Date: / / Page No.:

# Recursion - Function calling itself is called recursion

```
main()
{
  int k;
  k = f1(3);
  printf("%d", k);
}
int f1(int n)
{
  int s;
  if (n==1)
  return 1;
  s = n + f1(n-1);
  return s;
}
```

- Every recursive solution can be replaced by itarative solution.
- recursion का code छोटा होता है, इसका logic सोचना बहुत आसान है, इसलिए itaration से ज्यादा सही है इसे use करना
- सारी problems, recursion से नहीं बनती
- Recursive case (RC)
- Base case (BC)

here main() is not a recursive function

f1 is a recursive function

Ass. Tips →
1. Function name
2. Recursive case
3. Base case

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:39

# Program's memory

Date: / / Page No.:

main()

| K | |
|---|---|
| 6 | int K; |
| | K = f1(3); |
| | Printf ("%d", K); |

3

int f1(int n)

6

| n | |
|---|---|
| 3 | int S; |
| | if (n==1) |
| | return 1; |
| S | |
| 6 | S = n+f1(n-1); |
| | return S; |

2

int f1(int n)

3

| n | |
|---|---|
| 2 | int S; |
| | if (n==1) |
| | return 1; |
| S | |
| 3 | S = n1 + f1(n-1); |
| | return S; |

1

1

int f1 (int n)

| n | |
|---|---|
| 1 | int S; |
| | if (n==1) |
| | return1; |
| S | |
| | S = n1 + f1(n-1); |
| | return S; |

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:40

```
main()
{ int k;
  k = f1(3);
  printf("%d", k);
}

int f1(int n)
{ int s;

  if (n==1)
  return 1;        } BC
  s = n + f1(n-1);
  return s;        } RC
}
```

f1(3)
3 + f1(2)
2 + f1(1)
1

f1(10)
10 + f1(9)
9 + f1(8)

Sum(n)
n + sum(n-1)

Base case - वो situation जब function खुद को call करना बंद कर देगा

in a recursive function, make sure की वो कहीं पर खत्म हो रहा है, और जहाँ खत्म हो रहा है वही Base case है, you have to always think about this base case

Recursive case - this is our main formula to solve the problem.

recursion से किसी problem को solve करना है तो उसमे एक ऐसी लाइन लिखनी पड़ेगी जिसमे तुम उसी function को दोबारा call कर रहे होगे, लेकिन अब जब call कर रहे हो तो original problem से थोड़ा simpler version के लिए call करो। जैसे 10 number का निकालने से तो simple है 9 number का sum निकालना, [illegible] का कैसे निकालना है वो recursion पर छोड़ दो

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:40

Date: / / Page No.:

- अब add मे हम जो भी values लिख देंगे, वो चली जाएगी a और b मे e.g. add (10, 20); लिखने पर पहली value a मे जाएगी दुसरी value b मे जाएगी और उसका ~~some~~ sum हो जाएगा, last मे output आएगा sum is 30.

- या हो सकता है main function के अंदर दो variables हो उसके अंदर कोई value रखी हो और उन variables को हम add के parenthesis मे लिखे।

```
int x = 10 , y = 20;
add (x, y);
```

- या ये भी हो सकता है कि main function के अंदर हम user से values enter कराना चाहे तो वो भी कर सकते है

```
printf ("Enter two numbers");
scanf ("%d%d", &x, &y);
add (x, y);
```

- ये भी हो सकता है कि user से ना लेकर, कोई calculation कर रहे है उसका result आया है वो x मे रखे या y मे रखे, फिर उनको add करना है

main function मे तुम x और y मे data कैसे ला रहे हो इससे add function को रत्ती भर भी फर्क नही पड़ता

# add function को जो कोई call करेगा ना, वो जिन data को add कराना चाहता है वो data कहा से आएगा वो वही जाने means यह पर main function को ही लेना देना है इस बातसे।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:38

add function, takes something natur का हम बना रहे है, तो हमे दो value मिलेगी, और उन्हे वर्क करना हमारा काम है।

```
# include <conio.h>
# include <stdio.h>
void add (int, int);

main()
{
 ~~int~~ add (10,20);
      getch();
}
void add (int a, int b)
{
 int c;
 c=a+b;
 printf ("Sum is %d",c);
}
```

```
# include <conio.h>
# include <stdio.h>
void add (int, int);
main()
{ printf ("Enter two numbers");
 scanf ("%d%d", &x, &y);
 add (x, y); // Actual Arguments
 getch();
}
void add (int a, int b) // Formal Argument
{ int c;
 c = a+b;
 printf ("sum is %d", c);
}
```

Argument / Parameters

Actual arguments

Formal Arguments

```
# include <conio.h>
# include <stdio.h>
void add (int, int);
main()
{ int x = 10, y=20
 add (x, y);
 getch();
}
void add (int a, int b)
{ int c;
  c = a+b;
 printf ("sum is %d", c);
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:39

Date: / / Page No.:

Actual arguments ⇒ function को call करते समय जो paranthesis मे लिखते है उसे कहते है actual arguments

e.g. add (x, y);

Formal arguments ⇒ जब function define करते है और paranthesis मे जो variable बनाते है उसे कहते है Formal arguments.

e.g. void add (int a, int b)

- main function मे variables के नाम x, y है और add function मे a और b रखा है, अगर हम चाहे तो दोनो जगह same नाम के variable रख सकते है, इससे कोई फ़र्क नही पड़ता, लेकिन ये दोनो variables physically अलग - अलग होंगे -

main function के variable अलग है और add के अलग है

main के पास दो variables है, add के पास दो variable है, main के variable का data, add के variable मे जाएगा तो ये बिलकुल भी जरूरी नही है कि इन दोनो functions के variable का नाम same रखा जाए।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:39

Date: / / Page No.

3. Takes Nothing, Return something

e.g. int add ( )

takes nothing natur का है तो a और b की values user से लेंगे
takes something में user से data नहीं लेना होता है।

यहाँ add की body मे c की value print नही करवाना है क्योंकि हम उसे return करेंगे (return somthing natur)

अभी तक print इसलिए करा रहे थे क्योंकि returns nothing natur का था और function के खत्म हो जाने पर c की value destroy हो जाती जब return something natur का function है, तो इसका मतलब ये है कि जिसने function को call किया था, उसके पास value return होकर पहुंच जाएगी

value return होने के बाद वो function (e.g. main()) इसका जैसे चाहे use कर सकता है।

जो add से value return होकर आ रही है उसे हम s मे assign करवा लेंगे अब इसे हम चाहे तो print भी करवा सकते हैं।

```
# include <conio.h>
# include <stdio.h>
  int add();
  main()
{
  int s;
  s = add();
  printf("Sum is %d", s);
  getch();
}
  int add()
{
  int a, b, c;
printf("Enter two numbers");
scanf("%d%d", &a, &b);
  c = a+b;
  return c;
}
```

# return keyword की मदद से केवल एक ही value return करवाया जा सकता है और return आगर हो उस function के अंदर तो वो function उसी वक्त end हो जाता है

Date: / / Page No.:

Takes something, returns something

```
# include <conio.h>
# include <stdio.h>
int add (int, int);
main ()
{
  int x, y, s;
  printf ("Enter two numbers");
  scanf ("%d %d", &x, &y);
  s = add (x, y);
  printf ("sum is %d", s);
  getch ();
}
int add (int a, int b)
{
  int c;
  c = a+b;
  return c;
}
```

Assignment quations - Assignment 9

in assignment copy

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:39

Date: / / Page No.:

जितने भी predefine function है उनका declaration पहले से किसी file मे लिखा जा चुका है और इस file को बोलते है header file.

getch() का declaration ⇒ conio.h file मे है

# include < conio.h >

# → Preprocessor director
include → command

जैसे ही Program build होता है सबसे पहले preprocessor नाम का प्रोग्राम चलता है और ये आपके Program मे सिर्फ उसी line को देखता है जिसमे # लगा होता है शुरुवात में।

include is a command for preprocessor program

# include <conio.h> लिखने से preprocessor program समझ जाएगा की conio.h ( full form- consol input output) के अंदर जो लाइनस लिखि है उन सबको इस प्रोग्राम के अंदर include करवाना है। e.g. Preprocessor इस एक statment को हटाकर conio.h की सारी-सारी lines लाकर रख देगा

जब compiler इस प्रोग्राम को read करेगा तो उसे #include<conio.h> नही दिख रहा क्योंकि उसकी जगह पर तो बहुत सारी lines आ चुकी है ये lines declaration है कुछ predefine functions की जिनमे से getch भी एक है, तो यहाँ से getch() का declaration compiler को मिल जाता ~~getch का~~ declaration हमेशा केवल compiler के लिए होता है, ये execute नही होता है

Date: / / Page No.:

जितने भी predefine function है उनका declaration पहले से किसी file मे लिखा जा चुका है और इस file को बोलते है header file.

getch() का declaration ⇒ conio.h file मे है

# include < conio.h >

# → Preprocessor director
include → command

जैसे ही program build होता है सबसे पहले preprocessor नाम का प्रोग्राम चलता है और ये आपके program मे सिर्फ उसी line को देखता है जिसमे # लगा होता है शुरुवात में।

include is a command for preprocessor program

# include < conio.h > लिखने से preprocessor program समझ जाएगा की conio.h ( full form- consol input output) के अंदर जो लाइनस लिखि है उन सबको इस प्रोग्राम के अंदर include करवाना है। e.g. preprocessor इस एक statment को हटाकर conio.h की सारी-सारी lines लाकर रख देगा

जब compiler इस प्रोग्राम को read करेगा तो उसे #include<conio.h> नही दिख रहा क्योंकि उसकी जगह पर तो बहुत सारी lines आ चुकी है ये lines declaration है कुछ predefine functions की जिनमे से getch भी एक है, तो यहाँ से getch() का declaration compiler को मिल जाता है declaration हमेशा केवल compiler के लिए होता है, ये execute नही होता है

Date: / / Page No.

printf(), scanf() का declaration → stdio.h file में है

तो हम इनके declaration के लिए # include <stdio.h> लिखते हैं।

- C language के standerdisation के time इसके documents में लिखा हुआ है कि function declaration is recommended but not mandatory.

- Header file के अंदर function की definition नहीं होती सिर्फ function का declaration होता हैं।
  definition, library file में होती हैं
- main() को call ही नहीं कर रहे तो declaration की जरूरत भी नहीं है

~~Takes - Some~~

```
# include<conio.h>
# include <stdio.h>
void add();
main()
{ add();
  getch();
}
void add()
{ int a,b,c
  printf ("Enter two numbers");
  c=a+b;
  printf ("Sum is %d", c);
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:38

Date: / / Page No.:

2. Takes something returns nothing

अब हम add function बनाएंगे जिसका natur है takes something returns nothing.

अब returns nothing है तो void add ....

और takes something है तो void add(.....)

जब भी takes something natur का function बनाते है तो paranthesis मे हमें variables का declaration करना होता है।

कितने variables का ?

- इस फंक्शन (add) को अपना काम करने के लिए कितनी values की जरूरत हैं, जितनी values की जरूरत है उतने variables का declaration करेंगे।

void add ( int a , int b )

variables हमें यहाँ तीन लग रहे है पर values तो दो ही चाहिए।

इसके अंदर हर variable के लिए data type अलग लिखना होता है

अब a, b हमने paranthesis मे declair कर दिया है तो इसे definition के अंदर लिखने की जरूरत नही है और इस condition मे भी a, b, c local variables ही है

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:38

अब अगर a और b मे हमे तो data मिल चुका है, जिसे हमे add करना है, तो फिर हमे printf, scanf लिखने कर user से दुबारा a और b की value मांगने की जरूरत ही नहीं है।

Takes something natur का program है मतलब a और b में data आ चुका है, कहां से आया - जब call हुआ तब आया

और we know की function definition की पहली line ही function का declaration होती है with ;, declaration करते समय variable का नाम लिखना जरूरी नही है लिख देंगे तो भी कोई Problem नही है void add (int, int); Or void add(int a, int b)

अब जब function call करते है तभी function की defination चलती है तो function call करेंगे तो paranthesis खाली नहीं रख सकते

printf को call करते तब paranthesis भरे रहते है, क्योंकि printf function take something natur का define है।

getch — " — खाली रहते — " —
getch takes nothing natur का है।

यहाँ add, takes something natur का है।

```
# include <conio.h>
# include <stdio.h>
void add (int, int);
main ()
{ add (—, —);
  getch();
}
void add (int a, int b)
{ int c;
  c = a + b;
printf ("Sum is %d", c);
}
```

```
main()
{
  add();
  getch();
}

add()
{
  int a, b, c;
  printf ("Enter two numbers");
  scanf ("%d %d", &a, &b);
  c = a+b;
  printf ("Sum is %d", c);
}
```

here

add(); # function call

# function definition

# Function declaration

variable declaration ⇒ a, b, c are new words for compilier until we write declaration statment. declaration statment is always for compiler, it never execute. ( int a, b, c; )

इसे लिखने के दो तरीके हो सकते हैं - या तो add function के अंदर लिखो या तो add function के बाहर लिख दो।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

```
main()
{
  add();
  getch();
}
int a,b,c;  // global variables

add()
{
  printf("Enter two numbers");
  scanf("%d%d", &a, &b);
  c = a+b;
  printf("sum is %d", c);
}
```

```
main()
{
  add();
  getch();
}

add()
{
  int a,b,c  // local variable
  printf("Enter two numbers");
  scanf("%d%d", &a, &b);
  c = a+b;
  printf("Sum is %d", c);
}
```

**global variable** ⇒ ये variables function के end होने पर destroy नहीं होंगे / relese नहीं होंगे बल्की जब program end होगा तब relies होंगे।

the life time of globle variable - till the end of the program they will remain in the memory till the end of the program

**local variable** ⇒ ये variable का life time और scop केवल उस function की body ही रहती है जिस function के अंदर variable declair किया है। इस function के खत्म होते ही a, b, c की memory release हो जाएगी।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

We should prefer local variable because globle variable मेमोरी occupie करके रखते है through out the program. program को run होने के लिए memory limited मिलती है।

- We can assume global variable as a shared container. all the functions are performing different-different operations but they can use that shared variable any time.

global variable बनते है, पर हम इसे last preference देंगे क्योंकि वो मेमोरी ज्यादा occupie कर रहें हैं।

It is not a good idea to make a variable global unnecessa

---

अब जैसे variables का declaration करना होता है वैसे ही functions का भी declaration करना होता है

functions भी नए शब्द है जिनका अर्थ compiler को पहले से नहीं पता है। compiler have no idea about the word add. तो हम dectair करेंगे।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

अब से declare करना भी दो तरीके से हो सकता है
local declaration, global declaration

* Functions को memory तभी मिलती है जब वो call होते है। globally declair करने से memory पर कोई फर्क नहीं पड़ता।

declare कैसे करते है ?

declaration से पहले -
अगर हमारा function, returns nothing है, तो हमें इसके नाम से पहले इसका ~~writ~~ return type लिखना होता है।

return type - जो value return हो रही है उसका type (data type)
अगर int type की value return हो रही है तो int लिखो
अगर कुछ नहीं लिखा हुआ है तो भी int लिखा हुआ माना जाता है
अगर float typ की value return —"— float लिखो
अगर char —"— char लिखो
अगर कुछ भी value return नहीं हो रही है तो void लिखो

void नहीं लिखा होने पर भी function तो run कर रहा था पर लिखना ज्यादा accurate है, इसका मतलब है (void लिखने का) की function कुछ भी नहीं return कर रहा है।

और कुछ भी नहीं लिखा है इसका मतलब है int लिखा है। और warning मिलेगी की by default इसका return type int माना जा रहा है।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

As per the stander - we have to write void

```
main ()
{
    add();
    getch();
}
void add()
{
int a, b, c
printf ("Enter two numbers");
scanf ("%d %d", &a, &b);
c = a + b;
printf ("sum is %d", c);
}
```

function definition की पहली line को ; लगाकर लिखना ही function declaration है।

e.g void add ();

