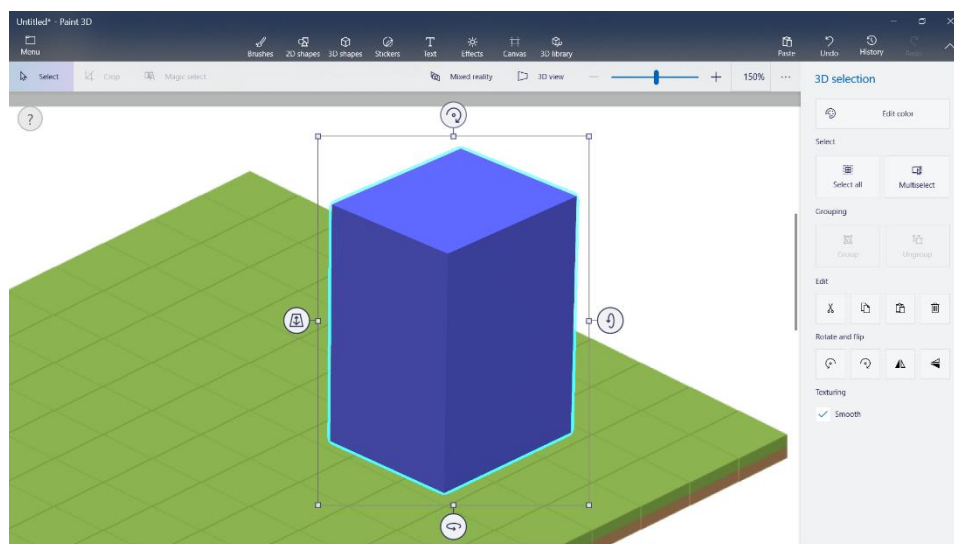
Map tersebut terdapat di sebuah research center di kota Berlin yang digunakan sebagai test subject tank kita. Detail detail tersebut dibuat dengan Menggambar shape pada P. 3D dengan sudut sesuai dengan perspektif isometrik. dengan 3d shape kita dapat menggambar platform secara 3D, namun sebelum masuk ke detail akan lebih baik menyempurnakan lantai terlebih dahulu.



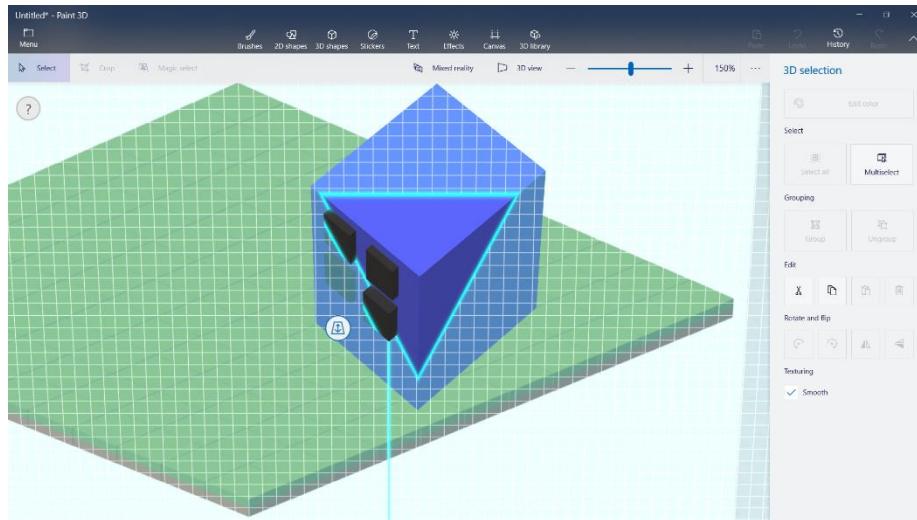
setelah menggambar lantai poles menyesuaikan dengan kondisi background yang akan digunakan, seperti contohnya background jerman, diusahakan membuat tiling agar mudah diatur saat memasukan gedung dan lain lainnya



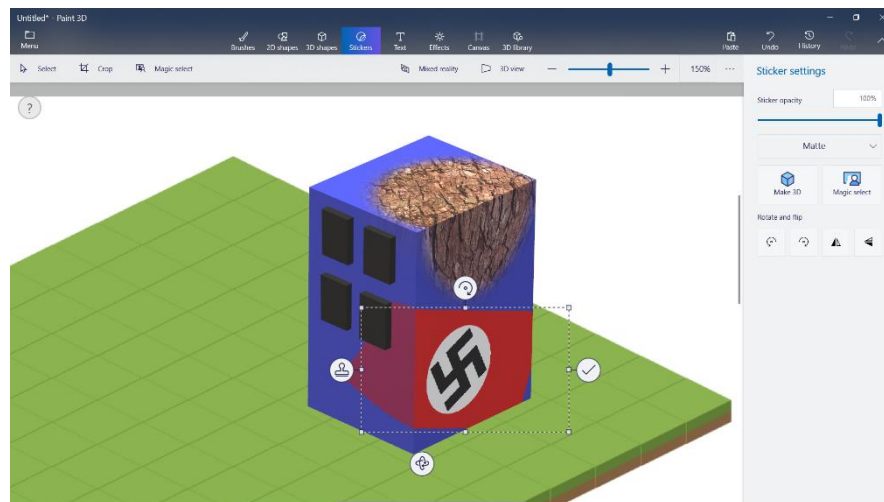
Setelah menggambar floor maka kita sudah bisa menambahkan detail detail yang kita inginkan. seperti contohnya gedung. sama seperti sebelumnya saya juga akan menggunakan perspektif isometrik pada gedung tersebut sehingga terlihat 3D. sama seperti sebelumnya saya akan menggunakan 3d shape kotak/persegi dalam penggambarannya.



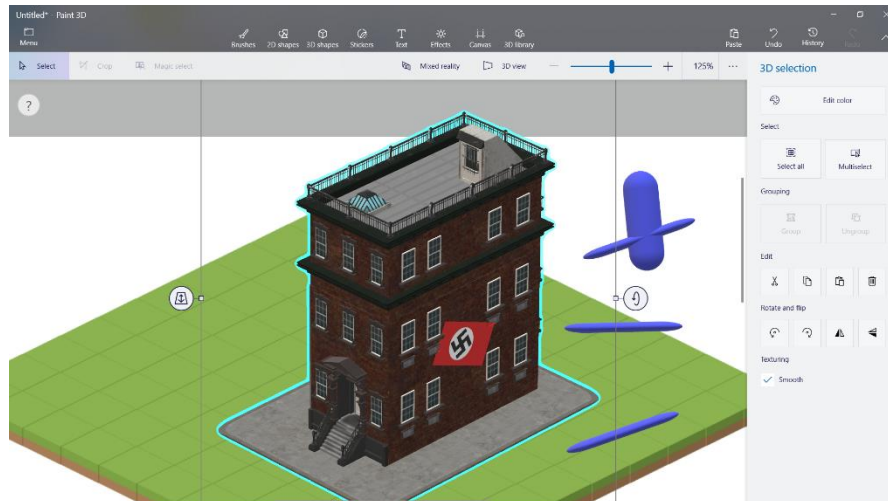
setelah itu kita dapat menambahkan kotak lain agar terlihat seperti memiliki jendela dan detail detail lain. kita dapat menggambarannya melalui 3D shape.



Agar tidak terlihat membosankan tentu detail detail tersebut perlu diberi sebuah wrap/skin agar terlihat lebih real. tentu kita dapat memasukannya dengan menggunakan menu sticker pada aplikasi Paint 3D.



Dengan panduan panduan tersebut maka kita dapat membuat gedung gedung asset pada paint 3d. sebagai contoh inspirasi saya akan melampirkan gedung yang saya buat.



perlu diketahui bahwa pengerjaan gedung tersebut tidak luput dari bantuan bantuan seperti skin dari internet, pagar 3d dari internet. karena jika kita membuat semua dari awal akan sangat memakan waktu.

setelah aset aset map selesai kita bisa langsung menyatukannya di paint 3D. dengan begitu kita bisa mendapatkan 3D map yang indah. dan sesuai dengan perspektif yang kita inginkan. perlu diketahui bahwa setiap melakukan export file pastikan Render dengan format “.PNG ” karena file yang support di SFML cpp hanya bisa menggunakan bmp, *png*, tga, jpg, gif, psd, hdr, *pic*. oleh karena itu saya menggunakan .PNG Karena sangat efisien dan mudah diolah.



setelah map selesai, adapun karakter yang harus kita desain. Karena Game ini Mengambil tema Nazi Tank. maka kita akan membuat Tank.



