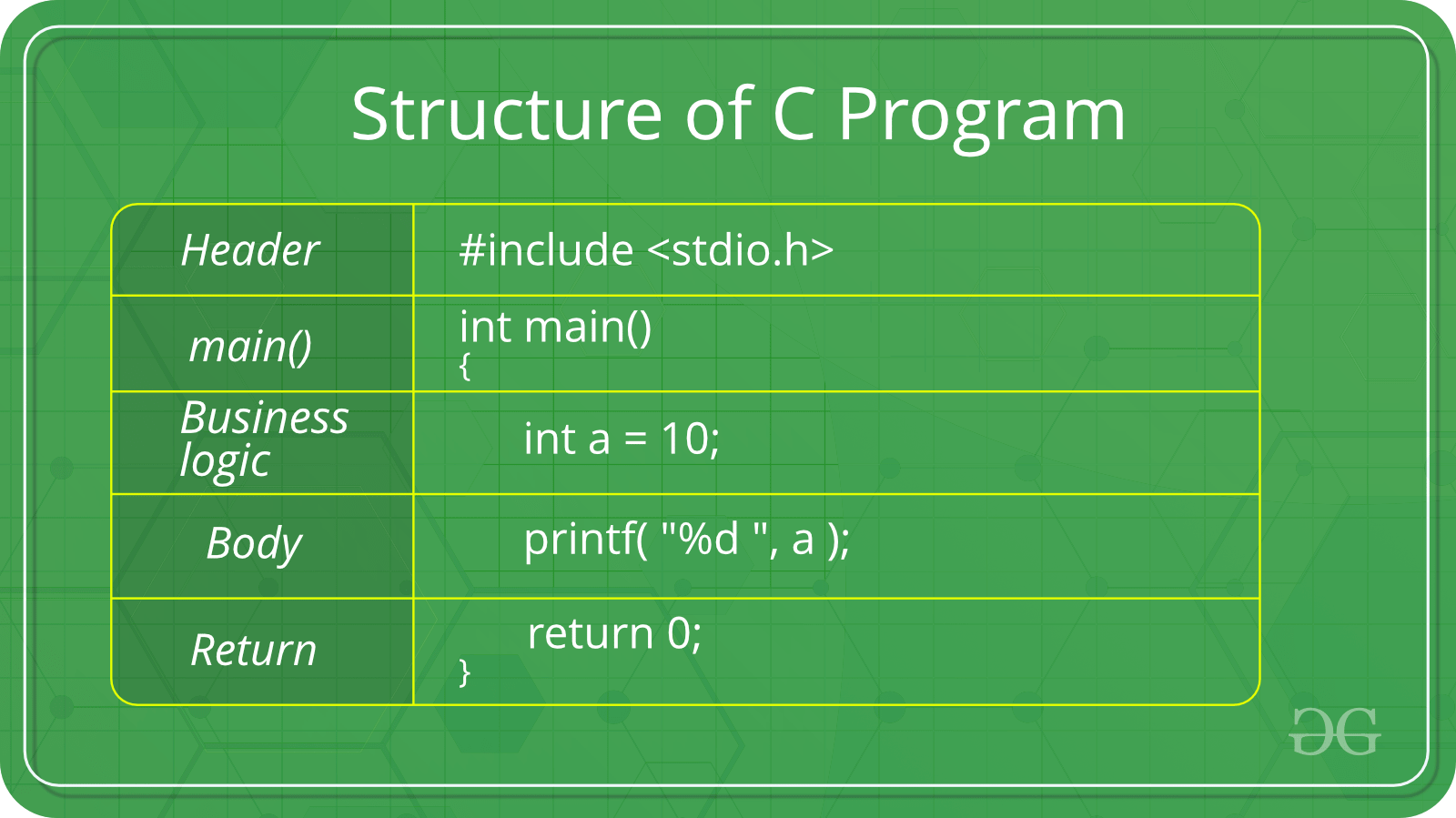
შესავალი

C არის სისტემური პროგრამირების ენა, საშუალებას გვაწლევს სტრუქტურულად ვწეროთ კოდი. მის შემქმენლად მოიხსენიებია დენის რიჩი 1969 და 1973 წლების პერიოდშ იქნემნებოდა აღნიშნული პროგრამირების ენა. მისი მთავარი მიზანია ოპერციული სისტემების დაწერა. ერთერთი მთავარი ფაქტორია ის რომ C არის Low-Level -ი ანუ პირდაპირი წვდომა გვაქვს მეხსიერებებთან. ეს ფაქტორი ხდის C ენას ოპერაციული სისტემების შემქნელად.

კოდის სტრუქტურა



პირველი Header ნაწილი არის ბიბლიოთეკა,standard input output

Main ნაწილი ანუ მთავარი ფუნქცია საიდანაც იწყება კოდი, აქედან ხდება კოდის გაშვება.

Business logic ცვლადები რომლებიც გამოყენებულნი არიან ფუნქციაში.

body აქ ხდება ჩვენთვის სასურველი კოდის შექმნა. მანიპულაცია, დალაგება, დაბეჭჭვდა და ა.შ.  
return ეს არის დასაბრუნებელი ტიპი. როდესაც პროგრამა დამთავრდება იგი მიიღებს რაიმე მნიშვნელობას.