अब ये भी दो तरीके से किया जा सकता है

1) Local declaration
2) global declaration

```
main()
{ void add (); // local declaration
    add ();
    getch();
}
void add()
{
int a, b, c
printf ("Enter two numbers");
scanf ("%d %d", &a, &b);
c = a+b;
printf ("sum is %d", c);
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:38

Function declaration globally करना ज्यादा better है क्योंकि इससे पुरे प्रोग्राम के लिए ये declare हो जाएगा, ना केवल एक function के लिए। और memory का तो कोई issue है ही नही क्योंकि function को मेमोरी तभी मिलती है जब function call होता है।

```
void add ();  // global declaration
main ()
{
   add ();
   getch();
}
   void add ()
{
  int a, b, c
  printf ("Enter two numbers");
  scanf ("%d %d", &a, &b);
  c = a+b;
  printf ("sum is %d", c);
}
```

add () user defined function था जिसका हमने declaration किया।

अब predefine functions जैसे getch(), printf(), scanf() का भी तो declaration होना चाहिए, ये pridefine है इसका मतलब ये नही है कि compiler को पहले से मालुम है कि ये क्या है compiler को keywords पहले से पता होते है पर functions नहीं और ये दोनो अलग चीज़ हैं।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:38

Date: / / Page No.

# Ways to define a function

1. Takes Nothing, Return nothing
2. Takes something, Return nothing
3. Takes Nothing, Return something
4. Take something, Return something

1.

Takes nothing, Return nothing.

- कोई भी function किसी दुसरे function के variable को ~~exis~~ acces नहीं कर सकता। हर function सिर्फ अपनी body मे बने variable को acces कर सकता है।

एक function ने दुसरे function को call किया है तो दुसरा function पहले वाले functions के variables को acces नहीं कर सकता।

अगर कोई function दुसरे function को call करता है और हम चाहते है कि वो पहले वाले function के variables को acces कर लें।

जब ऐसी requirment होती है तब इस situation में takes वाला concept आता है

function को define करने के चार तरीके है

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.

## Ways to define a function

1. Takes Nothing, Return nothing
2. Takes something, Return nothing
3. Takes Nothing, Return something
4. Take something, Return something

1.

Takes nothing, Return nothing.

- कोई भी function किसी दुसरे function के variable को ~~exis~~ acces नहीं कर सकता। हर function सिर्फ अपनी body मे बने variable को acces कर सकता है।

एक function ने दुसरे function को call किया है तो दुसरा function पहले वाले functions के variables को acces नही कर सकता।

अगर कोई function दुसरे function को call करता है और हम चाहते है कि वो पहले वाले function के variables को acces कर लें। जब ऐसी requirment होती है तब इस situation में takes वाला concept आता है

function को define करने के चार तरीके है

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.

for adding two numbers.

instead of program make a function (named as add) and this function's work will be to add two numbers

```
add ()
{
    int a, b, c;
    printf ("Enter two numbers");
    scanf ("%d %d", &a, &b);
    c = a+b;
    printf ("Sum is %d", c);
}
```

लेकिन अगर हमे इसे run करना है, तो ये run नहीं होगा

यहाँ error show होगी Undefined reference to 'WinMain'

मतलब इसमे main function नहीं है, main function के बिना ये चल नहीं सकता।

तो सिर्फ इसलिए कि हम इस function को चलाकर देखना चाहते है हम main function बनाएंगे। otherwise हमारा function बन चुका है

ये जो function बना है ये takes nothing, returns nothing variety का है।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

अब पता कैसे चलता है कि takes nothing, returns nothing है ?

* function की body के अंदर return नाम का keyword नहीं लिखा है, तो it means, ये returns nothing natur का function है।

* function defination मे curly bracket के ऊपर जहाँ function का नाम लिखा हैं ( add() ) उसके parenthisis अगर खाली हैं, तो it means, ये Takes nothing natur का function हैं

# किसी भी function की defination को देखकर ये बताया जा सकता है कि takes nothing | takes something | returns nothing | returns something हैं।

अब जब हम main function बनाते हैं।

तब ये sum होगा।

हम main को पहले | बाद मे अपनी choice के according लिख सकते हैं।

```
add()
{ int a, b, c ;
  printf ("Enter two numbers");
  scanf ("%d%d", &a, &b);
  c = a+b ;
  printf ("sum is %d", c);
}
main()
{ add();
  getch();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

7. Function is way to achieve modularization.

modularization is splitting up a bigger task into several smaller subtasks to reduce the complexity of the bigger task.

8. Functions are of two types

- Predefined function → printf, scanf, getch, exit
- User defined function → main()

# Koi bhi function tab execute hota hai jab hum use call karte hain

```
main ()
{
    a();
    b();
    a();
}
void a ()
{
    printf ("Hello");
}
void b ()
{ printf ("Bye");
  a();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:35

Date: / / Page No.

# here we have define three functions

एक program के अंदर कितने भी functions define करो, इस बात से कोई फ़र्क नही पड़ता की आप उन्हे किस sequance मे define कर रहे है।

in e.g. main(), a(), b()
but we can a(), main(), b() // b(), a(), main()
The execution of our program will remain same.

program के अंदर main नाम का function होना जरूरी है, तभी प्रोग्राम run होगा लेकिन main सबसे पहले बना हो ये जरूरी नहीं हैं।

# here we have call → a(); , b(); , printf();
कोई भी function तब execute होता है जब हम उसे call करते है।

जब हम a को call करेंगे ( a(); ) तब a() execute होगा
जब हम b को call करेंगे ( b(); ) तब b() execute होगा
जब हम printf को call करेंगे (printf(); ) तब printf चलेगा

# But हम main को call नही कर रहे है, पर फिर भी तो चलता है। main function को operating system call करता है

when we execute the program, operating system calls main function firstly, then our program start, that's why we have to define main().

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:35

Date: __/__/__ Page No.:

जब भी program run होता है run होने के लिए उसे ram में कुछ memory मिलती है (suppose - Programs memory)

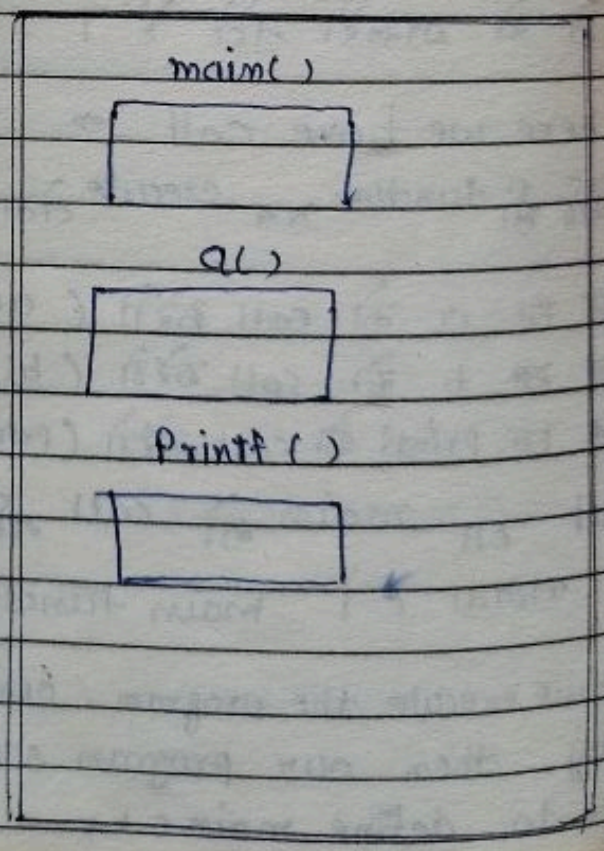
```
main()
{
  a();
  b();
  a();
}
void a()
{
  printf("Hello");
}
void b()
{
  printf("Bye");
  a();
}
```

- जब भी कोई function call होता है, चाहे हमने call किया हो या operating system ने। सबसे पहले वो RAM मे load होता है।

आपके program को run होने के लिए जो memory मिलि है उसी के अंदर कुछ हिस्सा (जगह), ये function occupie कर लेगा।

fog e.g. main function will ocupie some space

Program's Memory




- किसी भी function को मेमोरी दो कारणो के लिए मिलती है 1st) for storing the instruction 2nd) for variables

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:35

← control

Date: / / Page No.:

Now we have call the function main. So it's execution will start.

line 1 a(); ←

यहां a function call हुआ है, function जैसे ही call होता है, तो RAM मे load होता है।

now the control → a() ←

अभी main function की पहली line चल रही है और इसका मतलब है a function की पुरी defination का चलना।

Now in function a(), we have call function printf();, function जैसे ही call होता है, तो RAM मे load होता है,

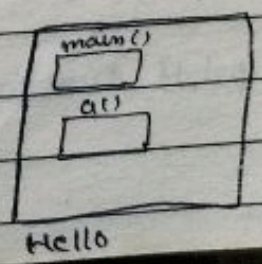
अब printf की defination जहां कही भी होगी तो execute होगी और इस execution से Hello print हो जाएगा।

printf function की coding finish हो गयी, code चल गया Hello print हो गया।

अब control exactly वही आएगा, जहां से function call हुआ था ( printf("Hello"); ← )

और printf function की memory relese हो गयी

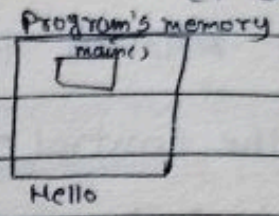
Program को जो memory मिली है उसमे से कुछ area consume हुआ है और बाकी free है। consumed and free का ratio change होता रहता है during execution of the program.




Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:36

Date: / / Page No.:

अब a function की भी सारी lines चल चुकी है, तो
अब control वहीं जाएगा जहा से function call हुआ था (a();)
और उस function (a()) की memory relieve हो जाएगी।




Now main function की 2nd line चलेगी

$II^{nd}$ line b(); ←

यहाँ function b call होगा जैसे ही call होगा, सबसे पहले RAM मे load होगा

now control b() ←

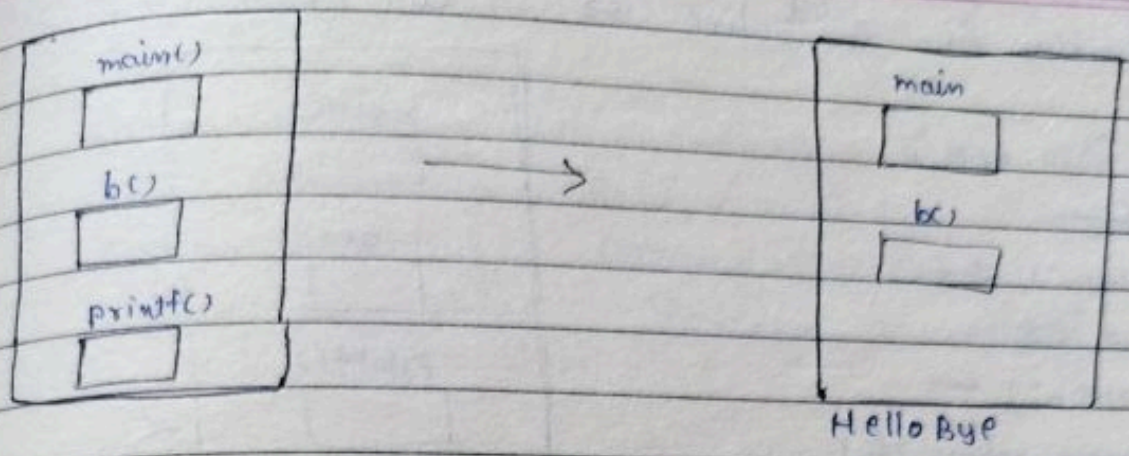
main function की 2nd line चल रही है इसका मतलब है b function की पुरी defination का चलना।

In b() → we have call printf(); , जैसे ही call होगा RAM मे load होगा printf की defination जहा कही भी होगी वो execute होगी और execution से Bye print हो जाएगा।

अब control exactly वापस वही आएगा जहाँ से function call हुआ था। ( printf("Bye"); ← )

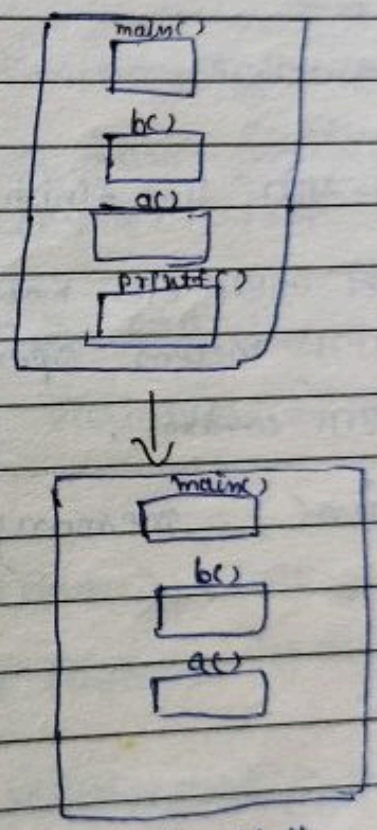
और printf function की memory relieve हो जाएगी।

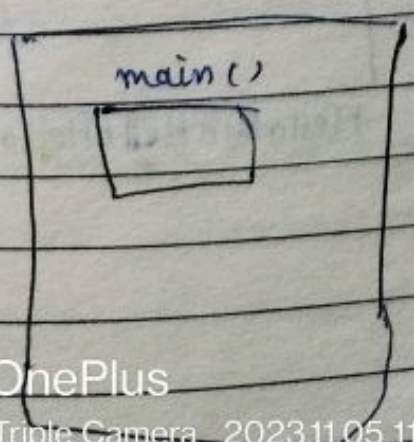
Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:36




अब b function की body मे दुसरी line चलेगी

```
a();
a()
printf ("Hello");
// Hello print होगा
printf ("Hello");
// printf() memory reliese
a की सारी line finish हो गयी
    a();
// a() memory reliese
b की सारी line finish हो गयी
    b();
// b() memory reliese
```




Hello Bye Hello

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:36

main function की III^rd line → a();

a load होगा RAM में

a() ←

printf("Hello"); ←

// Hello print होगा

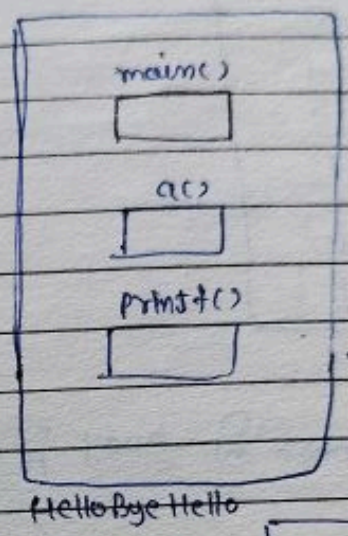
printf("Hello") ←

// printf memory reliese होगी।

a की सारी line चल ली है।

a(); ←

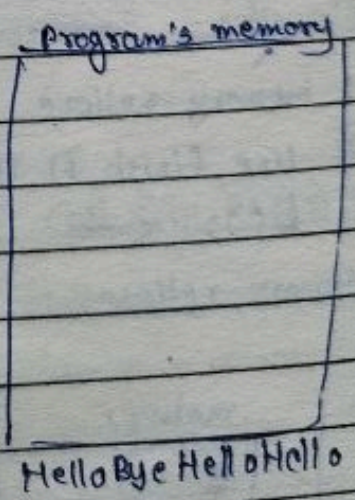
// a() memory reliese




main की भी सारी line finish हो चुकी हैं।

तो control वहाँ जाएगा जहा main function call हुआ था। means Operating System के पास जाएगा control.

और main() memory reliese




Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

## Arithmetic Instruction ⇒

An instruction which is used to manipulate data using operators is known as arithmetic instruction

arithmetic instruction = 3 + 4

here, + → operator

3, 4 → operand/Data

Types of operators according to the number of operand.

1) Unary operators
2) Binary operators
3) Ternary operator

in brif

1. Unary operators + , - , ++ , -- , sizeof
2. Arithmetic operators + , - , * , / , %
3. Bitwise operators & , | , ^ , ~ , >> , <<
4. Relational operators < , > , <= , >= , == , !=
5. Logical Operators ! , && , ||
6. Conditional operators ?:
7. Assignment operators = , += , -= , *= , %= , /=

Unary operators ⇒ In this operators there is only one operand, that is why this are called unary operators

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:23

Date: __/__/__ Page No:

+, - ⇒ these are sign operators, we give them first priority it means we solve it firstly.

examples ⇒ +5
-7

++ ⇒ Increment operator, in this, operand that we use is only can be a variable, it can't be a constant

++3; // error
++x; // correct

type ⇒

x++; post increment
++x; pre increment

output of the operator is same
x = x+1

output is same but the difference is priority

++x; pre increment first priority
x++; post increment last priority

priority means the sequence of applying operators as in BODMAS rull.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:23