Adapun tank tersebut digambar menggunakan perspektif yang sama dengan perspektif map dan didesain semenarik mungkin agar memikat para pemain yang kita buat yaitu isometric.

Adapun rudal dan airbox yang juga sama sama termasuk object dalam game ini di desain di dalam paint 3D. sehingga kita dapat melihat ouputnya seperti ini



Dengan menggunakan Paint 3d tersebut telah membantu saya membuat object object yang saya perlukan termasuk menu, score ,tutorial dan lain lain

Dalam pembuatan menu tutorial dan tutorial scoring saya menggunakan object yang telah ada ditambahkan dengan tulisan yang ada, yang berisikan cara menggerakan dan cara scoring. dalam pembuatan halaman tersebut saya menggunakan texture tank yang telah ada.

3.3. Penjelasan tentang cara menggerakan obyek

Setiap objek yang saya masukan ke dalam deklarasi object saya digerakan dengan cara menggunakan vector yaitu jika vector dari posisi mereka dikurangi sebanyak x maka akan mempengaruhi pergerakan secara horizontal. dan apabila dikurangi sebanyak y maka akan mempengaruhi pergerakan secara vertical. dalam pemrograman ini koordinat x dan ynya berada di koordinat kartesius kuadran ke 4

Pada cara penggunaan Tank saya menggunakan Vector dimana jika posisi tank awal diubah yaitu dengan cara dikurangi atau ditambah koordinat x dan y nya maka akan bergerak menuju ke arah yang dikurangi atau ditambah. Contoh : jika menekan w maka vector y akan dikurangi sebanyak (-50) maka gerakannya akan naik sebesar 50 ke y (gerakan ke atas)

Adapun list gerakannya :

```
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::W))
{
    if (Tank1Y > 200)
    {
        Tank1Y = Tank1Y - 2;
    }
}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::D))
{
    if (Tank1X < 700)
    {
        Tank1X = Tank1X + 2;
    }
}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::S))
{
    if (Tank1Y < 460)
    {
        Tank1Y = Tank1Y + 2;
    }
}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::A))
{
    if (Tank1X > 400)
    {
        Tank1X = Tank1X - 2;
    }
}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Up))
{
    if (Tank2Y > 200)
    {
        Tank2Y = Tank2Y - 2;
    }
}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Right))
{
    if (Tank2X < 1050)
    {
        Tank2X = Tank2X + 2;
    }
}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Down))
{
    if (Tank2Y < 460)
    {
        Tank2Y = Tank2Y + 2;
    }
}
```



```

}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Left))
{
    if (Tank2X > 700)
    {
        Tank2X = Tank2X - 2;
    }
}

```

Dimana di sini saya mendeklarasikan Posisi X dan y dari sebuah tank player 1 dengan variabel (Tank1X,Tank1Y) dan tank player 2 dengan variabel (Tank2x,Tank2y). dimana gerakannya saya batasi dengan menggunakan **if** (variabel koordinat Tank (>=) batas yang diinginkan). dan pergerakannya dengan (variabelkoordinatTank = variabelkoordinatTank (+-) nilai yang diinginkan)



Cara menggerakkan objek sudah saya masukan kedalam window tutorial yang dimana menjelaskan cara gerak benda tersebut, hal ini akan memudahkan pemain dalam memahami permainan lebih cepat

3.4. Penjelasan Tentang cara membuat score

Dalam Pembuatan score saya menggunakan deklarasi string dan variabel integer, sebagai wadah dalam pembuatan score. Dalam game saya ini, saya menggunakan 3 jenis score yaitu score player1, score player 2, scoreHitler. Dimana tiap tiap score ini akan mempengaruhi permainan dan sebagai syarat dalam menyelesaikan permainan.

Sistematika Score Awal

Poin Awal Hitler : 100000

Poin Awal Tank Player 1 : 0

Poin Awal Tank Player 2 : 0

Sistematika Pengumpulan Score saat permainan dimulai

Apabila Tembakkan Kena : (tankyangmenembak) + 300 poin

Apabila Terkena Rudal Hitler : (tankyangkenarudal) – 100 && ScoreHitler + 100

Apabila Mengambil Airdrop : (tankyangkenaairdrop) + 3000 poin

Syarat Kemenangan

Apabila salah satu player berhasil melewati scoreHitler maka dia memenangkan permainan.



untuk lebih mudah memahami penambahan score saya memutuskan untuk membuat window khusus tutorial scoring, agar pemain dapat lebih mudah memahami permainan yang saya buat.

3.5. Seluruh Penjelasan terkait dengan game

Game yang saya buat diawali dengan pembukaan yaitu menu, yang dimana menu itu sebagai pembuka game/ sebagai wadah persiapan menuju game. dalam menu ini saya menggunakan deklarasi window baru yang diberi perintah `sf::isKeyPressed` yang dimana jika menekan F nantinya kita akan dilanjutkan menuju menu tutorial



Dalam Main menu ini juga terdapat musik pengantar sebagai salah satu penanda bahwa game sudah run. dengan mengambil tema perang dunia 2 dengan fokus tank, maka background didekorasi sesuai dengan tema perang tersebut.

Setelah menekan F pada main menu, maka pemain akan ditunjukkan ke window Tutorial gerakan, yang dimana pemain akan disajikan dengan informasi mengenai Gerakan gerakan yang mampu dilakukan Tank tersebut, serta dengan sistematika Tembak.



Setelah Menekan F pada tutorial , maka pemain akan ditunjukkan ke window tutorial scoring, yang dimana pemain akan disajikan informasi mengenai sistem scoring serta power maupun tantangan yang ada.



Setelah menekan F pada tutorial terakhir, barulah player bisa merasakan permainan yang ingin dimainkan dengan menggunakan informasi informasi yang sudah ada. lalu selesaikan game dengan cara mengalahkan scoreHitler.



Lalu setelah menyelesaikan game, akan ada sebuah window yang menandakan pemenang dari pertandingan, serta tank yang menang akan dipajang dalam window tersebut dengan musik kemenangan yang menandakan bahwa game sudah selesai.



BAB IV

LISTING PROGRAM

4. List Program

```
#include <SFML\Graphics.hpp>
#include <SFML\Window.hpp>
#include <iostream>
#include <Windows.h>
#include <vector>
#include <math.h>
#include <SFML\Audio.hpp>

using namespace std;

const float bKiri = 300;
const float bKanan = 1200;
//batas kanan dan kiri object rudal

double rSpeed = 5;
double airspeed = 1;
//air speed dan rudal speed

string stringTank1 = "";
string stringTank2 = "";
string stringHitler = "";
//string score

int scoreTank1 = 0;
int scoreTank2 = 0;
int scoreHitler = 100000;
//starting score

const int screenx = 1600;
const int screeny = 900;
//batas layar

const int tank1x = 250;
const int tank1y = 130;
// tank 1
```

```

const int tank2x = 250;
const int tank2y = 130;
// tank 2

int getRandomNumber(int a, int b)
{
    static bool first = true;
    if (first)
    {
        srand(time(NULL));
        first = false;
    }
    int result = a + rand() % ((b + 1) - a);
    result = (result / 10) * 10;
    return result;
}

int getRandomNumber(int a, int b);
int main()
{
    sf::RenderWindow Menu(sf::VideoMode(1600, 900), "I Gusti
Ngurah Agung 072-0073", sf::Style::Close);
    //Untuk main Menu

    sf::RectangleShape Menuback(sf::Vector2f(1600.0f, 900.0f));
    sf::Texture MenuTexture;
    MenuTexture.loadFromFile("Menu.png");
    Menuback.setTexture(&MenuTexture);

    sf::SoundBuffer MenuSong;
    sf::Sound MenuSong1;
    MenuSong1.setVolume(60);
    if (!MenuSong.loadFromFile("Nazi.wav"))
        std::cout << "ERROR" << std::endl;
    //Main Menu Song

    MenuSong1.setBuffer(MenuSong);
    //Menu Song set

    while (Menu.isOpen())
    {

