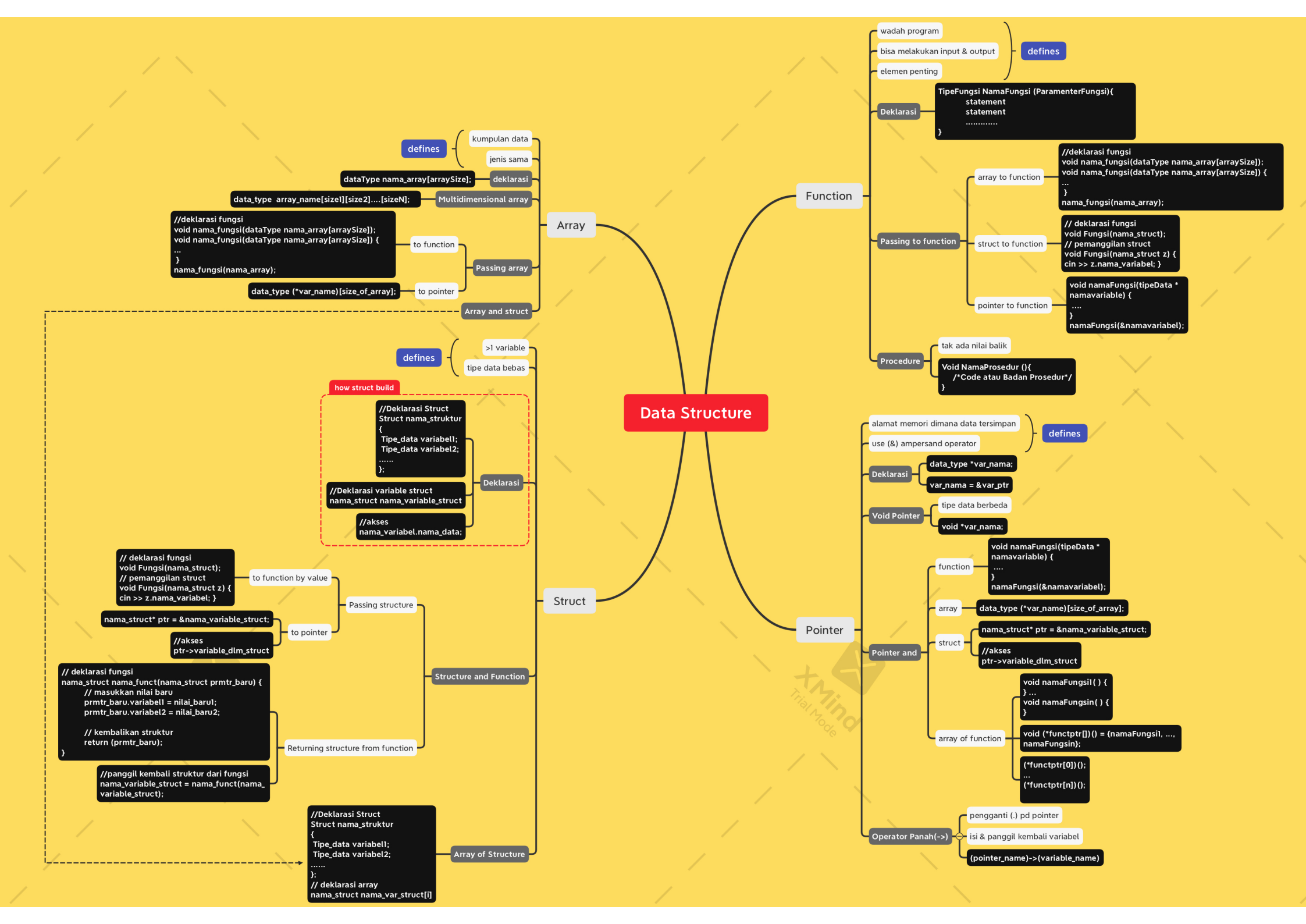
Data Structure
Array
defines
kumpulan data
jenis sama
deklarasi
dataType nama_array[arraySize];
Multidimensional array
data_type array_name[size1][size2]....[sizeN];
Passing array
to function
//deklarasi fungsi
void nama_fungsi(dataType nama_array[arraySize]);
void nama_fungsi(dataType nama_array[arraySize]) {
...
}
nama_fungsi(nama_array);
to pointer
data_type (*var_name)[size_of_array];
Array and struct
Struct
defines
>1 variable
tipe data bebas
how struct build
Deklarasi
//Deklarasi Struct
Struct nama_struktur
{
Tipe_data variabel1;
Tipe_data variabel2;
.....
};
//Deklarasi variable struct
nama_struct nama_variable_struct
//akses
nama_variabel.nama_data;
Passing structure
to function by value
// deklarasi fungsi
void Fungsi(nama_struct);
// pemanggilan struct
void Fungsi(nama_struct z) {
cin >> z.nama_variabel; }
to pointer
nama_struct* ptr = &nama_variable_struct;
//akses
ptr->variable_dlm_struct
Structure and Function
Returning structure from function
// deklarasi fungsi
nama_struct nama_funct(nama_struct prmtr_baru) {
// masukkan nilai baru
prmtr_baru.variabel1 = nilai_baru1;
prmtr_baru.variabel2 = nilai_baru2;
// kembalikan struktur
return (prmtr_baru);
}
//panggil kembali struktur dari fungsi
nama_variable_struct = nama_funct(nama_
variable_struct);
Array of Structure
//Deklarasi Struct
Struct nama_struktur
{
Tipe_data variabel1;
Tipe_data variabel2;
.....
};
// deklarasi array
nama_struct nama_var_struct[i]
Function
defines
wadah program
bisa melakukan input & output
elemen penting
Deklarasi
TipeFungsi NamaFungsi (ParamenterFungsi){
statement
statement
.....
}
Passing to function
array to function
//deklarasi fungsi
void nama_fungsi(dataType nama_array[arraySize]);
void nama_fungsi(dataType nama_array[arraySize]) {
...
}
nama_fungsi(nama_array);
struct to function
// deklarasi fungsi
void Fungsi(nama_struct);
// pemanggilan struct
void Fungsi(nama_struct z) {
cin >> z.nama_variabel; }
pointer to function
void namaFungsi(tipeData *
namavariable) {
....
}
namaFungsi(&namavariabel);
Procedure
tak ada nilai balik
Void NamaProsedur (){
/*Code atau Badan Prosedur*/
}
Pointer
defines
alamat memori dimana data tersimpan
use (&) ampersand operator
Deklarasi
data_type *var_nama;
var_nama = &var_ptr
Void Pointer
tipe data berbeda
void *var_nama;
Pointer and
function
void namaFungsi(tipeData *
namavariable) {
....
}
namaFungsi(&namavariabel);
array
data_type (*var_name)[size_of_array];
struct
nama_struct* ptr = &nama_variable_struct;
//akses
ptr->variable_dlm_struct
array of function
void namaFungsi1( ) {
} ...
void namaFungsin( ) {
}
void (*functptr[])() = {namaFungsi1, ...,
namaFungsin};
(*functptr[0])();
...
(*functptr[n])();
Operator Panah(->)
pengganti (.) pd pointer
isi & panggil kembali variabel
(pointer_name)->(variable_name)
XMind
Trial Mode