C++-ohjelman osat sijoitetaan eri muistialueille sen mukaan, millaisia muistin hallintaan liittyviä tarpeita niillä on. Alla on lyhyt kuvaus eri muistialueista ja niiden tyypillisistä käyttötarkoituksista.

### 1. \*\*Staattinen muistialue (Static Memory):\*\*

- \*\*Sisältö:\*\*

- Staattiset muuttujat (kuten globaalit muuttujat ja staattiset luokkamuuttujat).

- Staattiset vakiot.

- \*\*Miksi:\*\*

- Staattinen muistialue varataan ohjelman käynnistyessä ja vapautetaan vasta ohjelman päätyttyä.

- Sopii muuttujille, jotka säilyvät ohjelman koko suorituksen ajan.

```cpp

// Esimerkki staattisesta muuttujasta

static int globalVariable = 42;

```

### 2. \*\*Pino (Stack):\*\*

- \*\*Sisältö:\*\*

- Funktioiden paikalliset muuttujat.

- Funktioiden paluuarvot.

- Funktioiden kutsutiedot.

- \*\*Miksi:\*\*

- Pinossa sijaitsevat muuttujat liittyvät suorituksen aikaiseen funktion suoritukseen.

- Pino kasvaa ja pienenee automaattisesti funktion kutsun ja paluun myötä.

```cpp

// Esimerkki paikallisesta muuttujasta

void exampleFunction() {

int localVar = 10; // Paikallinen muuttuja pinoon

}

```

### 3. \*\*Dynaaminen muistialue (Heap):\*\*

- \*\*Sisältö:\*\*

- Dynaamisesti varatut muuttujat ja tietorakenteet (`new`, `malloc`).

- \*\*Miksi:\*\*

- Varataan ja vapautetaan ohjelman suorituksen aikana tarpeen mukaan.

- Soveltuu tilanteisiin, joissa muuttujien koko ei ole tiedossa käännösaikana.

```cpp

// Esimerkki dynaamisesti varatusta muuttujasta

int\* dynamicVar = new int;

```

### 4. \*\*Koodialue (Code Segment):\*\*

- \*\*Sisältö:\*\*

- Käännösaikana generoidun konekoodin (ohjelmakoodi).

- \*\*Miksi:\*\*

- Koodialueelle sijoitetaan itse ohjelman suoritettava koodi.

- Koodi on vain luettavissa eikä siihen voi kirjoittaa.

```cpp

// Esimerkki ohjelmakoodista

int main() {

// Ohjelman suoritettava koodi

return 0;

}

```

### 5. \*\*Vapaamuotoinen muistialue (Free Store):\*\*

- \*\*Sisältö:\*\*

- C++ `new` ja `delete`-operaattoreilla varatut muuttujat.

- \*\*Miksi:\*\*

- Vapaamuotoinen muistialue on osa heapista, mutta se erottuu siitä käytön ja vapauttamisen tyylistä.

- Tämä alue sisältää myös dynaamiset muuttujat, mutta niiden hallinta voi olla vapaamuotoisempaa kuin perinteisessä heapissa.

```cpp

// Esimerkki vapaamuotoisesti varatusta muuttujasta

int\* freeStoreVar = new int;

```

Yleensä ohjelman suoritusaikaan liittyvät muistialueet (stack, heap) ovat dynaamisia ja hallitaan automaattisesti, kun taas käännösaikana liittyvät muistialueet (staattinen muistialue, koodialue) ovat staattisia. On tärkeää ymmärtää näiden muistialueiden roolit ja käyttötapaukset ohjelman suunnittelussa ja toteutuksessa. Muistin hallintaan liittyvien virheiden välttämiseksi on myös tärkeää vapauttaa dynaamisesti varattu muisti asianmukaisesti.