```

```

        sf::Event Menu1;
        if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::F))
            Menu.close();
        while (Menu.pollEvent(Menu1))
        {
            if (MenuSong1.getStatus() == 0)
            {
                MenuSong1.play();
            }
            Menu.display();
            Menu.clear();
            Menu.draw(Menuback);
            if (Menu1.type == Menu1.Closed)
            {
                Menu.close();
            }
        }
    }

    sf::RenderWindow Tutorial(sf::VideoMode(1600, 900), "I
Gusti Ngurah Agung 072-0073 Tutorial Tank", sf::Style::Close);
    //window untuk tutor

    sf::RectangleShape Tutorial2(sf::Vector2f(1600.0f,
900.0f));
    sf::Texture Tutorialtexture;
    Tutorialtexture.loadFromFile("Tutorial.png");
    Tutorial2.setTexture(&Tutorialtexture);
    //Tutor texture

    while (Tutorial.isOpen())
    {
        sf::Event TutorialBar;
        if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::F))
            Tutorial.close();
        while (Tutorial.pollEvent(TutorialBar))
        {
            Tutorial.display();
            Tutorial.clear();
            Tutorial.draw(Tutorial2);
            if (TutorialBar.type == TutorialBar.Closed)

```

```

        {
            Tutorial.close();
        }
    }
}

sf::RenderWindow Scoring(sf::VideoMode(1600, 900), "I Gusti
Ngurah Agung 072-0073 Tutor score", sf::Style::Close);
//window untuk score

sf::RectangleShape Scoring2(sf::Vector2f(1600.0f, 900.0f));
sf::Texture Scoringtexture;
Scoringtexture.loadFromFile("TutorialScore.png");
Scoring2.setTexture(&Scoringtexture);
//score background

while (Scoring.isOpen())
{
    sf::Event ScoringBar;
    if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::F))
        Scoring.close();
    while (Scoring.pollEvent(ScoringBar))
    {
        Scoring.display();
        Scoring.clear();
        Scoring.draw(Scoring2);
        if (ScoringBar.type == ScoringBar.Closed)
        {
            Scoring.close();
        }
    }
}

sf::Font font;
font.loadFromFile("Franchise.ttf");
//font score

sf::SoundBuffer Tembak1;
sf::Sound Tembak11;
Tembak11.setVolume(60);
if (!Tembak1.loadFromFile("loaded.wav"))

```



```

        std::cout << "ERROR" << std::endl;
//sound tembak tank

sf::SoundBuffer Tembak2;
sf::Sound Tembak22;
if (!Tembak2.loadFromFile("loadedtank2.wav"))
    std::cout << "ERROR" << std::endl;
//sound tembak tank 2

Tembak11.setBuffer(Tembak1);
//set tembak 1

Tembak22.setBuffer(Tembak2);
//set tembak 2

sf::SoundBuffer Sound1;
sf::Sound Sound2;
if (!Sound1.loadFromFile("sound1.ogg"))
    std::cout << "ERROR" << std::endl;
//bg music

Sound2.setBuffer(Sound1);
//bg music set

sf::SoundBuffer Collision1;
sf::Sound Collision11;
if (!Collision1.loadFromFile("Tank.wav"))
    std::cout << "ERROR" << std::endl;
//suara ledak impact

Collision11.setBuffer(Collision1);
//set ledakan

sf::SoundBuffer Collision2;
sf::Sound Collision22;
if (!Collision2.loadFromFile("Collision2.wav"))
    std::cout << "ERROR" << std::endl;
//suara ledak impact tank 2

Collision22.setBuffer(Collision2);
//set ledakan

```

```

    sf::RenderWindow window(sf::VideoMode(screenx, screeny), "I
Gusti Ngurah Agung 072-0073 Games", sf::Style::Close);
    window.setFramerateLimit(60);
    //window untuk Games

    double Tank1X, Tank1Y;
    double Tank2X, Tank2Y;

    Tank1X = 2 * screenx / 7;
    Tank1Y = 2 * screeny / 5;

    Tank2X = 4.5 * screenx / 7;
    Tank2Y = 2 * screeny / 5;

    sf::RectangleShape Backgrnd(sf::Vector2f(1600.0f, 900.0f));
    sf::Texture BackgrndTexture;
    BackgrndTexture.loadFromFile("BackgroundAkhirnya2.png");
    Backgrnd.setTexture(&BackgrndTexture);
    //Background kota Berlin

    sf::Texture rudal;
    if (!rudal.loadFromFile("rudal.png"))
        return EXIT_FAILURE;
    sf::Sprite Rudal(rudal);
    double RudalX, RudalY;
    bool rudalNabrak = false;
    RudalX = getRandomNumber(bKiri, bKanan);
    RudalY = 0;
    //rudal

    sf::SoundBuffer Rudal1;
    sf::Sound Rudal11;
    Rudal11.setVolume(60);
    if (!Rudal1.loadFromFile("rudal.wav"))
        std::cout << "ERROR" << std::endl;
    //sound rudal

    Rudal11.setBuffer(Rudal1);

    sf::Texture airdrop;

```

```

if (!airdrop.loadFromFile("loot.png"))
    return EXIT_FAILURE;
sf::Sprite AIRDROP(airdrop);
double airX, airY;
bool airnabrak = false;
airX = getRandomNumber(bKiri, bKanan);
airY = -1000;
//airdrop

sf::SoundBuffer airl1;
sf::Sound airl1;
airl1.setVolume(60);
if (!airl1.loadFromFile("loot.wav"))
    std::cout << "ERROR" << std::endl;
//sound airdrop

airl1.setBuffer(airl1);

int duar;
duar = 0;
sf::CircleShape shape(9.f);
std::vector<sf::CircleShape> peluru;
shape.setFillColor(sf::Color::Black);
//bullet player 1

bool pencet = false;
int duar2;
duar2 = 0;
sf::CircleShape shape2(9.f);
std::vector<sf::CircleShape> peluru2;
shape2.setFillColor(sf::Color::Black);
//bullet player 2

sf::RectangleShape Tank(sf::Vector2f(160.0f, 80.0f));
Tank.setPosition(Tank1X, Tank1Y);
sf::Texture TankTexture;
TankTexture.loadFromFile("tank1.png");
Tank.setTexture(&TankTexture);
//Tank player 1

sf::RectangleShape Tank2(sf::Vector2f(160.0f, 80.0f));

```

```

Tank2.setPosition(Tank2X, Tank2Y);
sf::Texture TankTexture2;
TankTexture2.loadFromFile("tank2.png");
Tank2.setTexture(&TankTexture2);
//Tank player 2

while (window.isOpen())
{
    MenuSong1.stop();
    stringTank1 = "P1 SKOR : " + to_string(scoreTank1);
    sf::Text TextP1(stringTank1, font, 30);
    TextP1.setFillColor(sf::Color::Black);
    TextP1.setPosition(82, 25);

    stringTank2 = "P2 SKOR: " + to_string(scoreTank2);
    sf::Text TextP2(stringTank2, font, 30);
    TextP2.setFillColor(sf::Color::Black);
    TextP2.setPosition(1230, 19);

    stringHitler = "HITLER RUDAL: " +
to_string(scoreHitler);
    sf::Text TextHitler(stringHitler, font, 30);
    TextHitler.setFillColor(sf::Color::Yellow);
    TextHitler.setPosition(615, 25);

    window.clear(sf::Color::White);
    window.draw(Backgrnd);
    window.draw(Tank);
    window.draw(Tank2);
    window.draw(Rudal);
    window.draw(AIRDROP);
    for (int i = 0; i < peluru2.size(); i++)
    {
        window.draw(peluru2[i]);
    }
    for (int i = 0; i < peluru.size(); i++)
    {
        window.draw(peluru[i]);
    }
    window.draw(TextP1);
    window.draw(TextP2);
}

```

```

window.draw(TextHitler);
window.display();

if (scoreHitler < scoreTank1 || scoreHitler <
scoreTank2)
{
    window.close();
}

Tank.setPosition(Tank1X, Tank1Y);
Tank2.setPosition(Tank2X, Tank2Y);
AIRDROP.setPosition(airX, airY);
Rudal.setPosition(RudalX, RudalY);

if (RudalY > 400 || rudalNabrak)
{
    rudalNabrak = false;
    RudalY = 0;
    RudalX = getRandomNumber(bKiri, bKanan);
}
else
{
    RudalY = RudalY + rSpeed;
}

if (airY > 400 || airnabrak)
{
    airnabrak = false;
    airY = -500;
    airX = getRandomNumber(bKiri, bKanan);
}
else
{
    airY = airY + airspeed;
}

sf::Event tbarcntrl;
while (window.pollEvent(tbarcntrl))
{
    if (tbarcntrl.type == tbarcntrl.Closed)