```
main()
{
    int x = 5, y;
    y = x++;
    printf("%d %d", x, y);
}
```

try to dry run this programe

in third line of our program there are two operators =, ++ assign and post increment.

| X | Y |
|---|---|
| 5 | |

And as we give last priority to post increment, first we will operate assign operator

| X | Y |
|---|---|
| 5 | 5 |

asignment operator ⇒ by using it the value in the variable on the right side will be copied by the variable on the left side 5 = x++

Now after it post increment operator will be aplied the post increment operator's operant is x, so the value of x will be x+1.

| X | Y |
|---|---|
| 6 | 5 |

now output instruction printf will run and the values of x y will be print as our output

6, 5

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:23

Date: __/__/__ Page No.

-- ⇒ decrement operators

--x ; pre decrement
x-- ; post decrement } x = x-1

--x ; first priority
x-- ; last priority

---

Sizeof () ⇒ in () we write operant, operant can be
(1) data type
(2) variable
(3) constant

(1)
```
int x ;                    o/p
x = sizeof(float);         4
x = sizeof(char);          1
x = sizeof(int);           4
x = sizeof(double);        8
```

(2)
```
int x, y ;        x = sizeof(k);   o/p 4
float k;          x = sizeof(ch);  1
char ch;          x = sizeof(d);   8
double d;         x = sizeof(x);   4
                  x = sizeof(y);   4
```

# by default integer constant is int type.

(3)
```
int x ;
x = sizeof(35);     4
x = sizeof('A');    4
sizeof(5.67);       8
```

'A' = 65

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:24

# Arithmetic Operators

* / % (first priority)
\+ − (second priority)

in C language * / % operators have equal priority

if the instruction is 3 / 5 * 3 then what will happen

The operator which is in left side will be operated first.
It means / (divide) will be operat first.

| Examples | | first priority |
|---|---|---|
| | a/b * c | / |
| | a*b/c | * |
| | a+b*c | * |
| | a−b/c | / |
| | a−b+c | − |
| | a%b+c | % |
| | a+b−c | + |

other eg.

| | | | |
|---|---|---|---|
| 3+4 | 7 | 5 % 2 | 1 |
| 3−4 | −1 | 17 % 4 | 1 |
| 3*4 | 12 | 23 % 5 | 3 |
| 3/4 | 0 quotient | 25 % 5 | 0 |
| 3/4.0 | 0.75 | 3 % 5 | 3 |
| 3.0/4 | 0.75 | | |
| 3.0/4.0 | 0.75 | 2.5 % 2 | error |

(remaindar after division (modulo division))

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:24

Date: 16.12.21 Page No.

In C language

Q. If the output of $\frac{x}{5}$ is 2, than can you say this thing that x is completely divisible by 5.

Ans ⇒ No, x, 5 are integers (int type data) that's why output is also int type.
here x may be 11, 10, 12, 13, 14 the output will be 2.

Q. if the output of x % 5 is 0, than can you say this thing that x is completely divisible by 5.

Ans → Yes, output will be zero only when x is divisible by 5. x can be 10, 15, 5, 20, 25 ---- ∞

Q. What will be the output of x/10

x without last digit.

| | |
|---|---|
| 123/10 | 12 |
| 25612/10 | 2561 |
| 25/10 | 2 |

Q. What will be the output of x % 10

last digit of x

| | |
|---|---|
| 3521 % 10 | 1 |
| 23 % 10 | 3 |
| 3 % 10 | 3 |

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:24

2. write a program which takes day number of a week and display unique greeting message for the day.

```
main()
{ int x;
  printf ("Enter the day number");
  scanf ("%d", &x);

  switch(x)
  {
   case 1 :
    printf ("Happy Monday");
    break;
    case 2 :
    printf ("Happy Tuesday");
    break;
    case 3 :
    printf ("Happy wednesday");
    break;
    case 4 :
    printf ("Happy Thursday");
    break;
    case 5 :
    printf ("Happy Friday");
    break;
    case 6 :
    printf ("Happy Saturday");
    break;
    case 7 :
    printf ("Happy Sunday");

   default :
    printf ("invalid input");
  }
  getch();
}
```

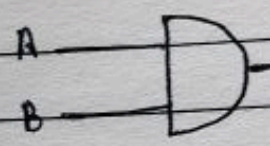
Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:34

# Bitwise Operators

| Operator | & | \| | ^ | ~ | >> | << |
|---|---|---|---|---|---|---|
| | and | or | x-OR | not | right shift | left shift |

**And**

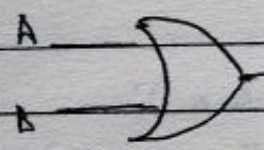
0 & 0 = 0
0 & 1 = 0
1 & 0 = 0
1 & 1 = 1




$Y = A \cdot B = A \& B$

**OR**

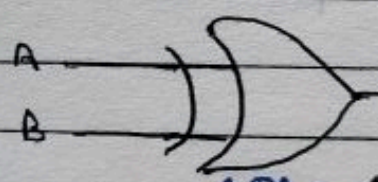
0 | 0 = 0
0 | 1 = 1
1 | 0 = 1
1 | 1 = 1




$Y = A+B = A|B$

**x-OR**

0 ^ 0 = 0
0 ^ 1 = 1
1 ^ 0 = 1
1 ^ 1 = 0

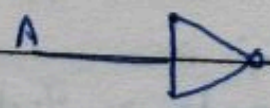



$Y = A \oplus B$

$A \oplus B = A \wedge B = \bar{A} \cdot \bar{B} + \bar{A} \cdot B$

**not**

~0 = 1
~1 = 0




$Y = \bar{A}$

$\sim A = B$
$\sim B = A$

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:24

2. Write a ~~program~~ menu driven program with following options:
a. Addition
b. Subtraction
c. Multiplication
d. Division
e. Exit

Date: / / Page No.

6.

```
main ()
{   int a, b, c, r ;
printf ("Enter two values");
scanf ("%d %d", &a, &b);
while (1)
{ printf ("Enter 1 for addition\n");
  printf ("Enter 2 for sub\n");
  printf ("Enter 3 for mult\n");
  printf ("Enter 4 for div\n");
  printf ("Enter 5 for exit");
  ~~printf~~ scanf ("%d", &c);
  printf ("%d", c);
switch (c)
{ case 1 :
  ~~prin~~ r = a+b;
  break;
  case 2 :
    r = a-b;
  break;
  case 3 :
    r = a*b;
  break;
  case 4 :
    r = a/b;
  break;
  case 5 :
    exit (0);
  default :
  printf ("invalid selection");
}
printf ("Result = %d\n", r);
getch ();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:34

Date: / / Page No.

# Binary Conversion

Trick ⇒ write the powers of two backwards (reverse)

512 256 128 64 32 16 8 4 2 1

अब हमे जिस नंबर का binary करना है, हम यह देखेंगे की वह नंबर 2 की पावर'स मे से किन-किन नंबरो का योग है।

जैसे 25 25 is the sum of 16, 8, 1

अब हमारा दिया गया नंबर जिन नंबर's का sum है, उनके नीचे हम 1, लिखेंगे और बाकी बीच मे या पीछे आ रहे numbers के नीचे 0 लिखेंगे।

| 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|
| | | | | 1 | 1 | 0 | 0 | 1 |

$(25)_{10} = (11001)_2$

| | 512 | 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|
| 61 | | | | | 1 | 1 | 1 | 1 | 0 | 1 |
| 92 | | | | 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| 83 | | | | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| 33 | | | | | 1 | 0 | 0 | 0 | 0 | 1 |

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:24

Date: / / Page No.:

# Functions

1. Function is block of code which has some name for identification

```
f1()
{
  code
}
```

← Function definition

2. Function names must be unique in program

3. Function definition, function call and function declaration are three different things.

```
f1();
```

← Function call

4. Any C program can have any number of functions, but minimum one function must be there.
If there is only one function in the program then its name must be main().

5. One function name must be main in the program. main() function is entry point of your program. program execution always begin from main function.

6. Function naming rules are same as of variables.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:34

practice

```
int x, y;
scanf("%d %d", &x, &y);
```

Scanf() मे अगर दो variable का use हम कर रहे है तो दो बार हम %d formate spacifier use करते है। formate spacifier के बीच मे हम (,) नही लगाते है आप चाहे तो space दे सकते है और आप चाहे तो (\n) भी लगा सकते है पर (,) कभी नही। then (,) लगाकर पहले variable का adress लिखे फिर (,) second variable का।

Now - पहले code मे हमने variable की value fix की हुयी थी अब हम चाहते है कि variable की value user के द्वारा inter की जाए।

```
main()
{
    int x, y;
    scanf("%d %d", &x, &y);
    printf(" %d + %d = %d", x, y, x+y);
    getch();
}
```

अब जब Program run होगा तब blank screen दिखेगी (क्योकि जैसे getch() के कारण screen रुक रही थी वैसे ही scanf() के कारण भी रुकेगी क्योकि ये ~~एक~~ दोनो input instruction है।

हमे जो data देना है वो हम type करेंगे फिर हम enter / space / tabe मे से कुछ भी दबाएंगे तब जाकर हमारा पहला data complete होगा और फिर दूसरा data लिखिये। आखिरी data के बाद enter press करना होगा।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:23

# Output instruction

जो normal user होगा, जिसको हमारा code नहीं दिख रहा है और उसे समझ मे भी नहीं आता है, उसके पास के सिर्फ हमारी .exe file है।

तो वो blank screen को देखकर confuse होगा। उसके लिए हम एक mesege print कराते हैं। जिसके लिए हम code मे एक लाइन printf() {Pridefined function} output instruction लिखेंगे।

```
main()
{
  int a,b ;
  printf (" Enter two numbers") ;
  scanf ("%d %d", &a, &b);
  printf ("%d + %d = %d", a , b , a+b);
  getch () ;
}
```

Now जब program run होगा तब सबसे पहली instruction (output) चलेगी और हमे screen पर वह message लिखा हुआ दिखेगा जिसे हमने printf() मे लिखा है।

If we want third variable then we can write

```
main()
{
  int a, b, c ;
  printf (" Enter two number") ;
  scanf ("%d %d", &a , &b) ;
  c = a+b ;
  printf ("Sum is %d", c) ;
  getch ()
}
```

or printf ("%d + %d = %d", a,b

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:23

Input instruction

scanf() predefined function

scanf() की help से keyboard से data जो इंटर होता है उसे हम एक variable मे पहुचाते है।

# getch() की help से हम एक charector देते है। but

# scanf() की help से हम एक data देते है। जो कि एक से ज्यादा keys से मिलकर बनता है।
जैसे आपकी age 19

⇒ scanf() की help से data input लेना और उसे variable मे लाना।

```
int x ;
scanf ("%d" ,&x );
```

here we can use only formate spacifier only in double codes. not other message is allow as we use in printf()

& ⇒ adress of (&), it is an m persent symbol

we use it in only scanf(), not in printf()

scanf() से जो key press की जाती है वो screen पर दिखते है
getch() से press की गयी key दिखाई नही देती screen पर

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:22

Switch case, goto control

1. the constant after the case can not be same for two. we can say that duplicate case is not possible.
Constant after each case should be unique.

2. it is not necessary that the constant after case will be in sequance, means 1, 2, 3 is not necess.

3. We can not use real constant after case
means case 4.3 error
case 4 ✓ int
case 'A' ✓ char

4. It is not neccessary to write default: in last we can write where ever we want

5. also write constant after case, we can not use variable.
we can use expration that only includes constants (e.g. 40+3)
be we can't (40+y)
(3>0) → it's result will be

6. we can also use continuty here, means if when you enter 1-10 then case execute for it

case 1 ... 10 ; → elipsis or continuty operator

case 'a' ... 'z' case 'A' ... 'Z'

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:32

Date: / / Page No.:

6. 7 atthease we can not write values by using,
Switch ( ) → we can write here veriable, expration
constant( integer )

```
switch ( expression)
{
   case constant :
        code
   case constant :
        code
   case constant :
        code
}
```

7.

for default :
we simply add

```
default :
code
```

```
default :
printf ("invailid input");
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

```
main()
{
    int x;
    printf ("Enter a number");
    scanf ("%d", &x);
    switch (x)
    {
      case 43 :
        printf ("Sunday");
      case -23 :
        printf ("Monday");
      case 0 :
        printf ("Tuesday");
      case 2 :
        printf ("Wednesday");
    }
    getch();
}
```

if we want to stop our program after any case we can use **break** after the code

output
Wednesday

if x is entered 2 then

if x is intered -23 then Monday Tuesday Wednesday

if x is intered 53 then screen will remain blank and after getting a charactor because of getch() program will end.

For this problem we use **default:**

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

default : it is a keyword.

if the value is not suitable for any case then the screen remains blank but here we want to print something appropriate

जिसका कोई case नहीं होता उसका default; होता है।

- we can't use continue, it is used only in loop.
  we can use break in switch case and also in loop.

elipsis or continuty operator.

```
main ()
{ int x ;
  printf ("Enter a number");
  scanf ("%d", &x);
  switch (x)
  {
   case 1 ... 10 :
   printf ("Sunday");
   case -23 :
   printf ("Monday");
   case 25 :
   printf ("Tuesday");
  default:
   printf (" invailid input");
  }
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

```
# include <stdio.h>
main ()
{
  int x, ~~y=0~~ ;
while(1)
{ printf (" Enter a charactor ");
  fflush (stdin);
  Scanf ("%c", &x);

  switch (x)
  {
    case 'a'...'z' :
      printf ("Small alphabet");
      break;
    case 'A'...'Z' :
      printf (" capital alphabet");
      break;
    default :
      printf (" invailid input");
  } }
  getch();
}
```

- It's a simple program of continuity operator.
- But if we use here while loop then firstly we have to write # include <stdio.h> and in between fflush (stdin);

here while (1)

this means non zero value so the program will never end.

for ending this program can understand in another program

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

Date: / / Page No.:

6.

```
main()
{
    int x ;
    while (1)
    {
        printf ("Enter a number or 0 to end");
        scanf ("%d", &x);
        switch (x)
        {
            case 1 ... 10 :
                printf ("1 to 10");
                break;
            case 11 ... 20 :
                printf ("11 to 20");
                break;
            case 0 :
                printf ("zero");
                exit (0);
        }
    }
    getch ();
}
```

# exit is a predefined function, which directly ends the program

We write exit in two type

1. exit (0); (normally used)
   it means normal termination of program.
2. exit (1);
   it means we have end our program forcefully because of any abnormality.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

Date: / / Page No.:

goto control instruction

सबसे बेकार चीज़ है।
never-ever use it
इससे logics poor होते है
आपके Sylabus में है
इसलिए पढ़ना है।

goto is a keyword

```
function()
{
    code
    goto a1;
    code
    code
a1:
    code
    code
```

goto is for jumping. jumping control from one place to another place and places can be identified ~~by~~ with this name (eg. a1:) this is called lable.

lable → a1 : , we use colon
it have no declaration. we can use various lables in our program
By the use of goto we can move in forward direction as well as in backward direction

Whenevere we write goto we use it with if()

```
function()
{
    code
    if()
    goto a1;
    code
    code
a1:
    code
    code
    if()
    goto a1;
    code
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

Date: / / Page No.

6. 1 Write a program which takes month number as an input and display number of days in that month.