#include <stdio.h>

int main(void)

{

printf("Hello C");

return 0;

}

განვიხილოთ კოდი: პირველ ხაზზე არის ბიბლიოთეკის დეკლარირება, ბიბლიოთეკა შეიცავს ფუნქციებს და იქ აღწერილი ფუნქციები შეგვიძლია გამოვიყენოთ ჩვენს კოდში.

# სიმბოლო აღნიშნავს გაშვებამდე preprocessor მიწოდებას კომპილერისთვის.

Int main (void){} მთავარი ფუნქცია რომელიც დაკომპილირების და გაშვების შემდეგ პროგრამა იწყება აღნიშნული ფუნქციიდან. Int არის ტიპი რომელიც მთავარ ფუნქციას ხდის მნიშვნელობის მქონეს, როდესაც მთავარი ფუნქცია დასრულდება ის მიიღებს რაიმე მნიშვნელობას. Void ნიშნავს რომ ეს ფუნქცია არ აბრუნებს არაფერს, არც გადაეცემა რაიმე პარამეტრი. ამაზე მერეც ვისაუბრებთ. Main არის სახელი მთავარი ფუნქციის. {} ფიგურული ფრჩხილები არის საზღვრები სადაც უნდა მოვათავსოთ ჩვენი პროგრამა. გამოიყენებიან ფუნქციებთან , ციკლებთან პირობით ოპერატორებთან, რომ შემოსაზღვრონ მათი სივრცე კოდში, მასში რაც ხდება მრჩება მასშივე და სხვა საზღვრებში არ გადადის თუ მის გარეთ არაა რაიმე აღწერილი რასაც შემდეგ მასში გამოვიყენებთ. Printf(“printing”); ფუნქციები არიან კოდის ნაწილები რომლებიც გამოიყენებიან იმ შემთხვევაში თუ რაღაც კოდის ნაწილი რამდენიმეჯერ გვინდა გამოვიყენოთ ამისთვის ერთიდაიგივე კოდის წერას რამდენიმეგან ხდება ფუნქციით შემოსაზღვრა და ყოველ საჭირო ადგილზე ფუნქციის დაძახება. ამარტივებს ასევე კოდის გააზრებას და უფრო მარტივად წაკითხვად ხდის. მაგ printf ფუნქცია მოიცავს კოდის ნაწილს და რამდენჯერაც დაბეჭდვა დაგვჭირდება ყოველ ჯერზე რომ კოდის ის ნაწილი ვწეროთ მეტად აირევა, გადიდდება და გართულდება პროგრამა. აღნიშნული ფუნქცია კონსოლზე ბეჭდავს მის შიგნით „“ ბრჭყალებს შორის მოთავსებულ სტრიქონს. მას დეტალურად კიდევ განვიხილავთ. Return 0; როგორც ზემოთ უკვე ვახსენე Int main (void ){} არის ფუნქცია პირველი ინტ პარამეტრი ნიშნავს იმას რომ ფუნქცია დასრულების შემდეგ მიიღებს პარამეტრს რომელიც იქნება ინტ ტიპის ხოლო ინტ ტიპში შეიძლება იყოს მოთავსებული ნატურალური რიცხვები რომლებიც ჩაეტევიან 4 ბაიტში. დაახლოებით 4 მილიარდი. უფრო დიდ რიცხვს მასში ვერ ჩავწერთ. პროგრამას გაშვებამდე სჭირდება მანქანურ ენაზე გადათარგმნა რომ გაიგოს პროცესორმა რა მითითებები შეასრულოს. ამისათვის გვჭირდება პროგრამა რომეიც გადაიყვანს დაწერილ კოდს ორობით ანუ მანქანურ ენაზე გადათარგმნის შემდეგ მოგვცემს დაკომპილირებულ ფაილს რომელიც იქნება უგკვე გაშვებადი და მისი გაშვებია ნიშნავს ჩვენი კოდის ამუშავება შესრულებას. ლინუქს აქვს ჩაშენებული, ხოლო ვინდოუსზე დაყენება იქნება საჭირო. მაგ mingw ან რომელიმე მზა IDE რომელშიც ასევე შეგვიძლია ვწეროთ პროგრამა. გვაქვს სხვადასხვა საშუალება უბრალო ტექსტ ფაილში ვწეროთ კოდი, ამას სჭირდება შემდეგ სასურველი გაფართოვება .c დაბოლოვებით რომ კომპილატორი მიხვდეს რასთან აქვს საქმე. მეორე ვარიანტია უკვე მზა პროგრამირების გარემოები devcpp, ms visual studio, codeblocks, eclipse და სხვა უამრავი რომელბიც გვიმარტივებენ კოდის წერას მაგ თავისით აბოლოვებენ სიტყვებს და გვთვაზობენ ალტერნატივას მაგ ბრჭყალების ჩასმას და ა.შ. საწყის ეტაპზე აჯობებს ლინუქსზე gedit ჩაშენებული ტექსტური ედიტორი გამოვიყენოთ ან ვინდოუსზე notepad ++. ყველაფრის ხელით წერა მოგვიწევს და უფრო უკეთესად გავიგებთ, მივეჩვევით კოდს.