```

```

        {
            window.close();
        }
    }

    //TABRAKAN Tank1
    if (((Tank1X >= (RudalX - 100)) && (Tank1X <= (RudalX
+ 100))) && ((Tank1Y >= (RudalY - 100)) && (Tank1Y <= (RudalY +
100))))
    {
        rudalNabrak = true;
        scoreHitler = scoreHitler + 100;
        scoreTank1 = scoreTank1 - 100;
        Rudal11.play();
    };

    //TABRAKAN Tank2
    if (((Tank2X >= (RudalX - 100)) && (Tank2X <= (RudalX
+ 100))) && ((Tank2Y >= (RudalY - 100)) && (Tank2Y <= (RudalY +
100))))
    {
        rudalNabrak = true;
        scoreHitler = scoreHitler + 100;
        scoreTank2 = scoreTank2 - 100;
        Rudal11.play();
    };

    //TABRAKAN loot
    if (((Tank1X >= (airX - 100)) && (Tank1X <= (airX +
100))) && ((Tank1Y >= (airY - 100)) && (Tank1Y <= (airY +
100))))
    {
        airnabrak = true;
        scoreTank1 = scoreTank1 + 3000;
        air11.play();
    };

    //TABRAKAN loot 2
    if (((Tank2X >= (airX - 100)) && (Tank2X <= (airX +
100))) && ((Tank2Y >= (airY - 100)) && (Tank2Y <= (airY +
100))))

```

```

{
    airnabrak = true;
    scoreTank2 = scoreTank2 + 3000;
    air11.play();
};

if (Sound2.getStatus() == 0)
    Sound2.play();

if (sf::Keyboard::isKeyPressed(sf::Keyboard::L))
{
    pencet = true;
}
if (duar2 >= 30 &&
!sf::Keyboard::isKeyPressed(sf::Keyboard::L) && pencet)
{
    Tembak22.play();
    pencet = false;
    shape2.setPosition(Tank2.getPosition().x + 90,
Tank2.getPosition().y + 30);
    peluru2.push_back(sf::CircleShape(shape2));
    duar2 = 0;
}
if (duar2 < 30)
{
    duar2++;
}

for (int h = 0; h < peluru2.size(); h++) //tank2
{
    peluru2[h].move(-8.0f, 0.0f);
    if
(peluru2[h].getGlobalBounds().intersects(Tank.getGlobalBounds()))
)
    {
        std::cout << "Player 2 hit";
        Collision22.play();
        peluru2.erase(peluru2.begin() + h);
        scoreTank2 = scoreTank2 + 300;
    }
}

```

```

        else if (peluru2[h].getPosition().x > 1600 ||
peluru2[h].getPosition().x < 300)
        {
            peluru2.erase(peluru2.begin() + h);
        }
    }
    if (duar >= 30 &&
sf::Keyboard::isKeyPressed(sf::Keyboard::Space))
    {
        Tembak11.play();
        shape.setPosition(Tank.getPosition().x + 90,
Tank.getPosition().y + 30);
        peluru.push_back(sf::CircleShape(shape));
        duar = 0;
    }
    if (duar < 30)
    {
        duar++;
    }
    for (int i = 0; i < peluru.size(); i++) //tank1
    {
        peluru[i].move(8.0f, 0.0f);
        if
(peluru[i].getGlobalBounds().intersects(Tank2.getGlobalBounds()))
    )
        {
            std::cout << "player hit";
            Collision11.play();
            peluru.erase(peluru.begin() + i);
            scoreTank1 = scoreTank1 + 300;
        }
        else if (peluru[i].getPosition().x > 1250 ||
peluru[i].getPosition().x < 0)
        {
            peluru.erase(peluru.begin() + i);
        }
    }
    if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::W))
    {
        if (Tank1Y > 200)
        {

```



```

        Tank1Y = Tank1Y - 2;
    }
}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::D))
{
    if (Tank1X < 700)
    {
        Tank1X = Tank1X + 2;
    }
}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::S))
{
    if (Tank1Y < 460)
    {
        Tank1Y = Tank1Y + 2;
    }
}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::A))
{
    if (Tank1X > 400)
    {
        Tank1X = Tank1X - 2;
    }
}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Up))
{
    if (Tank2Y > 200)
    {
        Tank2Y = Tank2Y - 2;
    }
}
if
(sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Right))
{
    if (Tank2X < 1050)
    {
        Tank2X = Tank2X + 2;
    }
}
if
(sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Down))

```

```

        {
            if (Tank2Y < 460)
            {
                Tank2Y = Tank2Y + 2;
            }
        }
        if
(sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Left))
        {
            if (Tank2X > 700)
            {
                Tank2X = Tank2X - 2;
            }
        }
        //Control Player 2

        //Control Player 1
    }

    sf::RenderWindow Finish(sf::VideoMode(1600, 900), "I Gusti
Ngurah Agung 072-0073 FINISH", sf::Style::Close);
    //Untuk Ending

    sf::RectangleShape Finish1(sf::Vector2f(1600.0f, 900.0f));
    sf::Texture Finish11;
    Finish11.loadFromFile("finish.png");
    Finish1.setTexture(&Finish11);
    //finish background

    sf::SoundBuffer FinishSong;
    sf::Sound FSong;
    FSong.setVolume(100);
    if (!FinishSong.loadFromFile("play1.ogg"))
        std::cout << "ERROR" << std::endl;
    //Finish Song

    FSong.setBuffer(FinishSong);
    //Finish song

    while (Finish.isOpen())
    {

```

```

sf::Event FinishGame;
if (scoreTank1 > scoreHitler)
{
    Tank.setPosition(450, 450);
    Finish.draw(Tank);

}
if (scoreTank2 > scoreHitler)
{
    Tank2.setPosition(450, 450);
    Finish.draw(Tank2);
}
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::F))
    Finish.close();
while (Finish.pollEvent(FinishGame))
{
    if (FSong.getStatus() == 0)
    {
        FSong.play();
    }
    Finish.display();
    Finish.clear();
    Finish.draw(Finish1);
    if (FinishGame.type == FinishGame.Closed)
    {
        Finish.close();
    }
}
}
std::vector<sf::CircleShape>();
return 0;
}

```

4.1. Penjelasan Class yang dibuat

Pada pembuatan Game ini saya menjalankan game dalam main.cpp, dengan menggunakan 5 window yang dideklarasikan berurutan . Dengan menggunakan Metode Buka Tutup Window. sehingga menjadi sebuah state machine yang menjalankan urutan dari game yang saya buat. Dalam pembuatan ini saya banyak menggunakan lib dari SFML untuk memanggil beberapa beberapa aset kepada window yang sudah saya declare, mulai dari window untuk Main Menu, lalu Window untuk Tutorial Gerakan, Window untuk Tutorial Scoring, Window Game, dan yang terakhir window untuk Finish Game.

Dalam mempercantik Game yang saya buat, saya menambahkan aset aset seperti lagu, sound effect dan texture background yang menarik. hal hal tersebut juga dijalankan di dalam main.cpp. Dengan prinsip penggunaannya ialah buka tutup window, dan jika lagu tidak mau terclose dengan sendirinya maka saya menambahkan (variabelSound).(stop()) sebagai code untuk melakukan perintah memberhentikan lagu.

Game ini tidak akan menarik tanpa adanya kontrol yang baik, dalam membuat kontrol yang baik saya memasukan nilai vector sebagai acuan benda tersebut dan nilai pengurangan dan penjumlahan sebagai pelaksana gerakan, baik maju maupun mundur ataupun atas dan bawah. code ini saya tempatkan di main.cpp

Skema menembak saya menggunakan deklarasi objek yaitu lingkaran pada window lalu memasukan skema gerakanya dengan sistem looping. Dengan prinsip collision jika mengenai lawan maka code penambahan score sebanyak 300 pun akan teraktivasi sehingga akan terjadi penambahan score pada player yang menembak tersebut. skema menembak ini saya tempatkan pada main.cpp

Skema Rudal jatuh menggunakan deklarasi sprite yang nantinya posisinya akan dilooping sehingga terjadi perulangan peluru jatuh. dengan prinsip collision jika mengenai tank maka tank yang terkena akan mendapatkan pengurangan score sebanyak 100 dan score hitler bertambah 100. skema rudal ini saya tempatkan di main.cpp

Setelah target kemenangan dicapai, maka akan timbul window dan menutup window game yang sebelumnya. pada window yang baru ini maka akan diisi dengan status kemangan dari player yang berhasil menyelesaikan/ menyalip scoreHitler.