```
main()
{ int mn;
printf ("Enter the month number");
scanf ("%d", &mn);

switch (mn)
{
case 1 :
case 3 :
case 5 :
case 7 :
case 8 :
case 10 :
case 12 :
printf (" 31 days");
break;
case 2 :
printf (" 29/28 days");
break:
case 4:
case 6:
case 9:
case 11:
} printf (" 30 days");
} getch();
```

2. Write unique

Date: / / Page No.:

1. C is a block structured programing Language
2. A C program can have any number of bloks
3. you can have inner blocks too
4. outer blocks are usually functions
5. Function is a block of code, which has some name

```
f1()
{
}
```

here f(1) is a name of function

do not put semicolon here (;)

6. No keyword is a function.
7. A C program can have any number of functions
8. For naming a function we can use the same rules which we apply for naming a variable
9. In a C program, all function must have unique name
10. Even in the smalles C program, there must be one function, and its name must be main()
11. Declaration statements can be written either inside a block or outside a block.
12. Action statement can only written inside the block.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

Date: 9/12/21 Page No.:

Instructions - Action statement

1. Input output Instruction
2. Arithmetic instruction
3. Control instruction

⊙ Output instruction ⇒ Program में लिखी गई वह instruction जिसके exicute होने से keyboard से enter किया गया data, हमारे program के अंदर variable में आता है, वह Output instruction है।

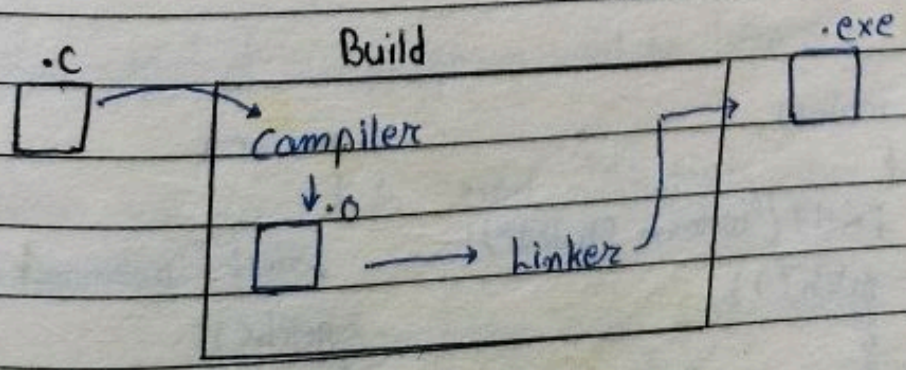
printf() Predefined Function

We use it
1. to print text on the screen
2. to print value of a variable or expression.

Ist program

```
main()
{ printf("WELCOME");
}
```

if we run this program by dable cleaking on .exe file this program ends very quick. we can see this program's output in our code bloks only. for this we will use getch().




Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

Date: / / Page No.:

getch() is a predefined function

its work is to take a charactor from the keyboard.
We use this as a trick - to hold the screen

Now ⇒

```
main()
{
    printf("WELCOME");
    getch();
}
```

11/12/21

Now - we want to print welcome student on our screen.

अगर हम उपर वाले program को modify करके बिना build किए हुए Run करेंगे तो, system हमे पहले से बनी हुई .exe file को Run करके दिखाएगा।

और अगर हम नया function बना रहे है तो इसे Run करने पर यह अपने आप पहले build होगा फिर Run होगा।

```
main()
{
printf("Welcome");
Printf("Students");
getch();
}
```

```
main()
{
printf("Welcome Students");
getch();
}
```

```
main()
{
printf("welcomestudents");
getch();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

Date: / / Page No.:

इन तीनो programs का output क्रमशः ⇒ WelcomeStudents
Welcome students
WelcomeStudents

अब अगर इस program के output को बारीकी से समझना चाहते है तो हम getch() का उपयोग program में कहीं भी कर सकते है।

हमारे program के output मे कोई भी word वहीं से print होता है जहाँ कर्सर होता है।

```
main()
{
    getch();
    printf("Welcome");
    getch();
    printf("students");
    getch();
}
```

if we try to understand the process of programming - firstly there will be a blank screen, here programs first action statment is working, when we type a charector it will end and the second statment will work and it will print welcome, then we type a charector, and students will be print and then we type a charector and program will be end.

And the output of our program will be like this. ⇒ WelcomeStudents

But Now we want to print 'Students' and 'Welcome' in separate lines.

So here we use new line character

back slash n ⇒ \n

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

we use it as printf ("\n") ~~for~~

When we use it \n ~~print~~ do not print either we see its effect - and the cursor will move to the starting of the new line.

```
main()
{ getch();
  printf("Welcome");
  printf("\n");
  printf("Students");
  getch();
}
```

(or)

```
main()
{
printf("welcome\n");
printf("students");
}
```

(or)

```
main()
{
  Printf("Welcome\n Students");
  getch();
}
```

The output of above 3 codes will be same

~~Escape Seq~~

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:22

Date: / / Page No.:

Escape Sequences — Escape sequences are those charectors which starts with \n.

These are spacial charectors

| | |
|---|---|
| \n | New line |
| \t | Tab space |
| \b | Backspace |
| \r | Carriage Return |
| \\ | print \ |
| \" | print " |
| \' | print ' |

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:22

printf() can be used to print value of a variable or expression

```
main()
{
    int a = 5 ;
    printf("a");
    getch();
}
```

here we have write a is "" quats and the output will be → a so this is not ~~wr~~ right

```
main()
{
    int a = 5 ;
    printf(a);
    getch();
}
```

→ This is also wrong.

Now what is ~~wr~~ right

```
main()
{
    int a = 5 ;
    printf("%d", a);
    getch()
}
```

Format Specifier → %d int
%c char
%f float
%H double

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:22

```
main()
{
    int i, j ;

  for ( i=1 ; i<=4 ; i++)
{
   for ( j=1 ; j<=5 , j++ )
   {
      if()
       printf ("*");
      else
         printf ("$");
   }
      printf ("\n");
  }
    getch();
}
```

| i | j | i=2 | i=3 |
|---|---|---|---|
| 1<=4 | 2<=4 | 3<=4 | |
| 1<=5.... | 1<=4.... | 1<=4.... | |
| if (j<=1) | if(1<=2) | if(1<=3) | |

```
*
* *  -  -
* *  *  -
* *  *  *
```

i ii

4<=4
1<=4
if (1<=4)

## Star pattern

j colomn

| | | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| | 1 | * | | | |
| i | 2 | * | * | | |
| rows | 3 | * | * | * | |
| | 4 | * | * | * | * |

| i | j |
|---|---|
| 1 | 1 |
| 2 | 1 2 |
| 3 | 1 2 3 |
| 4 | 1 2 3 4 |

| i | j |
|---|---|
| 1 | j<=1 |
| 2 | j<=2 |
| 3 | j<=3 |
| 4 | j<=4 |

$\boxed{j <= i}$

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:32

Date: / / Page No.:

now we want to print - a = 6

// a = 6

```
main ()
{
    int a = 6 ;
  printf ( "a = %d" , a ) ;
  getch ( ) ;
}
```

now we want to print - // value of a is 6 and b is 7

```
main ( )
{
    int a = 6 , b = 7 ;
  printf ( " value of a is %d and b is %d", a, b) ;
  getch ( ) ;
}
```

now // 6+7 = 13

here a+b is expression

```
main ( )
{
    int a = 6 , b = 7 ;
  printf ( " %d + %d = %d", a, b, a+b ) ;
  getch ();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:22

```
main()
{
    int i, j ;

    for(i=1; i<=4; i++)
{
    for ( j=1 ; j<=4 ; j++)
    {
     if ( j<=i)
       pritf ("*");
     else
          printf (" ");
     }
      printf ("\n");
    }

    getch();
}
```

```
main()
{   int i, j, lines;
    printf ("Enter the number of lines");
    scanf ("%d", &lines);
 for (i=1, i<=lines; i++)
 {
  for (j=1; j<=lines; j++)
  {
   if ( j<=i)
     printf ("*");
   else
        printf (" ");
   }
     printf("\n");
  }  getch();
 }
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:32

Date: 9/12/2021 Page No.:

char m = 'A';

# Character Encoding technique → ASCII

(American standard code for information interchange)

| | | |
|---|---|---|
| | | 0 |
| space → | ' ' | 32 |
| | '0' | 48 |
| | '1' | 49 |
| | '2' | 50 |
| | ⋮ | |
| | '9' | 57 |
| | '@' | 64 |
| | 'A' | 65 |
| | 'B' | 66 |
| | ⋮ | |
| | 'Z' | 90 |
| | | |
| | 'a' | 97 |
| | 'b' | 98 |
| | 'z' | 122 |
| | | |
| | | 255 |

Single ~~qod~~ qoutes is used only on the character contant.
Single qoutes is not used on ASCII code

char m = 'A';
⇒ char m = 65;

int a = 65;
int a = 'A';

# Compiler will consider 'A' as 65
That's why we can rewrite our declaration statement as given above.

Total charecter constants = 256
& ASCII codes are 0 to 255

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

Date: __/__/____ Page No.:

# Star Pattern

```
main()
{ printf("*");
  getch();
}
```

```
main()
{ printf("*****");
  getch();
}
```

```
main()
{ int i=1;
  while (i<=5)
  { printf("*****");
    i++;
  }
  getch();
}
```

```
main()
{ int j;
  for (j=1; j<=5; j++)
  { printf("*");
  }
  getch();
}
```

```
main()
{ int i, j;

  for (i=1; i<=4; i++)
  {
    for (j=1; j<=5; j++)
    { printf("*");
    }
    printf("\n");
  }
  getch();
}
```

```
* * * * *
* * * * *
* * * * *
* * * * *
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:31

Date: / / Page No.

Keywords = Reserved words = 32 in C

| | | | |
|---|---|---|---|
| auto | double | if | sizeof |
| break | enum | long | short |
| case | else | return | typedef |
| char | extern | register | union |
| continue | for | static | unsigned |
| const | float | struct | void |
| do | goto | signed | volatile |
| default | int | switch | while |

Declaration Statement - Data type declaration instruction

instruction = Statement = Command

1 Declaration Statement

Data type ⇒ int
Primitive char
float
double
void

C is a case sensitive language. At the end of each statment we have to write ;

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:20

Date: / / Page No.:

```
main()
{
    int a, b, c;
    printf ("Enter three numbers");
    scanf ("%d %d %d", &a, &b, &c);
if (a>=b && a>=c)
    printf ("%d", a);
    else
    {   if (b>=a && b>=c)
        printf ("%d", b);
        else
        printf ("%d", c);
    }
  getch();
}
```

```
main()
{   int a, b, c;
    printf ("Enter three numbers");
    scanf ("%d %d %d", &a, &b, &c);
  if (a>=b && a>=c)
    printf ("%d", a);
else { if (b>=c)
        printf ("%d", b);
      else
        printf ("%d", c);
    }
  getch();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:28

1 byte = 8 bits

int a, b ; a, b is variable and a, b is int type variable

char m ; m is variable and m is char type

float k ; k is a variable and k is float type variable

double d1 ; d1 is a variable and its double type variable

in int type variable we can store

int type variable ⇒ 4 bytes → integer constant

char type variable ⇒ 1 bytes → Character constant

float type variable ⇒ 4 bytes → Real constant

double type variable ⇒ 8 bytes → Real constant

float is single precision
double is double precision
double is more accurate

float k = 0.7 ;
double d1 = 0.7 ;

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

```
main()
{   int a,b,c;
    printf ("~~%d %d~~ Enter three numbers");
    scanf ("%d %d %d", &a, &b, &c);
    if ( a>=b && a>=c)
      printf ("%d", a);
    else
        { if ( b>c)
            printf ("%d", b);
          else
            printf ("%d", c);
        }
    getch();
}
```

```
main()
{ int a,b,c;
  printf ("Enter three numbers");
  scanf ("%d %d %d", &a, &b, &c);
  if (a>b)
    { if (a>c)
        pritf ("%d", a);
      else
        printf ("%d", c);
    }
    else
    { if (b>c)
        printf ("%d", b);
      else
        printf ("%d", c);
    } getch();
}
```

```
main()
{ int a,b,c;
  printf ("Enter three numbers");
  scanf ("%d %d %d, &a, &b, &c);
  if (a>b)
    a>c ? printf("%d" a): printf("%d", c);
  else
    b>c ? printf("%d", b) : printf("%d", c);
  getch();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:28

Date: __/__/__ Page No.:

```
main()
{
    int a,b,c;
    printf ("Enter three numbers);
    scanf ("%d %d %d", &a, &b, &c);

    a>b ? a>c ? printf("%d",a) : printf("%d",c): b>c ? printf("%d",b): printf(%d,c);
    getch();
}
```

```
main()
{
    int a,b,c;
    printf ("Enter three numbers");
    scanf ("%d %d %d", &a, &b, &c);

printf ("%d", a>b ? a>c ? a:b : b>c ? b:c);
    getch();
}
```

**Preprocessor** :- when we build our program firstly preprocessor program runs and read .c file. in our .c file line that have starting symbol # will be read by preprocessor

**# preprocessor directive**

whatever preprocessor does is not a part of c language. c language starts where compiler strats its work

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:28

```
main ( )
{ int i = 1 ;
  while ( i <= 10 )
  { printf ("MySirG");
  i += 2
  }
```

1 <= 10 , 3 <= 10 , 5 <= 10 , 7 <= 10 , 9 <= 10

5 times this will be true and after it this will be false

```
main ( )
{ int i = 1 ; //initialization
  while ( i <= 10)   // condition
  { printf ("MySirG");
  i += 2;            // flow
  }
}
```

Date: / / Page No.

## break

1. break is a keyword.
2. it can be used only in the body of loop or switch
3. it terminates the execution of loop

```
while ( condition )
{
    code
    code
    if ( condition )
        break ;
    code
}
```

## Continue

1. continue is a keyword.
2. It can only be used in the body of loop
3. it transfers the control to next iteration.

```
while ( condition )
{
    code
    code
    if ( condition )
        continue ;
    code
}
```

# the continue statement skip some code inside the loop and continues with the next iteration.

# It is mainly used for a condition so that we can skip some lines of code for a particular condition

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:29

Entry Control Loop

while

```
while (condition)
{
  code
  code
}
```

Exit Control loop

do while

```
do
{
    code
    code
} while (condition);
```

```
int i = 1;
while (i <= 10)
{
  printf ("%d", i);
  i++;
}
```

```
int i = 1;
do
{
    printf ("%d", i);
    i++;
} while (i <= 10);
```

9 times

| | | |
|---|---|---|
| Condition cheak | 11 | 10 |
| Condition true | 10 | 9 |
| Condition false at | 11 <= 10 | 11 <= 10 |
| output will be | printed 1-10 | printed 1-10 |

The difference between while and do while is ⇒

if the condition is false
- while's code will not run even once.
- do while's code will run once in every situation

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:29

Date: / / Page No.:

# Entry Control Loop

## for

```
for ( ; ; )
{
  code
  code
}
```

```
for ( initialization; condition; flow)
{
  code
  code
}
```

```
int i;
for ( i=1 ; i<=10 ; i++ )
{
  printf ("%d", i);
}
```

```
for (i=1 ; i<=10 ; i++)
{
  printf ("%d", i);
}
```

T F

we can write ⇒

```
int i=1;
for ( ; ; )
{
  if (i==11)
  break;
printf ("%d", i);
i++;
}
```

infinite loop

```
int i=1;
for( ; ; )
{
  printf ("%d", i);
  i++;
}
```

we can write ⇒

```
while ( )
{
  for ( ; ; )
  {
  }
}
```

# you can write any loop inside any loop.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:30

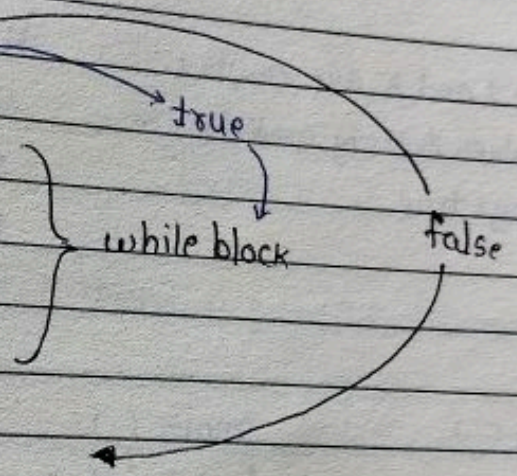
while

Iterative Control instruction | loop

methods of implementation ⇒ 1) while
2) do while
3) for

while is a keyword

```
while (condition)
{
    code
    code
}
```




condition of while is true → Control while block के अंदर आएगा → while block के अंदर का code execute होगा → while की body end होने पर → Control वापस while के पास पहुंचेगा → then again condition check होगी → फिर से true है तो फिर से block के अंदर आएगा → जब condition false हो जाएगी तो loop end हो जाएगा।

iterative means repeat

repeatation हो रहा है इसलिए ही इसे loop कहते हैं।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:30

3

Date: / / Page No.:

1) Read and understand question
2) Test case
3) Dry run

1) Read and understand question.
2) imagine output screen.
3) take a example

# if you are watching a solution of a problem then that problem is worthless for you.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:19

Date: / / Page No.:

If you are not born with a silver spoon then skillset is something that will help you improve your life & becoming independent.

Be very impatient and tought to yourself
Be very patient and kind to others.

Without discipline, you are nothing
Embrace ideas, however always insist on actions and executions.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:19

Date: / / Page No.:

Kernel ⇒ kernel is a program that directly interact with hardware. The kernal is an important part of the operating system.

System software ⇒ It is also a program. System software are device drivers. System software interacts with kernal.

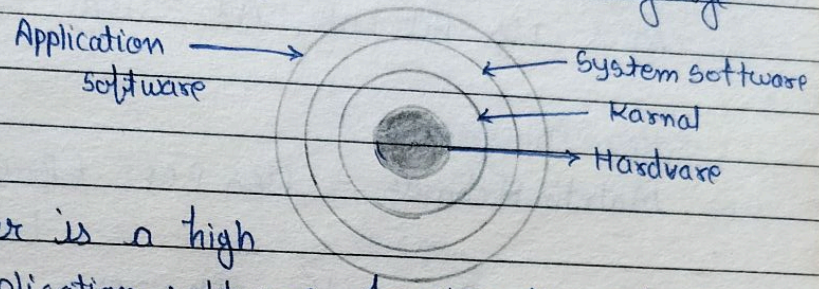
Application software ⇒ Calculator, Microsoft office, point, word, zoom,

Application software's code interact with system software then system software's code/program interact with kernal and then kernal's code/program interact with hardware

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:19

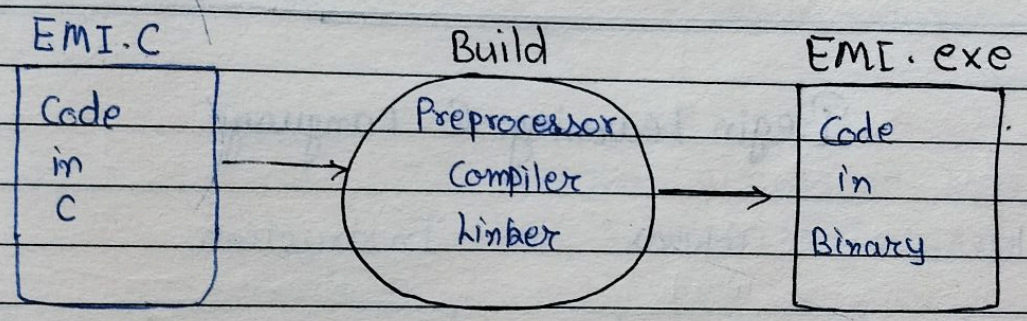
Date: 27/11/21 Page No.:

C is middle level language = High level + Low level language




High level :- Calculater is a high level application software. In this the coder has no mean with the hardware.

Low level :- for karnal and system software's coading low level lenguage is required.




Source File

Executable file (Software)

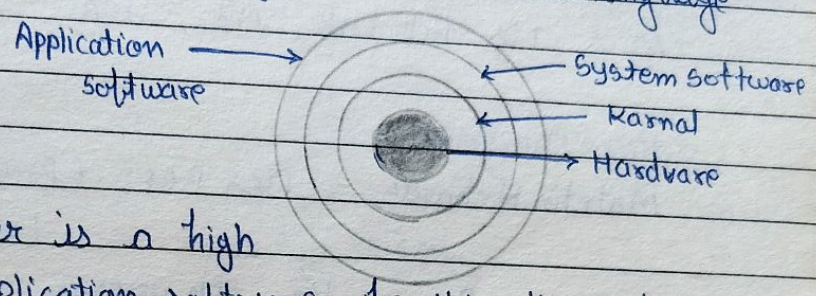
1. Create a source file and write a source code in C lengu
2. Build software

IDE = Integrated Development Enviroment

Code Editor
Build Tools (Preprocessor, compiler, linker, Debugging Tool

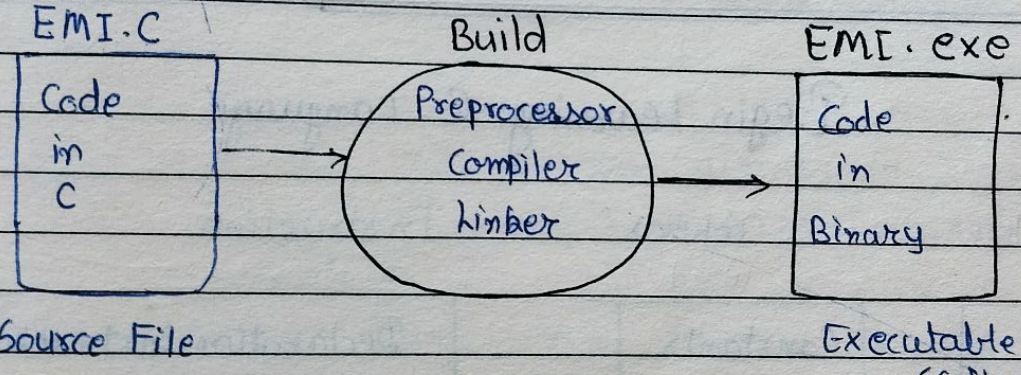
Date: 27/11/21 Page No.:

C is middle level language = High level + Low level language




High level :- Calculater is a high level application software. In this the coder has no mean with the hardware.

Low level :- for karnal and system software's coading low level language is required.




1. Create a source file and write a source code in C lengu
2. Build software

IDE = Integrated Development Enviroment

Code Editor
Build Tools (Preprocessor, compiler, linker, Debugging Tool

Date: 30/11/21 Page No.:

# History of C language

AT & T's Bell Lab, NJ, USA

Ken Thompson
Martin Richards - 1966 BCPL (Basic Combined Programing Language)

1969 - B - Ken Thompson
1972 - C - Dennis Ritchie

Unix opreting system

---

# Begin Learning C Language

| Basic Letter | Tokens (Word) | Instruction (Sentence) |
|---|---|---|
| a to z<br>A to Z<br>0 to 9<br>% & ( [ ] \<br>: " , ? +<br>= , all keys of keybord | Constants<br>Variables<br>Keywords<br>Function<br>Macro<br>Label | - Declaration instruction<br>- Action statement |

syntact - grammar
error - mistake

syntactical error - grammatical mistake

Program

(Set of instructions)

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:20

Date: / / Page No.:

Constants = Data = Constants = Information

| Primary Constants | Secondary Constants |
|---|---|
| - Integer - 23, 95, -234, 0 | Array. |
| - Real - 3.4, -0.06, 4.0 | String. |
| -Character - 'single letter' | Pointer |
| | structure |
| | union |
| | enumerator |

Integer = 5, -30, 10000, 9131044717, 8574104986 12
0, -1, 12, 3, - 9954957863

Real - 3.0, .0, -7.3, 0.23, 666.25

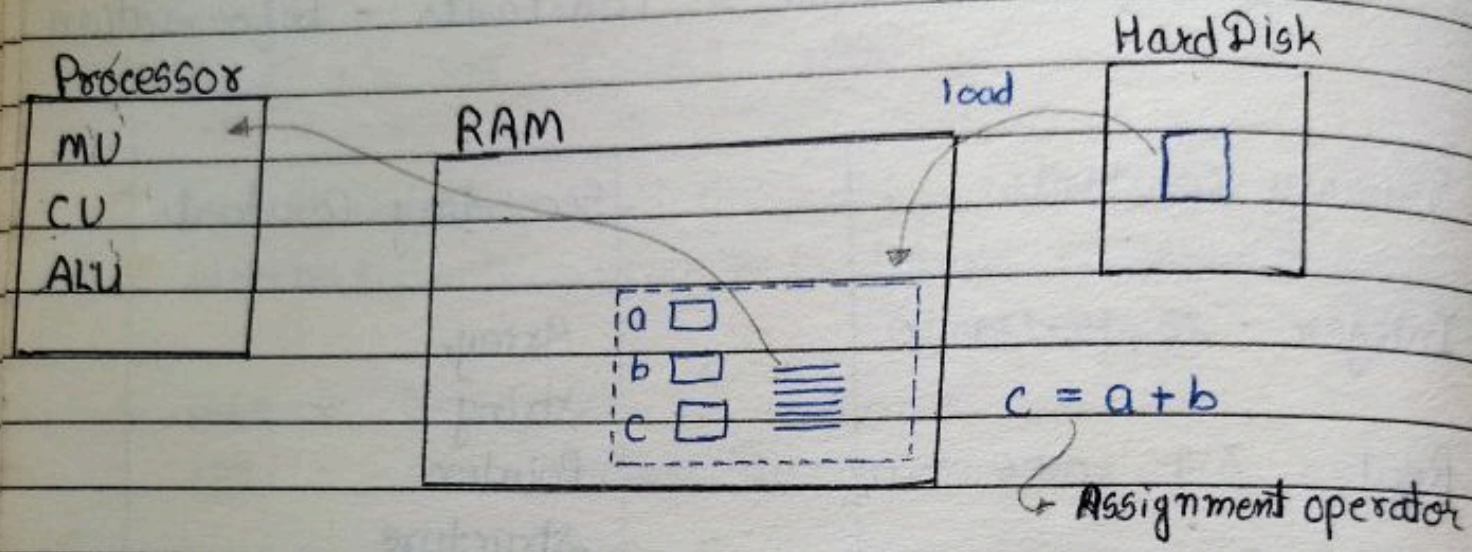
Character - ' ', '_', '?', '5', 'A', 'z', 'a'
'*', '/', '.'

String - "Bhopal" "4.3"

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:20

Date: 2/12/21 Page No.:

# Variables




Variables are memory locations where we store our data.

Variable name is any combination of alphabets, digits and underscore No other symbol is allowed.
Variable name can not start with a digit

A, a3, x-4, da, da_23_ab

x-1 ⇒ it cannot be a variable name because we have used hyfen (-) and in variable name no other symbol is allowe expect underscore.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:20

Date: / / Page No.:

Now we can also write ~~it as~~ ⇒ this code as ⇒

```
main ()
{ int year;                                  // 1997, 2020
  printf ("Enter year");
  scanf ("%d", &year);

  if ( year % 400 == 0 || year % 100 && year % 4 == 0)
     printf ("Leap year");
  else
     printf ("Non leap year");

  getch ();
}
```

Write a program to find greater among three numbers

```
main ()
{   int a, b, c;
    printf ("Enter three numbers");
    scanf ("%d %d %d", &a, &b, &c);
   if (a >= b && a >= c)
   printf ("%d", a);
   else
       { if (b >= a && b >= c)
           printf ("%d", b);
         else
             { if (c >= a && c >= b)
                 printf ("%d", c);
             }
       }
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:28

we can also write it as

```
main()
{ int n ;
  printf ("Enter the number");
  scanf ("%d", &n);
printf ( n > 0 ? "Positive" : "Non positive");
  getch ();
}
```

else if lader

```
if (condition)
    code
else
    {   if (condition)
            code
        else { if (condition)
                code
               else { if (condition)
                       code
                    else
                       code
                    }
             }
    }
```

else if lader

```
if (condition)
  code
else if (condition)
  code
else if (condition)
  code
else if (condition)
  code
else
  code
```

we can write it as (because if else is one action statment)

The style of writting the code in this way is called else if ladder.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:26

year

Sentuary → ( % 400 == 0 ) → Leap year
Sentuary → not divisible → Noneleapyear

Non sentury → ( % 4 == 0 ) → Leap
Non sentury → not divisible → Non leap

**nested if else** ⇒ if के अंदर if else
else के अंदर if else

```
main()
{
   int year;
   printf ("enter year");
   scanf ("%d", &year);
if (year % 100 == 0)
  {
    if ( year % 400 == 0)
     printf ("Leap year");
    else
     printf ("Non leap year");
  }
else
  { if (year % 4 == 0)
     printf ("Leap year");
    else
     print ("Non leap year");
  }
 getch();

}
```

This type of code ↑ is called nested if else.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:27

Date: / / Page No.:

30 sep. 2022

Test for certificate

Objective type test

each quation has 4 options 1 is right option

no nagative marking . in certificate marks will not display

have to clear test ⇒ passing marks 50 %

for downloading certificate ⇒ click on certificates then after clearing the test there will be available a download button

if fail in test → then you must have to reset the test

test reset request ⇒ go on profile ⇒ support ⇒ Raise ticket

in description write → please reset my test
Batch code SB
subject → issue in test
category → other technical issue

Then click on Raise Ticket

it will take 24 hours after it you will get notified

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:18

Date: / / Page No.:

| | 512 | 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|
| 33 → | | | | | 1 | 0 | 0 | 0 | 0 | 1 |
| 83 → | | | | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| | | | | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

int x ;
x = 33 & 83 ;

output = $(1)_{10}$

x = 55|91

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 55 → | | 1 | 1 | 0 | 1 | 1 | 1 |
| 91 → | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

output $(127)_{10}$

x = 62^39

| | | | | | | |
|---|---|---|---|---|---|---|
| 62 → | 1 | 1 | 1 | 1 | 1 | 0 |
| 39 → | 1 | 0 | 1 | 0 | 0 | 1 |
| | 0 | 1 | 0 | 1 | 1 | 1 |

output $(23)_{10}$

x = ~55

| | | | | | | |
|---|---|---|---|---|---|---|
| 55 → | 1 | 1 | 0 | 1 | 1 | 1 |
| | 0 | 0 | 1 | 0 | 0 | 0 |

output $(8)_{10}$

x = 75>>2

75 → 1 0 0 1 0 1 1

output = $(18)_{10}$

>>2 → 0 0 1 0 0 1 0 1 1

x = 12 << 3

1 1 0 0

1 1 0 0 0 0 0

output = $(96)_{10}$

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:25

Obs ⇒ screen recorder

Date: / / Page No.:

Home key - will move your cursor to the beginning line

End key - will move your cursor to the end of line.

Delete key - The letters of the text will be erased from the right side of the cursor.

ctrl + Back space - will erase complete word

windows + V ⇒ all your previous copyed thing will be here.
↳ you have to enable it than it will work

windows + [→] ⇒ for opening two windows side by side.
right arrow key
you can do this thing by your mouse cursur just catch the window and put it on right side it will automatic give you the option

if in your tars bar ⇒ crome, fiEx, VS code - - - -
(in this condition if you want to open crome then press ⊞ + 1
if VS code then press ⊞ + 3

Windos + D ⇒ your all windows will be minimize and you will see your desktop / you can do it through mose right side extreme lower

Windows + tabe ⇒ you can see all your open windows

alt + tabe ⇒ you can open any your previous open window

I am a good boy!

if your cursur is on end point and you press Shift + End then all your text will be selected.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:18

Date: __/__/__ Page No.:

# Relational Operators / Comparision operators

| | | |
|---|---|---|
| < | less than | inequality operators — $I^{st}$ priority |
| > | greater than | |
| <= | less than or equal to | |
| >= | greater than or equal to | |
| == | equal to | equality operators — $II^{nd}$ priority |
| != | not equal to | |

Result of Relational operators → 0 or 1 , 0 if false relation
1 if true relation

| | output |
|---|---|
| x = 5 < 4 ; | 0 |
| x = 3 != 4 ; | 1 |
| x = 4 < 4 ; | 0 |
| x = 5 > 4 > 3 ; | 0 → x = 1 > 3 ; fals → 0 |
| x = 0 != 0 < 1 ; | 1 |

# Logical Operators

| | |
|---|---|
| ! | Not operator ( Unary ) Top prioriit > Rela., Bitwi. Arith etc. |
| && | And |
| \|\| | OR |

! Not operator ⇒ every non zero value is true

!0 → 0 → false
!1 → 1 → true
!-1 → 1 → true
!5 → 1 → true

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:25

Date: / / Page No.:

I am a good / boy.

if your cursure ↗ is here and you press Shift + Home i am a good will be selected.

if yo press Shift + End boy. will be selected.