4.2. Penjelasan mengenai menggerakan objek dalam class

Dalam sistem pergerakan objek Tank saya menjalankannya dalam main.cpp dengan sistematis gerak menggunakan vector sebagai acuan. yang dimana koordinat benda titik x dan y akan mengalami perpindahan apabila dilakukan pengurangan maupun penjumlahan.

```
}  
  
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::W))  
{  
    if (Tank1Y > 200)  
    {  
        Tank1Y = Tank1Y - 2;  
    }  
}  
  
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::D))  
{  
    if (Tank1X < 700)  
    {  
        Tank1X = Tank1X + 2;  
    }  
}  
  
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::S))  
{  
    if (Tank1Y < 400)  
    {  
        Tank1Y = Tank1Y + 2;  
    }  
}  
  
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::A))  
{  
    if (Tank1X > 400)  
    {  
        Tank1X = Tank1X - 2;  
    }  
}  
  
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Up))  
{  
    if (Tank2Y > 200)  
    {  
        Tank2Y = Tank2Y - 2;  
    }  
}  
  
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Right))  
{  
    if (Tank2X < 1050)  
    {  
        Tank2X = Tank2X + 2;  
    }  
}  
  
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Down))  
{  
    if (Tank2Y < 400)  
    {  
        Tank2Y = Tank2Y + 2;  
    }  
}  
  
if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::Left))  
{  
    if (Tank2X > 700)  
    {  
        Tank2X = Tank2X - 2;  
    }  
}
```

Seperti yang dapat dilihat dalam source code diatas pergerakan menggunakan penjumlahan dan pengurangan. karena pada window yang dibuka pada SFML koordinat yang digunakan ada kuadran yang keempat maka setiap pengurangan pada nilai y maka benda akan naik ke atas sedangkan jika y dijumlah maka benda akan turun kebawah.

Object rudal dan airdrop memiliki sistem kerja yaitu benda turun menuju kebawah dengan batas yang ditentukan setelah turun maka jika mengenai player maka loop di set ulang kembali. Setiap lokasi rudal yang turun diatur oleh Fungsi getRandomNumber yang dimana fungsi tersebut memberikan posisi random pada rudal yang akan turun. begitu juga dengan airdrop. namun sistem airdrop dibuat lebih lambat, agar keseimbangan skor tetap terjaga.

```
if (RudalY > 400 || rudalNabrak)
{
    rudalNabrak = false;
    RudalY = 0;
    RudalX = getRandomNumber(bKiri, bKanan);
}
else
{
    RudalY = RudalY + rSpeed;
}

if (airY > 400 || airNabrak)
{
    airNabrak = false;
    airY = -500;
    airX = getRandomNumber(bKiri, bKanan);
}
else
{
    airY = airY + airspeed;
}
```

4.3. Penjelasan Metoda Tumbukan Tabrakan

Dalam game yang saya buat terjadi beberapa tumbukan dan tabrakan. salah satu seperti object peluru. dalam object peluru ini saya menggunakan circleShape sebagai fungsi yang akan digunakan sebagai object. object inilah yang nanti akan berinteraksi dengan object lainnya yaitu tank yang akan memberikan feedback balik berupa suara dan penambahan score

```

}
if (duar >= 30 && sf::Keyboard::isKeyPressed(sf::Keyboard::Space))
{
    Tembaki1.play();
    shape.setPosition(Tank.getPosition().x + 90, Tank.getPosition().y + 30);
    peluru.push_back(sf::CircleShape(shape));
    duar = 0;
}
if (duar < 30)
{
    duar++;
}

for (int i = 0; i < peluru.size(); i++) //tank1
{
    peluru[i].move(8.0f, 0.0f);
    if (peluru[i].getGlobalBounds().intersects(Tank2.getGlobalBounds()))
    {
        std::cout << "player hit";
        Collision1.play();
        peluru.erase(peluru.begin() + i);
        scoreTank1 = scoreTank1 + 300;
    }
    else if (peluru[i].getPosition().x > 1250 || peluru[i].getPosition().x < 0)
    {
        peluru.erase(peluru.begin() + i);
    }
}
}

```

Bisa dilihat dalam sourcecode tersebut terjadi penembakan dengan input space yang dimana, deklarasi musik bernilai true, gerakan peluru bernilai true. sehingga peluru keluar dengan bunyi tembakan. lalu dari pihak Tank akan mengambil nilai dari getGlobalBounds sehingga pihak Tank dapat mendeteksi Tabrakan bola yang datang. lalu setelah mengenai Tank, peluru tersebut pun kembali reset dengan Fungsi erase pada SFML.

Object rudal dan airdrop memiliki sistem kerja yaitu benda turun menuju kebawah dengan batas yang ditentukan setelah turun maka jika mengenai player maka loop di set ulang kembali. Setiap lokasi rudal yang turun diatur oleh Fungsi getRandomNumber yang dimana fungsi tersebut memberikan posisi random pada rudal yang akan turun. begitu juga dengan airdrop. namun sistem airdrop dibuat lebih lambat, agar keseimbangan skor tetap terjaga.

Lalu setelah sampai pada objek Tank. maka tabrakan terjadi maka music collision yang sudah di declare sebelumnya akan mulai bernilai true dan akan memainkan suara. serta memberikan skor 100 poin ke scoreHitler dan mengurangi skor player sejumlah 100.

4.4. Penjelasan Seluruh Program

Game yang dijalankan menggunakan tujuh library, yakni `iostream`, `window`, `vector`, `math.h`, dan 3 lib dari SFML, yaitu `graphic.hpp`, `audio.hpp`, `window.hpp`. Pada awal program ini saya juga mendeklarasikan using namespace `std`. yg pertama saya lakukan ada melakukan deklarasi untuk `const`, `double`, `string`, `int`, dan `const int`. lalu mendklarasikan `get random number` sebagai alat untuk object object yang akan saya gunakan kedepannya.

Dalam source code saya, saya membagi game saya ini menjadi 5 bagian : yang pertama bagian Menu. Pada bagian menu saya mendaklarasikan SFML `renderwindow` dengan variabel yang saya namakan Menu saya memilih ukuran window saya 1600 x 900 pixel karena merupakan resolusi yang umum dipakai di berbagai komputer. lalu saya mendeklarasikan object berupa `Rectangle` dengan ukuran yang sama dengan resolusi saya sehingga kesannya menjadi sebuah background dari Menu saya nantinya. `rectangle` tersebut tentu saya beri sebuah texture yaitu berupa file PNG yang sudah saya olah yaitu `Menu.png` lalu set texturenya kepada `rectangle` tersebut.

selain `rectangle`, jika ingin menu memilki suara maka hendaknya mendeklarasikan SFML `sound buffer`. variable ini sifatnya bebas sehingga kita dapat mengaturnya sesuka hati. selain itu kita juga `menddeclare sound` sebagai wadah untuk menyimpan file sound yang ingin kita gunakan, setelah set buffer di set. maka sekarang waktunya membuat Menu.

konsep dari main menu ini ialah jika player menekan huruf F maka itu akan berlanjut ke window selanjutnya. kita bisa mulai dari `while window open` (saat window itu terbuka) kita masukan `sf event` sebagai deklarasi valid pada suatu acara. yang pertama kita harus ingat ialah melakukan `draw` terhadap texture `rectangle` yang kita buat, serta melakukan `draw` juga pada window sehingga window bisa di render. dengan ditambahnya `SF iskeypressed F` lalu ditambahkan dengan fungsi jika ditekan f maka `Menu.Close` yang akan bernilai true, sehingga tujuan awal untuk membuat Menu tersebut close dengan menekan F sudah berhasil. yang perlu dilakukan selanjutnya adalah melengkapi audio yang sudah dideklarasikan tadi agar bernilai true sehingga audio pun berbunyi.

Setelah Menu komplit, selanjutnya saya ingin membuat window untuk tutorial. yang dimana sama seperti sebelumnya mendeklarasikan window lalu ditambahkan `rectangle` sebagai background. disini texture yang saya gunakan sudah mengandung isian isian dari tutorial yang ingin saya sampaikan sehingga cukup melakukan set pada texture tersebut dan lakukan pembukaan

window seperti sebelumnya menggunakan while window is open. lakukan draw window dan texture Dan jangan lupa untuk menambahkan Iskeypressed F untuk menutup windowsnya.