Windows + P ⇒ in case (😐) you have two connected monitor you can control it.

windows + L ⇒ For locking your pc

mouse whele. you can press it and open any search result in which tabe you wcant.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:19

Date: / / Page No.:

dry run

$x = !5 > -2$ ; $x = 1 > -2$ ; true output 1

output = 1

$x = !(5 > -2)$ ;

And && $x > 0$ && $y > 0$

exp1 && exp2

T && T = T

T && F = F

F && X = F

OR || exp1 || exp2

F || F = F

F || T = T

T || X = T

Assignment operators - = it prints the right side value in left side

a = 4

assignment operator में left side में हमेशा variable होता है।

memory location / Container

int x = 5;

x = x+3;

content

4 = a ; error

3 = 4 ; error

variable =

memory location

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:25

Date: / / Page No.:

# Compound Assignment Operation

+= -= *= /= %= >>= &=

x += 3 ; x = x+3

x* = 5 ; x = x*5

x& = 8 ; x = x&8

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:25

Date: / / Page No.:

## Control instruction ⇒

We have written a program of 7 lines and we want that after our 3rd line, 7th line of our code will run. means we do not want to run 4th, 5th and 6th line of our code.

Why - e.g. ATM machine. The code have written once for this software.

In this software there is a instruction written for withdraw money. But if you do not want to withdraw money and you just want to cheak your balance.

may be in futur you want to withdraw money but currently you just want to cheak balance.

That is why. control instructions are important

Control instruction -

1. Decision Control
2. Iterative Control
3. Switch case control
4. goto control

Decision Control instruction or Selection Control instruction

if

if else

Conditional operator

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:26

Keyword and keyword is never called function

if ( ) Expration — condition (T or F)

```
{
    code
    code
}
```

if block

if true → if block will run

if false → if block will not run ✗

```
main()
{
  line 1 ;
  line 2 ;
  line 3 ;
  line 4 ;
  line 5 ;
  line 6 ;
  line 7 ;
  line 8 ;
  line 9 ;
  line 10 ;
}
```

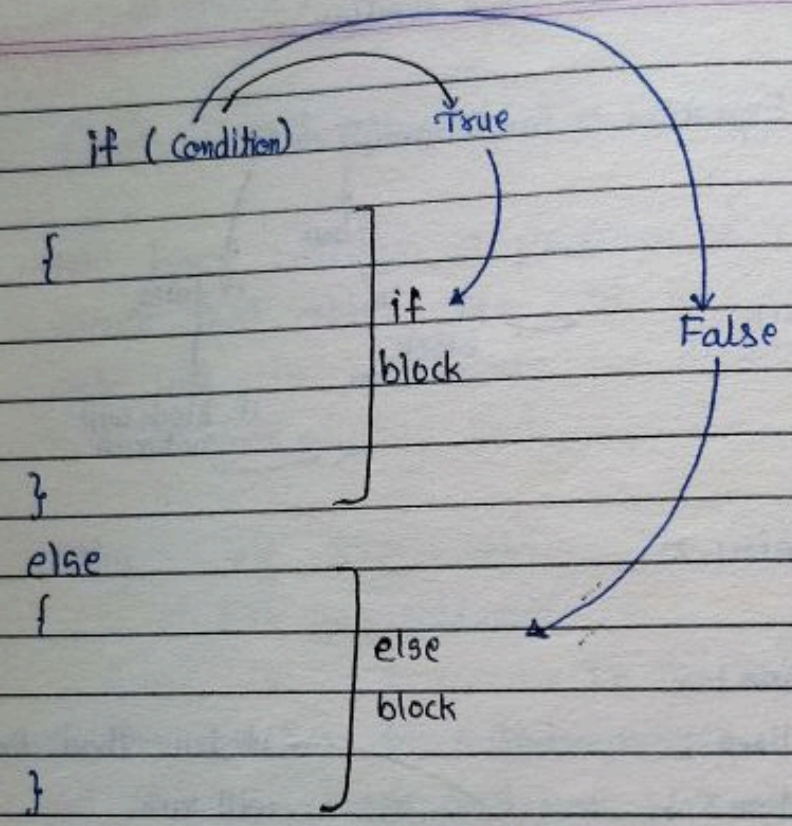
```
main()
{
  line 1 ;
  line 2 ;
  line 3 ;
  if (expration) condition)
  { line 4 ;
    line 5 ;
    line 6 ;
  }
  line 7 ;
  line 8 ;
  line 9 ;
  line 10 ;
}
```

if block

→ if true than line 4, 5, 6 will run

→ false (0) means line 4, 5, 6 will not run

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:26




else have no ( ), meanse it have no condition, it depends on if's (expration/condition).

if का block खत्म होते ही else लिखना है। हम बीच मे कोई और लाइन नही लिख सकते, it will show error.

else कभी अकेला use नही हो सकता क्योंकि उसका meaning ही नही है।

else, if दोनो ही keyword है।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:26

: colon

Conditional operator ⇒ (? :) ⇒ (Ternary operator)

exp1 ? exp2 : exp3;

True

False

```
main()
{   int n;
    printf("Enter the number");
    scanf("%d", &n);
  if (n>0)
    printf("Positive");
  else
     printf("Non positive");
    getch();
  }
```

```
main()
{   int n;
    printf("Enter the number");
    scanf("%d", &n);
    n>0 ? printf("Positive") : printf("Nonpositive");
    getch();
  }
```

exp1 ? exp2 : exp3; it is one expression

if exp1 is true then exp2 will work

if exp1 is false then exp3 will work

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:26

23 Nov - 30 Jan - 30 Sep.

C language / Batch Code : SB  Date: 23/11/21 Page No.:

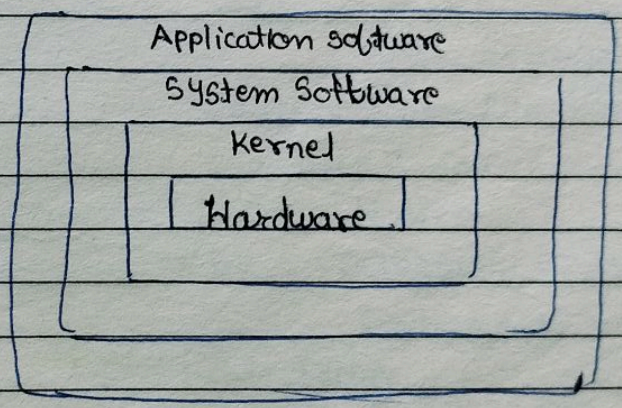
https://premium.mysirg.com ((create account here))

here - recording of Live classes
(shub01690@gmail.com) Assignments  nidhu@01690
- Certificate
- Discussion Forum
- support
- Account

https://coder.mysirg.com | user Name NIDHIYADAV
Pass. → 919799

---

Computer




Hardware ⇒ (tangible) RAM, hard disk, cache memory
Processor, Touchpad, USB, mouse etc

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:18

Date: / / Page No.:

| | | | |
|---|---|---|---|
| ✓ | | 1 | Orientation lecture |
| ✓ | | 2 | Fundamentals for C language |
| ✓ | | 3 | What is programing, Role of C, IDE |
| ✓ | | 4 | History, Begin learning C, Constants |
| ✓ | | 5 | Variables and keywords |
| ✓ | | 6 | Data type declaration instruction |
| ✓ | | 7 | float vs double, ASCII, block, Rules |
| ✓ | | 8 | Output instruction, printf, getch, build and run |
| ✓ | | 9 | Printf, scanf, escape sequence, formate specifier |
| ✓ | | 10 | Unary operators |
| ✓ | | 11 | Arithmetic Operators |
| ✓ | | 12 | Bitwise, relational, logical, assignment operators, if...else |
| ✓ | • | 13 | Conditional Operator |
| ✓ | | 14 | nested if else, else if ladder |
| ✓ | • | 15 | while |
| ✓ | • | 16 | break and continue |
| ✓ | • | 17 | Loop |
| ✓ | • | 18 | do while, for loop |
| ✓ | • | 19 | Doubt handling |
| ✓ | • | 20 | more on loops |
| ✓ | | 21 | star pattern logic |
| ✓ | | 22 | Switch case, goto control |
| ✓ | • | 23 | Functions introduction |
| ✓ | • | 24 | Ways to define a function |
| ✓ | | 25 | Recursion |
| ✓ | | 26 | Array introduction |
| ✓ | | 27 | arrays and functions |
| ✓ | | 28 | Arrays and functions |
| ✓ | | 29 | 2d Arrays |

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:52

Date: / / Page No.:

30 Strings
31 Strings and function
32 multiply strings, pointer, pointer's arithmetic
33 Applications of pointers, call by reference
34 structure
35 Vidio lesson 50
36 Union, enum, typedef, storage classes
37 Preprocessor, Last class

Assignment - 1 | Printf, scanf
Assignment - 2 | Operators
Assignment - 3 | Decision control
Assignment - 4 | while loop
Assignment - 5 | loop
Assignment - 6 | More on Loops
Assignment - 7 | Star pattern problems
Assignment - 8 | Switch
Assignment - 9 | functions
Assignment - 10 | Recursion
Assignment - 11 | Arrays
Assignment - 12 | Arrays and functions
Assignment - 13 | 2d arrays
Assignment - 14 | Strings
Assignment - 15 | strings and functions
Assignment - 16 | multiple strings
Assignment - 17 | Pointer, structure, DMA

Test for Certificate

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:52

Date: / / Page No.:

# Recursion - Function calling itself is called recursion

```
main()
{
  int k;
  k = f1(3);
  printf("%d", k);
}
int f1(int n)
{
  int s;
  if (n==1)
  return 1;
  s = n + f1(n-1);
  return s;
}
```

- Every recursive solution can be replaced by itarative solution.
- recursion का code छोटा होता है, इसका logic सोचना बहुत आसान है, इसलिए itaration से ज्यादा सही है इसे use करना
- सारी problems, recursion से नहीं बनती
- Recursive case (RC)
- Base case (BC)

here main() is not a recursive function

f1 is a recursive function

Ass. Tips →
1. Function name
2. Recursive case
3. Base case

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:39

# Program's memory

Date: / / Page No.:

main()

| K | |
|---|---|
| 6 | int K; |
| | K = f1(3); |
| | Printf ("%d", K); |

3

int f1(int n)

6

| n | |
|---|---|
| 3 | int S; |
| | if (n==1) |
| | return 1; |
| S | |
| 6 | S = n+f1(n-1); |
| | return S; |

2

int f1(int n)

3

| n | |
|---|---|
| 2 | int S; |
| | if (n==1) |
| | return 1; |
| S | |
| 3 | S = n1 + f1(n-1); |
| | return S; |

1

1

int f1 (int n)

| n | |
|---|---|
| 1 | int S; |
| | if (n==1) |
| | return1; |
| S | |
| | S = n1 + f1(n-1); |
| | return S; |

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:40

```
main()
{ int k;
  k = f1(3);
  printf("%d", k);
}

int f1(int n)
{ int s;

  if (n==1)
  return 1;        } BC
  s = n + f1(n-1);
  return s;        } RC
}
```

f1(3)
3 + f1(2)
2 + f1(1)
1

f1(10)
10 + f1(9)
9 + f1(8)

Sum(n)
n + sum(n-1)

Base case - वो situation जब function खुद को call करना बंद कर देगा

in a recursive function, make sure की वो कहीं पर खत्म हो रहा है, और जहाँ खत्म हो रहा है वही Base case है, you have to always think about this base case

Recursive case - this is our main formula to solve the problem.

recursion से किसी problem को solve करना है तो उसमे एक ऐसी लाइन लिखनी पड़ेगी जिसमे तुम उसी function को दोबारा call कर रहे होगे, लेकिन अब जब call कर रहे हो तो original problem से थोड़ा simpler version के लिए call करो। जैसे 10 number का निकालने से तो simple है 9 number का sum निकालना, [illegible] का कैसे निकालना है वो recursion पर छोड़ दो

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:40

Date: / / Page No.:

- अब add मे हम जो भी values लिख देंगे, वो चली जाएगी a और b मे e.g. add (10, 20); लिखने पर पहली value a मे जाएगी दुसरी value b मे जाएगी और उसका ~~some~~ sum हो जाएगा, last मे output आएगा sum is 30.

- या हो सकता है main function के अंदर दो variables हो उसके अंदर कोई value रखी हो और उन variables को हम add के parenthesis मे लिखे।

```
int x = 10 , y = 20;
add (x, y);
```

- या ये भी हो सकता है कि main function के अंदर हम user से values enter कराना चाहे तो वो भी कर सकते है

```
printf ("Enter two numbers");
scanf ("%d%d", &x, &y);
add (x, y);
```

- ये भी हो सकता है कि user से ना लेकर, कोई calculation कर रहे है उसका result आया है वो x मे रखे या y मे रखे, फिर उनको add करना है

main function मे तुम x और y मे data कैसे ला रहे हो इससे add function को रत्ती भर भी फर्क नही पड़ता

# add function को जो कोई call करेगा ना, वो जिन data को add कराना चाहता है वो data कहा से आएगा वो वही जाने means यह पर main function को ही लेना देना है इस बातसे।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:38

add function, takes something natur का हम बना रहे है, तो हमे दो value मिलेगी, और उन्हे वर्क करना हमारा काम है।

```
# include <conio.h>
# include <stdio.h>
void add (int, int);

main()
{
 ~~int~~ add (10,20);
      getch();
}
void add (int a, int b)
{
 int c;
 c=a+b;
 printf ("Sum is %d",c);
}
```

```
# include <conio.h>
# include <stdio.h>
void add (int, int);
main()
{ printf ("Enter two numbers");
 scanf ("%d%d", &x, &y);
 add (x, y); // Actual Arguments
 getch();
}
void add (int a, int b) // Formal Argument
{ int c;
 c = a+b;
 printf ("sum is %d", c);
}
```

Argument / Parameters

Actual arguments

Formal Arguments

```
# include <conio.h>
# include <stdio.h>
void add (int, int);
main()
{ int x = 10, y=20
 add (x, y);
 getch();
}
void add (int a, int b)
{ int c;
  c = a+b;
 printf ("sum is %d", c);
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:39

Date: / / Page No.:

Actual arguments ⇒ function को call करते समय जो paranthesis मे लिखते है उसे कहते है actual arguments

e.g. add (x, y);

Formal arguments ⇒ जब function define करते है और paranthesis मे जो variable बनाते है उसे कहते है Formal arguments.

e.g. void add (int a, int b)

- main function मे variables के नाम x, y है और add function मे a और b रखा है, अगर हम चाहे तो दोनो जगह same नाम के variable रख सकते है, इससे कोई फ़र्क नही पड़ता, लेकिन ये दोनो variables physically अलग - अलग होंगे -

main function के variable अलग है और add के अलग है

main के पास दो variables है, add के पास दो variable है, main के variable का data, add के variable मे जाएगा तो ये बिलकुल भी जरूरी नही है कि इन दोनो functions के variable का नाम same रखा जाए।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:39

Date: / / Page No.

3. Takes Nothing, Return something

e.g. int add ( )

takes nothing natur का है तो a और b की values user से लेंगे
takes something में user से data नहीं लेना होता है।

यहाँ add की body मे c की value print नही करवाना है क्योंकि हम उसे return करेंगे (return somthing natur)

अभी तक print इसलिए करा रहे थे क्योंकि returns nothing natur का था और function के खत्म हो जाने पर c की value destroy हो जाती जब return something natur का function है, तो इसका मतलब ये है कि जिसने function को call किया था, उसके पास value return होकर पहुंच जाएगी

value return होने के बाद वो function (e.g. main()) इसका जैसे चाहे use कर सकता है।

जो add से value return होकर आ रही है उसे हम s मे assign करवा लेंगे अब इसे हम चाहे तो print भी करवा सकते हैं।

```
# include <conio.h>
# include <stdio.h>
  int add();
  main()
{
  int s;
  s = add();
  printf("Sum is %d", s);
  getch();
}
  int add()
{
  int a, b, c;
printf("Enter two numbers");
scanf("%d%d", &a, &b);
  c = a+b;
  return c;
}
```

# return keyword की मदद से केवल एक ही value return करवाया जा सकता है और return आगर हो उस function के अंदर तो वो function उसी वक्त end हो जाता है

Date: / / Page No.:

Takes something, returns something

```
# include <conio.h>
# include <stdio.h>
int add (int, int);
main ()
{
  int x, y, s;
  printf ("Enter two numbers");
  scanf ("%d %d", &x, &y);
  s = add (x, y);
  printf ("sum is %d", s);
  getch ();
}
int add (int a, int b)
{
  int c;
  c = a+b;
  return c;
}
```

Assignment quations - Assignment 9

in assignment copy

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:39

Date: / / Page No.:

जितने भी predefine function है उनका declaration पहले से किसी file मे लिखा जा चुका है और इस file को बोलते है header file.

getch() का declaration ⇒ conio.h file मे है

# include < conio.h >

# → Preprocessor director
include → command

जैसे ही Program build होता है सबसे पहले preprocessor नाम का प्रोग्राम चलता है और ये आपके Program मे सिर्फ उसी line को देखता है जिसमे # लगा होता है शुरुवात में।

include is a command for preprocessor program

# include <conio.h> लिखने से preprocessor program समझ जाएगा की conio.h ( full form- consol input output) के अंदर जो लाइनस लिखि है उन सबको इस प्रोग्राम के अंदर include करवाना है। e.g. Preprocessor इस एक statment को हटाकर conio.h की सारी-सारी lines लाकर रख देगा

जब compiler इस प्रोग्राम को read करेगा तो उसे #include<conio.h> नही दिख रहा क्योंकि उसकी जगह पर तो बहुत सारी lines आ चुकी है ये lines declaration है कुछ predefine functions की जिनमे से getch भी एक है, तो यहाँ से getch() का declaration compiler को मिल जाता ~~getch का~~ declaration हमेशा केवल compiler के लिए होता है, ये execute नही होता है

Date: / / Page No.:

जितने भी predefine function है उनका declaration पहले से किसी file मे लिखा जा चुका है और इस file को बोलते है header file.

getch() का declaration ⇒ conio.h file मे है

# include < conio.h >

# → Preprocessor director
include → command

जैसे ही program build होता है सबसे पहले preprocessor नाम का प्रोग्राम चलता है और ये आपके program मे सिर्फ उसी line को देखता है जिसमे # लगा होता है शुरुवात में।

include is a command for preprocessor program

# include < conio.h > लिखने से preprocessor program समझ जाएगा की conio.h ( full form- consol input output) के अंदर जो लाइनस लिखि है उन सबको इस प्रोग्राम के अंदर include करवाना है। e.g. preprocessor इस एक statment को हटाकर conio.h की सारी-सारी lines लाकर रख देगा

जब compiler इस प्रोग्राम को read करेगा तो उसे #include<conio.h> नही दिख रहा क्योंकि उसकी जगह पर तो बहुत सारी lines आ चुकी है ये lines declaration है कुछ predefine functions की जिनमे से getch भी एक है, तो यहाँ से getch() का declaration compiler को मिल जाता है declaration हमेशा केवल compiler के लिए होता है, ये execute नही होता है

Date: / / Page No.

printf(), scanf() का declaration → stdio.h file में है

तो हम इनके declaration के लिए # include <stdio.h> लिखते हैं।

- C language के standerdisation के time इसके documents में लिखा हुआ है कि function declaration is recommended but not mandatory.

- Header file के अंदर function की definition नहीं होती सिर्फ function का declaration होता हैं।
  definition, library file में होती हैं
- main() को call ही नहीं कर रहे तो declaration की जरूरत भी नहीं है

~~Takes - Some~~

```
# include<conio.h>
# include <stdio.h>
void add();
main()
{ add();
  getch();
}
void add()
{ int a,b,c
  printf ("Enter two numbers");
  c=a+b;
  printf ("Sum is %d", c);
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:38

Date: / / Page No.:

2. Takes something returns nothing

अब हम add function बनाएंगे जिसका natur है takes something returns nothing.

अब returns nothing है तो void add ....

और takes something है तो void add(.....)

जब भी takes something natur का function बनाते है तो paranthesis मे हमें variables का declaration करना होता है।

कितने variables का ?

- इस फंक्शन (add) को अपना काम करने के लिए कितनी values की जरूरत हैं, जितनी values की जरूरत है उतने variables का declaration करेंगे।

void add ( int a , int b )

variables हमें यहाँ तीन लग रहे है पर values तो दो ही चाहिए।

इसके अंदर हर variable के लिए data type अलग लिखना होता है

अब a, b हमने paranthesis मे declair कर दिया है तो इसे definition के अंदर लिखने की जरूरत नही है और इस condition मे भी a, b, c local variables ही है

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:38

अब अगर a और b मे हमे तो data मिल चुका है, जिसे हमे add करना है, तो फिर हमे printf, scanf लिखने कर user से दुबारा a और b की value मांगने की जरूरत ही नहीं है।

Takes something natur का program है मतलब a और b में data आ चुका है, कहां से आया - जब call हुआ तब आया

और we know की function definition की पहली line ही function का declaration होती है with ;, declaration करते समय variable का नाम लिखना जरूरी नही है लिख देंगे तो भी कोई Problem नही है void add (int, int); Or void add(int a, int b)

अब जब function call करते है तभी function की defination चलती है तो function call करेंगे तो paranthesis खाली नहीं रख सकते

printf को call करते तब paranthesis भरे रहते है, क्योंकि printf function take something natur का define है।

getch — " — खाली रहते — " —
getch takes nothing natur का है।

यहाँ add, takes something natur का है।

```
# include <conio.h>
# include <stdio.h>
void add (int, int);
main ()
{ add (—, —);
  getch();
}
void add (int a, int b)
{ int c;
  c = a + b;
printf ("Sum is %d", c);
}
```

```
main()
{
  add();
  getch();
}

add()
{
  int a, b, c;
  printf ("Enter two numbers");
  scanf ("%d %d", &a, &b);
  c = a+b;
  printf ("Sum is %d", c);
}
```

here

add(); # function call

# function definition

# Function declaration

variable declaration ⇒ a, b, c are new words for compilier until we write declaration statment. declaration statment is always for compiler, it never execute. ( int a, b, c; )

इसे लिखने के दो तरीके हो सकते हैं - या तो add function के अंदर लिखो या तो add function के बाहर लिख दो।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

```
main()
{
  add();
  getch();
}
int a,b,c;  // global variables

add()
{
  printf("Enter two numbers");
  scanf("%d%d", &a, &b);
  c = a+b;
  printf("sum is %d", c);
}
```

```
main()
{
  add();
  getch();
}

add()
{
  int a,b,c  // local variable
  printf("Enter two numbers");
  scanf("%d%d", &a, &b);
  c = a+b;
  printf("Sum is %d", c);
}
```

**global variable** ⇒ ये variables function के end होने पर destroy नहीं होंगे / relese नहीं होंगे बल्की जब program end होगा तब relies होंगे।

the life time of globle variable - till the end of the program they will remain in the memory till the end of the program

**local variable** ⇒ ये variable का life time और scop केवल उस function की body ही रहती है जिस function के अंदर variable declair किया है। इस function के खत्म होते ही a, b, c की memory release हो जाएगी।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

We should prefer local variable because globle variable मेमोरी occupie करके रखते है through out the program. program को run होने के लिए memory limited मिलती है।

- We can assume global variable as a shared container. all the functions are performing different-different operations but they can use that shared variable any time.

global variable बनते है, पर हम इसे last preference देंगे क्योंकि वो मेमोरी ज्यादा occupie कर रहें हैं।

It is not a good idea to make a variable global unnecessa

---

अब जैसे variables का declaration करना होता है वैसे ही functions का भी declaration करना होता है

functions भी नए शब्द है जिनका अर्थ compiler को पहले से नहीं पता है। compiler have no idea about the word add. तो हम dectair करेंगे।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

अब से declare करना भी दो तरीके से हो सकता है
local declaration, global declaration

* Functions को memory तभी मिलती है जब वो call होते है। globally declair करने से memory पर कोई फर्क नहीं पड़ता।

declare कैसे करते है ?

declaration से पहले -
अगर हमारा function, returns nothing है, तो हमें इसके नाम से पहले इसका ~~writ~~ return type लिखना होता है।

return type - जो value return हो रही है उसका type (data type)
अगर int type की value return हो रही है तो int लिखो
अगर कुछ नहीं लिखा हुआ है तो भी int लिखा हुआ माना जाता है
अगर float typ की value return —"— float लिखो
अगर char —"— char लिखो
अगर कुछ भी value return नहीं हो रही है तो void लिखो

void नहीं लिखा होने पर भी function तो run कर रहा था पर लिखना ज्यादा accurate है, इसका मतलब है (void लिखने का) की function कुछ भी नहीं return कर रहा है।

और कुछ भी नहीं लिखा है इसका मतलब है int लिखा है। और warning मिलेगी की by default इसका return type int माना जा रहा है।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

As per the stander - we have to write void

```
main ()
{
    add();
    getch();
}
void add()
{
int a, b, c
printf ("Enter two numbers");
scanf ("%d %d", &a, &b);
c = a + b;
printf ("sum is %d", c);
}
```

function definition की पहली line को ; लगाकर लिखना ही function declaration है।

e.g void add ();

अब ये भी दो तरीके से किया जा सकता है

1) Local declaration
2) global declaration

```
main()
{ void add (); // local declaration
    add ();
    getch();
}
void add()
{
int a, b, c
printf ("Enter two numbers");
scanf ("%d %d", &a, &b);
c = a+b;
printf ("sum is %d", c);
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:38

Function declaration globally करना ज्यादा better है क्योंकि इससे पुरे प्रोग्राम के लिए ये declare हो जाएगा, ना केवल एक function के लिए। और memory का तो कोई issue है ही नही क्योंकि function को मेमोरी तभी मिलती है जब function call होता है।

```
void add ();  // global declaration
main ()
{
   add ();
   getch();
}
   void add ()
{
  int a, b, c
  printf ("Enter two numbers");
  scanf ("%d %d", &a, &b);
  c = a+b;
  printf ("sum is %d", c);
}
```

add () user defined function था जिसका हमने declaration किया।

अब predefine functions जैसे getch(), printf(), scanf() का भी तो declaration होना चाहिए, ये pridefine है इसका मतलब ये नही है कि compiler को पहले से मालुम है कि ये क्या है compiler को keywords पहले से पता होते है पर functions नहीं और ये दोनो अलग चीज़ हैं।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:38

Date: / / Page No.

# Ways to define a function

1. Takes Nothing, Return nothing
2. Takes something, Return nothing
3. Takes Nothing, Return something
4. Take something, Return something

1.

Takes nothing, Return nothing.

- कोई भी function किसी दुसरे function के variable को ~~exis~~ acces नहीं कर सकता। हर function सिर्फ अपनी body मे बने variable को acces कर सकता है।

एक function ने दुसरे function को call किया है तो दुसरा function पहले वाले functions के variables को acces नहीं कर सकता।

अगर कोई function दुसरे function को call करता है और हम चाहते है कि वो पहले वाले function के variables को acces कर लें।

जब ऐसी requirment होती है तब इस situation में takes वाला concept आता है

function को define करने के चार तरीके है

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.

## Ways to define a function

1. Takes Nothing, Return nothing
2. Takes something, Return nothing
3. Takes Nothing, Return something
4. Take something, Return something

1.

Takes nothing, Return nothing.

- कोई भी function किसी दुसरे function के variable को ~~exis~~ acces नहीं कर सकता। हर function सिर्फ अपनी body मे बने variable को acces कर सकता है।

एक function ने दुसरे function को call किया है तो दुसरा function पहले वाले functions के variables को acces नही कर सकता।

अगर कोई function दुसरे function को call करता है और हम चाहते है कि वो पहले वाले function के variables को acces कर लें। जब ऐसी requirment होती है तब इस situation में takes वाला concept आता है

function को define करने के चार तरीके है

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.

for adding two numbers.

instead of program make a function (named as add) and this function's work will be to add two numbers

```
add ()
{
    int a, b, c;
    printf ("Enter two numbers");
    scanf ("%d %d", &a, &b);
    c = a+b;
    printf ("Sum is %d", c);
}
```

लेकिन अगर हमे इसे run करना है, तो ये run नहीं होगा

यहाँ error show होगी Undefined reference to 'WinMain'

मतलब इसमे main function नहीं है, main function के बिना ये चल नहीं सकता।

तो सिर्फ इसलिए कि हम इस function को चलाकर देखना चाहते है हम main function बनाएंगे। otherwise हमारा function बन चुका है

ये जो function बना है ये takes nothing, returns nothing variety का है।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

अब पता कैसे चलता है कि takes nothing, returns nothing है ?

* function की body के अंदर return नाम का keyword नहीं लिखा है, तो it means, ये returns nothing natur का function है।

* function defination मे curly bracket के ऊपर जहाँ function का नाम लिखा हैं ( add() ) उसके parenthisis अगर खाली हैं, तो it means, ये Takes nothing natur का function हैं

# किसी भी function की defination को देखकर ये बताया जा सकता है कि takes nothing | takes something | returns nothing | returns something हैं।

अब जब हम main function बनाते हैं।

तब ये sum होगा।

हम main को पहले | बाद मे अपनी choice के according लिख सकते हैं।

```
add()
{ int a, b, c ;
  printf ("Enter two numbers");
  scanf ("%d%d", &a, &b);
  c = a+b ;
  printf ("sum is %d", c);
}
main()
{ add();
  getch();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

7. Function is way to achieve modularization.

modularization is splitting up a bigger task into several smaller subtasks to reduce the complexity of the bigger task.

8. Functions are of two types

- Predefined function → printf, scanf, getch, exit
- User defined function → main()