Proses pada tutorial score cukup persis karena yang kita hanya ingin tampilkan ialah window yang mengandung texture yang kita telah buat. sehingga cukup ulangi saja proses sebelumnya namun diganti source texturenya dengan texture yang baru

Setelah Menu, Tutorial, Tutorial scoring selesai, saatnya berfokus pada window milik game. seperti biasa yang kita lakukan adalah melakukan beberapa deklarasi seperti deklarasi window, deklarasi suara, deklarasi font yang akan digunakan, deklarasi object object yang akan kita gunakan. pada deklarasi object saya menginginkan tank player berada pada kiri tengah window dan player2 di kanan tengah, hal ini dapat dilakukan dengan membagi posisi tank sesuai dengan resolusi window yang digunakan. setelah semua lengkap maka barulah kita membuka window seperti dengan cara sebelumnya, while window is open. lalu yang pertama kita lakukan ialah melakukan set string yang telah kita deklarasikan diatas tadi sesuai dengan yang kita inginkan. Sesuai dengan konsep awal saya menggunakan 3 score yang dimana di window ini sudah saya set sebagaimana tempat yang tepat, dengan warna yang tepat. lalu draw semua asset, setelah itu pasang sistem tembak menggunakan looping for dengan nilai penambahan. pasang kondisi menang yang dimana window akan tertutup. lalu masukan asset rudal yang dimana getRandomNumber sangat berguna dalam memberikan nilai random sebagai letak spawn dari peluru, jangan lupa untuk melakukan looping erase.begin sehingga rudal akan terus berjatuhan. lakukan hal yang sama pada airdrop.

Setelah semua sistem dan aset siap, selanjutnya masuk ke pengaktifan efek collision sesuai dengan konsep yaitu setiap terkena rudal poin hitler +100 dan poin player -100. sedangkan jika terkena airdrop maka poin +3000. setelah itu lanjutkan dengan mengaktifkan trigger tembak tank dengan menambahkan sf::iskeypressed space untuk player 1 dan sf::iskeypressed L untuk player 2. lalu tambahkan efek collision dari setiap tembakan yang dikeluarkan seperti penambahan efek suara dan score yang didapat saat menembak. lalu setelah semua siap tambahkan semua sistem gerak untuk tank. yang dimana menggunakan vector sehingga dalam penggerakannya koordinat dari x,y suatu tank akan dikurangi/ditambah. perlu diingat bahwa window yang di declare di SFML menggunakan koordinat kartesius pada kuadran ke 4. sehingga nilai y tank dikurangi tank akan bergerak ke atas. dan jika nilai y tank ditambah tank akan bergerak ke bawah.

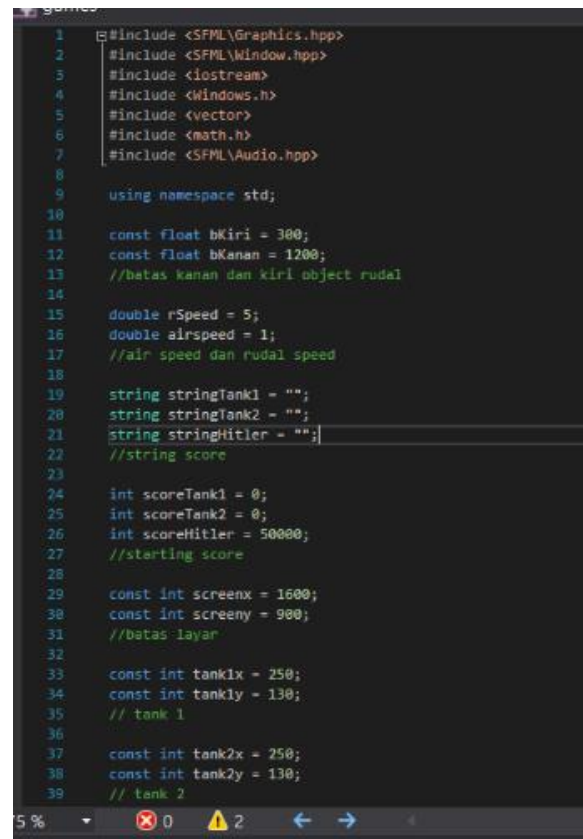
Setelah window game selesai, lanjut ke window game over. sama seperti sebelumnya kita mendeclare window, audio, background. lalu sama seperti sebelumnya kita membuka window dengan while window is open. lalu kita masukan Object tank yang berhasil memenangkan game. lalu setelah itu masukan input keyboard f lagi sebagai penanda game selesai melalui sf::iskeypressed F. lengkapi window seperti sebelumnya. untuk menghindari beberapa bug saya menambahkan vector pada bawah source code. setelah semua selesai maka game pun siap dimainkan.

BAB V

SCREEN SHOT PROGRAM DENGAN KETERANGAN

Keterangan :

Pada program dibawah ini merupakan deklarasi deklarasi yang nantinya akan digunakan pada object object pada game ini. yang paling atas terdapat `#include` yang merupakan library yang saya gunakan dalam membuat game ini. `const float bkiri` dan `bkanan` merupakan batas batas yang saya gunakan dalam object object tertentu seperti peluru dan batas sumbu x rudal dan batas sumbu x airdrop. `double rspeed` dan `airspeed` merupakan deklarasi yang saya gunakan untuk mengatur kecepatan rudal dan airdrop nantinya. `string` sebagai tipe data yang saya gunakan untuk mengatur mendisplay text score nantinya. sedangkan integer nantinya akan digunakan sebagai starting score bagi masing masing player dan score untuk hitler. `const int screen x` dan `screen y` dideklarasikan `vector x` dan `y` yang nantinya akan menjadi batas dari window maupun sebagai patokan tank dalam menentukan titik awal. lalu `const tank` merupakan ukuran dari texture tank



```
1 #include <SFML\Graphics.hpp>
2 #include <SFML\Window.hpp>
3 #include <iostream>
4 #include <windows.h>
5 #include <vector>
6 #include <math.h>
7 #include <SFML\Audio.hpp>
8
9 using namespace std;
10
11 const float bkiri = 300;
12 const float bkanan = 1200;
13 //batas kanan dan kiri object rudal
14
15 double rSpeed = 5;
16 double airspeed = 1;
17 //air speed dan rudal speed
18
19 string stringTank1 = "";
20 string stringTank2 = "";
21 string stringHitler = "";
22 //string score
23
24 int scoreTank1 = 0;
25 int scoreTank2 = 0;
26 int scoreHitler = 50000;
27 //starting score
28
29 const int screenx = 1600;
30 const int screeny = 900;
31 //batas layar
32
33 const int tank1x = 250;
34 const int tank1y = 130;
35 // tank 1
36
37 const int tank2x = 250;
38 const int tank2y = 130;
39 // tank 2
```

Keterangan :

Dimulai dari deklarasi `getRandomNumber` yang merupakan sebuah cara untuk mendapatkan nilai acak dari suatu inputan, hal tersebut akan digunakan dalam mengacak rudal jatuh. lalu ada deklarasi `sf::RenderWindow` yang dimana hal tersebut akan membuat variabel window yang dimana saya menggunakan resolusi 1600x900 dengan nama dan nrp saya sebagai judul pada title bar. selanjutnya saya mendeklarasikan rectangle shape sebagai background yang nantinya akan diisi texture. saya juga mendeklarasikan sound buffer yang nantinya dipakai untuk melakukan play suara terhadap perintah yang dilakukan. lalu kita mulai dalam pembukaan window untuk main menu. yaitu dengan melakukan loop while/ketika, sehingga jika syarat terpenuhi kita bisa menambahkan hal hal yg diinginkan. jangan lupa menambahkan draw dan `sf::isKeyPressed` agar dapat berlanjut menuju windows lain.

```
ames (Global Sco

int getRandomNumber(int a, int b)
{
    static bool first = true;
    if (first)
    {
        srand(time(NULL));
        first = false;
    }
    int result = a + rand() % ((b + 1) - a);
    result = (result / 10) * 10;
    return result;
}

int getRandomNumber(int a, int b);
int main()
{
    sf::RenderWindow Menu(sf::VideoMode(1600, 900), "I Gusti Ngurah Agung 072-0073", sf::Style::Close);
    //Untuk main Menu

    sf::RectangleShape MenuBack(sf::Vector2f(1600.0f, 900.0f));
    sf::Texture MenuTexture;
    MenuTexture.loadFromFile("Menu.png");
    MenuBack.setTexture(&MenuTexture);

    sf::SoundBuffer MenuSong;
    sf::Sound MenuSong1;
    MenuSong1.setVolume(60);
    if (!MenuSong.loadFromFile("Nari.wav"))
        std::cout << "ERROR" << std::endl;
    //Main Menu Song

    MenuSong1.setBuffer(MenuSong);
    //Menu Song set

    while (Menu.isOpen())
    {
        sf::Event Menu1;
        if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::F))
            Menu.close();
        while (Menu.pollEvent(Menu1))
        {
            if (MenuSong1.getStatus() == 0)
            {
                MenuSong1.play();
            }
            Menu.display();
            Menu.clear();
            Menu.draw(MenuBack);
            if (Menu1.type == Menu1.Closed)
            {
                Menu.close();
            }
        }
    }
}
```

Keterangan :

sama seperti sebelumnya pada 2 window kali ini kita perlu mendeklarasikan window, lalu menambahkan rectangle sebagai background pada window. lalu saat sudah mulai looping jangan lupa menambahkan draw dan sf::isKeyPressed sebagai initiator untuk lanjut menuju next window. hal ini berlaku untuk kedua window ini karena tugasnya sama yaitu untuk menampilkan tutorial game yang akan dimainkan

```
sf::RenderWindow Tutorial(sf::VideoMode(1600, 900), "I Gusti Ngurah Agung 072-0073 Tutorial Tank", sf::Style::Close);
//window untuk tutor

sf::RectangleShape Tutorial2(sf::Vector2f(1600.0f, 900.0f));
sf::Texture TutorialTexture;
TutorialTexture.loadFromFile("Tutorial.png");
Tutorial2.setTexture(&TutorialTexture);
//Tutor texture

while (Tutorial.isOpen())
{
    sf::Event TutorialBar;
    if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::F))
        Tutorial.close();
    while (Tutorial.pollEvent(TutorialBar))
    {
        Tutorial.display();
        Tutorial.clear();
        Tutorial.draw(Tutorial2);
        if (TutorialBar.type == TutorialBar.Closed)
        {
            Tutorial.close();
        }
    }
}

sf::RenderWindow Scoring(sf::VideoMode(1600, 900), "I Gusti Ngurah Agung 072-0073 Tutor score", sf::Style::Close);
//window untuk score

sf::RectangleShape Scoring2(sf::Vector2f(1600.0f, 900.0f));
sf::Texture ScoringTexture;
ScoringTexture.loadFromFile("TutorialScore.png");
Scoring2.setTexture(&ScoringTexture);
//score Background

while (Scoring.isOpen())
{
    sf::Event ScoringBar;
    if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::F))
        Scoring.close();
    while (Scoring.pollEvent(ScoringBar))
    {
        Scoring.display();
        Scoring.clear();
        Scoring.draw(Scoring2);
        if (ScoringBar.type == ScoringBar.Closed)
        {
            Scoring.close();
        }
    }
}
```

Keterangan :

dalam source kali ini hanya berisi deklarasi deklarasi variabel yang akan digunakan dalam window game.

```
sf::Font font;
font.loadFromFile("Franchise.ttf");
//font score

sf::SoundBuffer Tembak1;
sf::Sound Tembak1;
Tembak1.setVolume(60);
if (!Tembak1.loadFromFile("loaded.wav"))
    std::cout << "ERROR" << std::endl;
//sound Tembak Tank

sf::SoundBuffer Tembak2;
sf::Sound Tembak2;
if (!Tembak2.loadFromFile("loadedTank2.wav"))
    std::cout << "ERROR" << std::endl;
//sound Tembak Tank 2

Tembak1.setBuffer(Tembak1);
//set tembak 1

Tembak2.setBuffer(Tembak2);
//set tembak 2

sf::SoundBuffer Sound1;
sf::Sound Sound2;
if (!Sound1.loadFromFile("sound1.ogg"))
    std::cout << "ERROR" << std::endl;
//bg music

Sound2.setBuffer(Sound1);
//bg music set

sf::SoundBuffer Collision1;
sf::Sound Collision1;
if (!Collision1.loadFromFile("Tank.wav"))
    std::cout << "ERROR" << std::endl;
//suara ledak impact

Collision1.setBuffer(Collision1);
//set ledakan

sf::SoundBuffer Collision2;
sf::Sound Collision2;
if (!Collision2.loadFromFile("Collision2.wav"))
    std::cout << "ERROR" << std::endl;
//suara ledak impact Tank 2

Collision2.setBuffer(Collision2);
//set ledakan
```

Keterangan :

code dibawah ini banyak seperti sebelumnya merupakan deklarasi deklarasi. dengan tambahan Tank1x dan 1y sebagai deklarasi posisi tank di window.

```
sf::RenderWindow window(sf::VideoMode(screenx, screeny), "I Gusti Ngurah Agung 072-0073 Games", sf::Style::Close);
window.setFramerateLimit(60);
//window untuk Games

double Tank1X, Tank1Y;
double Tank2X, Tank2Y;

Tank1X = 2 * screenx / 7;
Tank1Y = 2 * screeny / 5;

Tank2X = 4.5 * screenx / 7;
Tank2Y = 2 * screeny / 5;

sf::RectangleShape Backgrnd(sf::Vector2f(1600.0f, 900.0f));
sf::Texture BackgrndTexture;
BackgrndTexture.loadFromFile("BackgroundAkhilrmys2.png");
Backgrnd.setTexture(&BackgrndTexture);
//Background kota Berlin

sf::Texture rudal;
if (!rudal.loadFromFile("rudal.png"))
    return EXIT_FAILURE;
sf::Sprite Rudal(rudal);
double RudalX, RudalY;
bool rudalMabrak = false;
RudalX = getRandomNumber(bKiri, bKanan);
RudalY = 0;
//rudal

sf::SoundBuffer Rudal1;
sf::Sound Rudal11;
Rudal11.setVolume(60);
if (!Rudal1.loadFromFile("rudal.wav"))
    std::cout << "ERROR" << std::endl;
//sound rudal

Rudal11.setBuffer(Rudal1);

sf::Texture airdrop;
if (!airdrop.loadFromFile("loot.png"))
    return EXIT_FAILURE;
sf::Sprite AIRDROP(airdrop);
double airX, airY;
bool airnebrak = false;
airX = getRandomNumber(bKiri, bKanan);
airY = -1000;
//airdrop

sf::SoundBuffer air1;
```

Keterangan :

Dimana disini window looping game sudah mulai diinisiasi, dan diawali dengan code string sebagai penampil score pada game yang akan di run. lalu terdapat fungsi draw sebagai penampil texture dll. selanjutnya penambahan sistem tembak yang sudah mulai diinclude yang dimana ini akan menggambarkan peluru pada window game tersebut. dan terakhir penambahan sistem score dibawah dengan kondisi menang jika score hitler sudah tersalip oleh player.

```
while (window.isOpen())
{
    MenuSong1.stop();
    stringTank1 = "P1 SKOR : " + to_string(scoreTank1);
    sf::Text TextP1(stringTank1, font, 30);
    TextP1.setFillColor(sf::Color::Black);
    TextP1.setPosition(62, 25);

    stringTank2 = "P2 SKOR : " + to_string(scoreTank2);
    sf::Text TextP2(stringTank2, font, 30);
    TextP2.setFillColor(sf::Color::Black);
    TextP2.setPosition(1238, 19);

    stringHitler = "HITLER RUJAL : " + to_string(scoreHitler);
    sf::Text TextHitler(stringHitler, font, 30);
    TextHitler.setFillColor(sf::Color::Yellow);
    TextHitler.setPosition(615, 25);

    window.clear(sf::Color::White);
    window.draw(Backgrnd);
    window.draw(Tank);
    window.draw(Tank2);
    window.draw(Rujal);
    window.draw(AIRBORP);
    for (int i = 0; i < peluru2.size(); i++)
    {
        window.draw(peluru2[i]);
    }
    for (int i = 0; i < peluru.size(); i++)
    {
        window.draw(peluru[i]);
    }
    window.draw(TextP1);
    window.draw(TextP2);
    window.draw(TextHitler);
    window.display();

    if (scoreHitler < scoreTank1 || scoreHitler < scoreTank2)
    {
        window.close();
    }

    Tank.setPosition(Tank1X, Tank1Y);
    Tank2.setPosition(Tank2X, Tank2Y);
    AIRBORP.setPosition(alrX, alrY);
    Rujal.setPosition(RujalX, RujalY);
}
```


Keterangan :

Penambahan rudal pada window game dengan mengatur getrandomnumber sebagai pengisi posisi pada x dan y pada window. serta penggunaan variabel rspeed airtpeed untuk mengatur kecepatan turunya rudal dan airdrop. selain itu juga sudah terdapat penambahan collision untuk rudal pada player dengan output berupa suara dan score.

```
{
    rudalNabrak = false;
    RudalY = 0;
    RudalX = getRandomNumber(bKiri, bKanan);
}
else
{
    RudalY = RudalY + rSpeed;
}

if (airY > 400 || airtabrak)
{
    airtabrak = false;
    airY = -500;
    airX = getRandomNumber(bKiri, bKanan);
}
else
{
    airY = airY + airtpeed;
}

sf::Event tberctrl;
while (window.pollEvent(tberctrl))
{
    if (tberctrl.type == tberctrl.Closed)
    {
        window.close();
    }
}

//TABRAKAN Tank1
if (((Tank1X >= (RudalX - 100)) && (Tank1X <= (RudalX + 100))) && ((Tank1Y >= (RudalY - 100)) && (Tank1Y <= (RudalY + 100))))
{
    rudalNabrak = true;
    scoreHitler = scoreHitler + 100;
    scoreTank1 = scoreTank1 - 100;
    Rudal11.play();
};

//TABRAKAN Tank2
if (((Tank2X >= (RudalX - 100)) && (Tank2X <= (RudalX + 100))) && ((Tank2Y >= (RudalY - 100)) && (Tank2Y <= (RudalY + 100))))
{
    rudalNabrak = true;
    scoreHitler = scoreHitler + 100;
    scoreTank2 = scoreTank2 - 100;
    Rudal11.play();
};
```

Keterangan :

Penambahan collision terhadap peluru tank yang saling bertabrakan, dengan output berupa music dan score. serta sistem tembak juga ditambahkan disini dengan menekan space pada player 1 dan menekan L pada player 2.

```
    }  
    for (int h = 0; h < peluru2.size(); h++) //tank2  
    {  
        peluru2[h].move(-8.0f, 0.0f);  
        if (peluru2[h].getGlobalBounds().intersects(Tank2.getGlobalBounds()))  
        {  
            std::cout << "Player 2 hit";  
            Collision22.play();  
            peluru2.erase(peluru2.begin() + h);  
            scoreTank2 = scoreTank2 + 300;  
        }  
        else if (peluru2[h].getPosition().x > 1600 || peluru2[h].getPosition().x < 300)  
        {  
            peluru2.erase(peluru2.begin() + h);  
        }  
    }  
    if (duar >= 30 && sf::Keyboard::isKeyPressed(sf::Keyboard::Space))  
    {  
        Tembaki1.play();  
        shape.setPosition(Tank.getPosition().x + 90, Tank.getPosition().y + 30);  
        peluru.push_back(sf::CircleShape(shape));  
        duar = 0;  
    }  
    if (duar < 30)  
    {  
        duar++;  
    }  
    for (int i = 0; i < peluru.size(); i++) //Tank1  
    {  
        peluru[i].move(8.0f, 0.0f);  
        if (peluru[i].getGlobalBounds().intersects(Tank2.getGlobalBounds()))  
        {  
            std::cout << "player hit";  
            Collision11.play();  
            peluru.erase(peluru.begin() + i);  
            scoreTank1 = scoreTank1 + 300;  
        }  
        else if (peluru[i].getPosition().x > 1250 || peluru[i].getPosition().x < 0)  
        {  
            peluru.erase(peluru.begin() + i);  
        }  
    }  
    if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::N))  
    {  
        if (Tank1V > 200)  
        {  
            Tank1V = Tank1V - 2;  
        }  
    }  
    if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::D))
```

Keterangan :

Penambahan Sistem gerak object pada tank 1 dan tank 2 yang dimana dipengaruhi koordinat x dan y, dimana jika tombol ditekan maka sistem geraknya akan melakukan penjumlahan/pengurangan terhadap vector object semula.

```
    }  
}  
if (sf::Keyboard::IsKeyPressed(sf::Keyboard::Key::W))  
{  
    if (Tank1Y > 200)  
    {  
        Tank1Y = Tank1Y - 2;  
    }  
}  
if (sf::Keyboard::IsKeyPressed(sf::Keyboard::Key::D))  
{  
    if (Tank1X < 700)  
    {  
        Tank1X = Tank1X + 2;  
    }  
}  
if (sf::Keyboard::IsKeyPressed(sf::Keyboard::Key::S))  
{  
    if (Tank1Y < 460)  
    {  
        Tank1Y = Tank1Y + 2;  
    }  
}  
if (sf::Keyboard::IsKeyPressed(sf::Keyboard::Key::A))  
{  
    if (Tank1X > 400)  
    {  
        Tank1X = Tank1X - 2;  
    }  
}  
if (sf::Keyboard::IsKeyPressed(sf::Keyboard::Key::Up))  
{  
    if (Tank2Y > 200)  
    {  
        Tank2Y = Tank2Y - 2;  
    }  
}  
if (sf::Keyboard::IsKeyPressed(sf::Keyboard::Key::Right))  
{  
    if (Tank2X < 1050)  
    {  
        Tank2X = Tank2X + 2;  
    }  
}  
if (sf::Keyboard::IsKeyPressed(sf::Keyboard::Key::Down))  
{  
    if (Tank2Y < 460)  
    {  
        Tank2Y = Tank2Y + 2;  
    }  
}  
if (sf::Keyboard::IsKeyPressed(sf::Keyboard::Key::Left))
```

Keterangan :

setelah window terakhir selesai dimainkan dan mendapatkan pemenang, maka diperlukannya window game over agar kita dapat melihat pemenang dari permainan tersebut. adapun. pendeklarasian dilakukan sama seperti yang sebelumnya namun yang berbeda disini ialah scoreplayer yang berhasil menyalip score hitler akan ditampilkan texture playernya sehingga memberikan rasa bangga akan kemenangan yang diraih.

```
sf::RenderWindow Finish(sf::VideoMode(1600, 900), "I Gusti Ngurah Agung 072-0072 FINISH", sf::Style::Close);
//Untuk Ending

sf::RectangleShape Finish1(sf::Vector2f(1600.0f, 900.0f));
sf::Texture Finish11;
Finish11.loadFromFile("finish.png");
Finish1.setTexture(&Finish11);
//Finish Background

sf::SoundBuffer FinishSong;
sf::Sound FSong;
FSong.setVolume(100);
if (!FinishSong.loadFromFile("play1.ogg"))
    std::cout << "ERROR" << std::endl;
//Finish Song

FSong.setBuffer(FinishSong);
//Finish song

while (Finish.isOpen())
{
    sf::Event FinishGame;
    if (scoreTank1 > scoreHitler)
    {
        Tank.setPosition(450, 450);
        Finish.draw(Tank);
    }
    if (scoreTank2 > scoreHitler)
    {
        Tank2.setPosition(450, 450);
        Finish.draw(Tank2);
    }
    if (sf::Keyboard::isKeyPressed(sf::Keyboard::Key::F))
        Finish.close();
    while (Finish.pollEvent(FinishGame))
    {
        if (FSong.getStatus() == 0)
        {
            FSong.play();
        }
        Finish.display();
        Finish.clear();
        Finish.draw(Finish1);
        if (FinishGame.type == FinishGame.Closed)
        {
            Finish.close();
        }
    }
}
```

BAB VI

KESIMPULAN

Game Nazi Project : Tank merupakan game yang sangat unik dan menyenangkan, karena baik dari desain map sampai dengan sistem scoringnya memiliki feel yang tidak akan dimiliki oleh game lain. mulai dari menembak musuh, menghindari peluru, merebutkan airdrop dan rasa tegang yang diberikan saat melihat skor yang memiliki banyak misteri.

Dari pembuatan game ini mahasiswa menjadi lebih terasah karena tidak hanya dari mengasah kompetensi dalam pemrograman namun juga membantu mahasiswa yang bosan dalam mempelajari Pemrograman Orientasi objek. adapun SFML yang digunakan pada game ini yang bertujuan untuk memudahkan membuat objek.