# Koi bhi function tab execute hota hai jab hum use call karte hain

```
main ()
{
    a();
    b();
    a();
}
void a ()
{
    printf ("Hello");
}
void b ()
{ printf ("Bye");
  a();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:35

Date: / / Page No.

# here we have define three functions

एक program के अंदर कितने भी functions define करो, इस बात से कोई फ़र्क नही पड़ता की आप उन्हे किस sequance मे define कर रहे है।

in e.g. main(), a(), b()
but we can a(), main(), b() // b(), a(), main()
The execution of our program will remain same.

program के अंदर main नाम का function होना जरूरी है, तभी प्रोग्राम run होगा लेकिन main सबसे पहले बना हो ये जरूरी नहीं हैं।

# here we have call → a(); , b(); , printf();
कोई भी function तब execute होता है जब हम उसे call करते है।

जब हम a को call करेंगे ( a(); ) तब a() execute होगा
जब हम b को call करेंगे ( b(); ) तब b() execute होगा
जब हम printf को call करेंगे (printf(); ) तब printf चलेगा

# But हम main को call नही कर रहे है, पर फिर भी तो चलता है। main function को operating system call करता है

when we execute the program, operating system calls main function firstly, then our program start, that's why we have to define main().

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:35

Date: __/__/__ Page No.:

जब भी program run होता है run होने के लिए उसे ram में कुछ memory मिलती है (suppose - Programs memory)

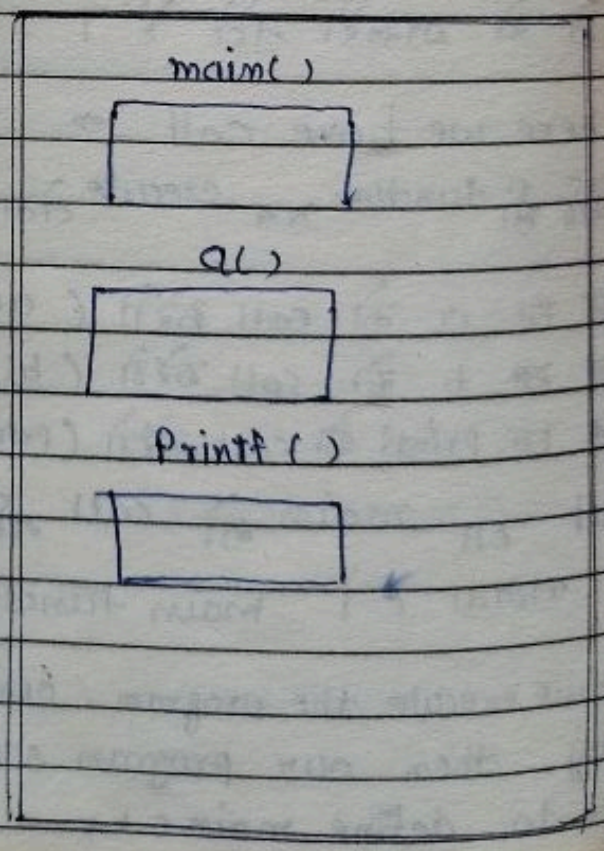
```
main()
{
  a();
  b();
  a();
}
void a()
{
  printf("Hello");
}
void b()
{
  printf("Bye");
  a();
}
```

- जब भी कोई function call होता है, चाहे हमने call किया हो या operating system ने। सबसे पहले वो RAM मे load होता है।

आपके program को run होने के लिए जो memory मिलि है उसी के अंदर कुछ हिस्सा (जगह), ये function occupie कर लेगा।

fog e.g. main function will ocupie some space

Program's Memory




- किसी भी function को मेमोरी दो कारणो के लिए मिलती है 1st) for storing the instruction 2nd) for variables

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:35

← control

Date: / / Page No.:

Now we have call the function main. So it's execution will start.

line 1 a(); ←

यहां a function call हुआ है, function जैसे ही call होता है, तो RAM मे load होता है।

now the control → a() ←

अभी main function की पहली line चल रही है और इसका मतलब है a function की पुरी defination का चलना।

Now in function a(), we have call function printf();, function जैसे ही call होता है, तो RAM मे load होता है,

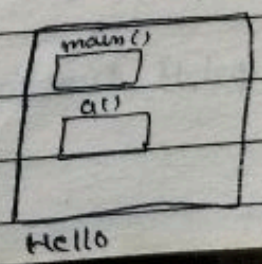
अब printf की defination जहां कही भी होगी तो execute होगी और इस execution से Hello print हो जाएगा।

printf function की coding finish हो गयी, code चल गया Hello print हो गया।

अब control exactly वही आएगा, जहां से function call हुआ था ( printf("Hello"); ← )

और printf function की memory relese हो गयी

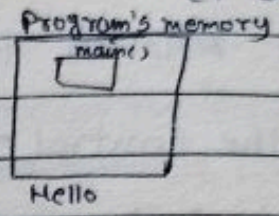
Program को जो memory मिली है उसमे से कुछ area consume हुआ है और बाकी free है। consumed and free का ratio change होता रहता है during execution of the program.




Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:36

Date: / / Page No.:

अब a function की भी सारी lines चल चुकी है, तो
अब control वहीं जाएगा जहा से function call हुआ था (a();)
और उस function (a()) की memory relieve हो जाएगी।




Now main function की 2nd line चलेगी

$II^{nd}$ line b(); ←

यहाँ function b call होगा जैसे ही call होगा, सबसे पहले RAM मे load होगा

now control b() ←

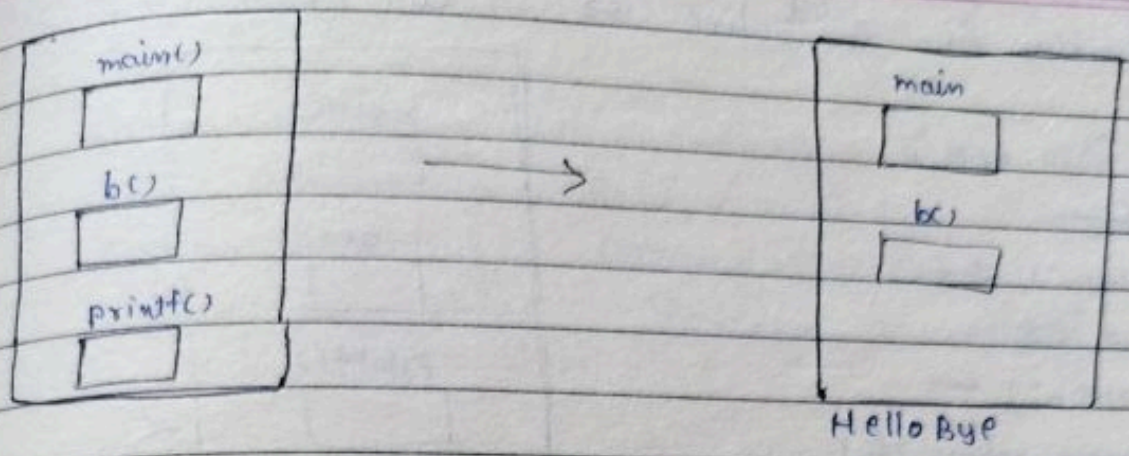
main function की 2nd line चल रही है इसका मतलब है b function की पुरी defination का चलना।

In b() → we have call printf(); , जैसे ही call होगा RAM मे load होगा printf की defination जहा कही भी होगी वो execute होगी और execution से Bye print हो जाएगा।

अब control exactly वापस वही आएगा जहाँ से function call हुआ था। ( printf("Bye"); ← )

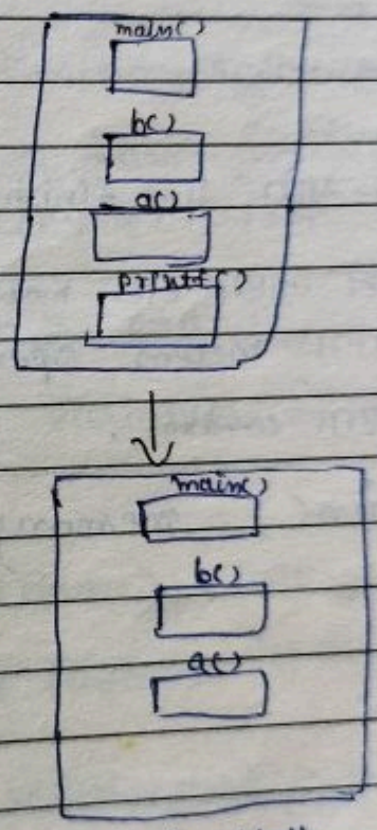
और printf function की memory relieve हो जाएगी।

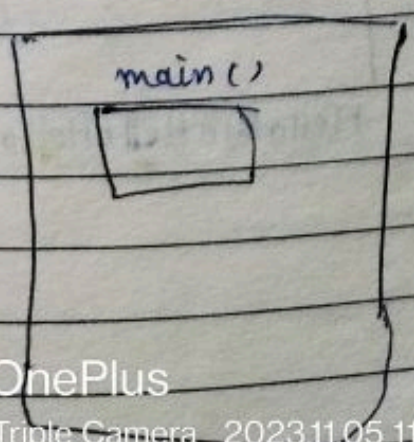
Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:36




अब b function की body मे दुसरी line चलेगी

```
a();
a()
printf ("Hello");
// Hello print होगा
printf ("Hello");
// printf() memory reliese
a की सारी line finish हो गयी
    a();
// a() memory reliese
b की सारी line finish हो गयी
    b();
// b() memory reliese
```




Hello Bye Hello

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:36

main function की III^rd line → a();

a load होगा RAM में

a() ←

printf("Hello"); ←

// Hello print होगा

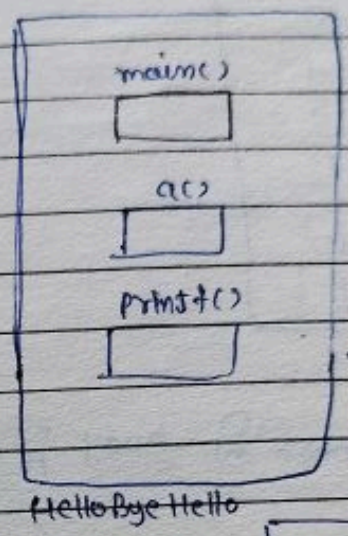
printf("Hello") ←

// printf memory reliese होगी।

a की सारी line चल ली है।

a(); ←

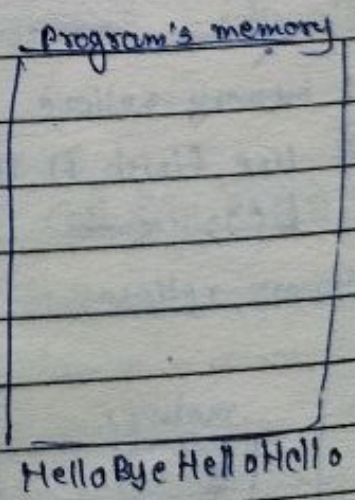
// a() memory reliese




main की भी सारी line finish हो चुकी हैं।

तो control वहाँ जाएगा जहा main function call हुआ था। means Operating System के पास जाएगा control.

और main() memory reliese




Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:37

Date: / / Page No.:

## Arithmetic Instruction ⇒

An instruction which is used to manipulate data using operators is known as arithmetic instruction

arithmetic instruction = 3 + 4

here, + → operator

3, 4 → operand/Data

Types of operators according to the number of operand.

1) Unary operators
2) Binary operators
3) Ternary operator

in brif

1. Unary operators + , - , ++ , -- , sizeof
2. Arithmetic operators + , - , * , / , %
3. Bitwise operators & , | , ^ , ~ , >> , <<
4. Relational operators < , > , <= , >= , == , !=
5. Logical Operators ! , && , ||
6. Conditional operators ?:
7. Assignment operators = , += , -= , *= , %= , /=

Unary operators ⇒ In this operators there is only one operand, that is why this are called unary operators

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:23

Date: __/__/__ Page No:

+, - ⇒ these are sign operators, we give them first priority it means we solve it firstly.

examples ⇒ +5
-7

++ ⇒ Increment operator, in this, operand that we use is only can be a variable, it can't be a constant

++3; // error
++x; // correct

type ⇒

x++; post increment
++x; pre increment

output of the operator is same
x = x+1

output is same but the difference is priority

++x; pre increment first priority
x++; post increment last priority

priority means the sequence of applying operators as in BODMAS rull.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:23

```
main()
{
    int x = 5, y;
    y = x++;
    printf("%d %d", x, y);
}
```

try to dry run this programe

in third line of our program there are two operators =, ++ assign and post increment.

| X | Y |
|---|---|
| 5 | |

And as we give last priority to post increment, first we will operate assign operator

| X | Y |
|---|---|
| 5 | 5 |

asignment operator ⇒ by using it the value in the variable on the right side will be copied by the variable on the left side 5 = x++

Now after it post increment operator will be aplied the post increment operator's operant is x, so the value of x will be x+1.

| X | Y |
|---|---|
| 6 | 5 |

now output instruction printf will run and the values of x y will be print as our output

6, 5

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:23

Date: __/__/__ Page No.

-- ⇒ decrement operators

--x ; pre decrement
x-- ; post decrement } x = x-1

--x ; first priority
x-- ; last priority

---

Sizeof () ⇒ in () we write operant, operant can be
(1) data type
(2) variable
(3) constant

(1)
```
int x ;                    o/p
x = sizeof(float);         4
x = sizeof(char);          1
x = sizeof(int);           4
x = sizeof(double);        8
```

(2)
```
int x, y ;        x = sizeof(k);   o/p 4
float k;          x = sizeof(ch);  1
char ch;          x = sizeof(d);   8
double d;         x = sizeof(x);   4
                  x = sizeof(y);   4
```

# by default integer constant is int type.

(3)
```
int x ;
x = sizeof(35);     4
x = sizeof('A');    4
sizeof(5.67);       8
```

'A' = 65

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:24

# Arithmetic Operators

* / % (first priority)
\+ − (second priority)

in C language * / % operators have equal priority

if the instruction is 3 / 5 * 3 then what will happen

The operator which is in left side will be operated first.
It means / (divide) will be operat first.

| Examples | | first priority |
|---|---|---|
| | a/b * c | / |
| | a*b/c | * |
| | a+b*c | * |
| | a−b/c | / |
| | a−b+c | − |
| | a%b+c | % |
| | a+b−c | + |

other eg.

| | | | |
|---|---|---|---|
| 3+4 | 7 | 5 % 2 | 1 |
| 3−4 | −1 | 17 % 4 | 1 |
| 3*4 | 12 | 23 % 5 | 3 |
| 3/4 | 0 quotient | 25 % 5 | 0 |
| 3/4.0 | 0.75 | 3 % 5 | 3 |
| 3.0/4 | 0.75 | | |
| 3.0/4.0 | 0.75 | 2.5 % 2 | error |

(remaindar after division (modulo division))

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:24

Date: 16.12.21 Page No.

In C language

Q. If the output of $\frac{x}{5}$ is 2, than can you say this thing that x is completely divisible by 5.

Ans ⇒ No, x, 5 are integers (int type data) that's why output is also int type.
here x may be 11, 10, 12, 13, 14 the output will be 2.

Q. if the output of x % 5 is 0, than can you say this thing that x is completely divisible by 5.

Ans → Yes, output will be zero only when x is divisible by 5. x can be 10, 15, 5, 20, 25 ---- ∞

Q. What will be the output of x/10

x without last digit.

| | |
|---|---|
| 123/10 | 12 |
| 25612/10 | 2561 |
| 25/10 | 2 |

Q. What will be the output of x % 10

last digit of x

| | |
|---|---|
| 3521 % 10 | 1 |
| 23 % 10 | 3 |
| 3 % 10 | 3 |

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:24

2. write a program which takes day number of a week and display unique greeting message for the day.

```
main()
{ int x;
  printf ("Enter the day number");
  scanf ("%d", &x);

  switch(x)
  {
   case 1 :
    printf ("Happy Monday");
    break;
    case 2 :
    printf ("Happy Tuesday");
    break;
    case 3 :
    printf ("Happy wednesday");
    break;
    case 4 :
    printf ("Happy Thursday");
    break;
    case 5 :
    printf ("Happy Friday");
    break;
    case 6 :
    printf ("Happy Saturday");
    break;
    case 7 :
    printf ("Happy Sunday");

   default :
    printf ("invalid input");
  }
  getch();
}
```

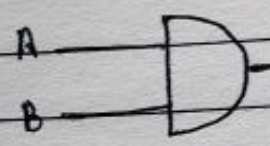
Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:34

# Bitwise Operators

| Operator | & | \| | ^ | ~ | >> | << |
|---|---|---|---|---|---|---|
| | and | or | x-OR | not | right shift | left shift |

**And**

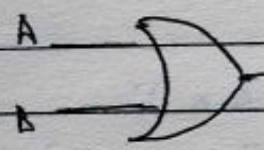
0 & 0 = 0
0 & 1 = 0
1 & 0 = 0
1 & 1 = 1




$Y = A \cdot B = A \& B$

**OR**

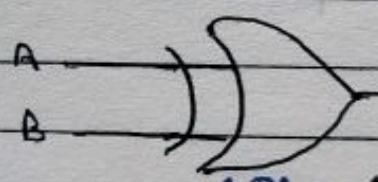
0 | 0 = 0
0 | 1 = 1
1 | 0 = 1
1 | 1 = 1




$Y = A+B = A|B$

**x-OR**

0 ^ 0 = 0
0 ^ 1 = 1
1 ^ 0 = 1
1 ^ 1 = 0

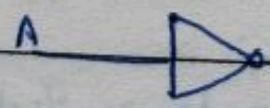



$Y = A \oplus B$

$A \oplus B = A \wedge B = \bar{A} \cdot \bar{B} + \bar{A} \cdot B$

**not**

~0 = 1
~1 = 0




$Y = \bar{A}$

$\sim A = B$
$\sim B = A$

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:24

2. Write a ~~program~~ menu driven program with following options:
a. Addition
b. Subtraction
c. Multiplication
d. Division
e. Exit

Date: / / Page No.

6.

```
main ()
{   int a, b, c, r ;
printf ("Enter two values");
scanf ("%d %d", &a, &b);
while (1)
{ printf ("Enter 1 for addition\n");
  printf ("Enter 2 for sub\n");
  printf ("Enter 3 for mult\n");
  printf ("Enter 4 for div\n");
  printf ("Enter 5 for exit");
  ~~printf~~ scanf ("%d", &c);
  printf ("%d", c);
switch (c)
{ case 1 :
  ~~prin~~ r = a+b;
  break;
  case 2 :
    r = a-b;
  break;
  case 3 :
    r = a*b;
  break;
  case 4 :
    r = a/b;
  break;
  case 5 :
    exit (0);
  default :
  printf ("invalid selection");
}
printf ("Result = %d\n", r);
getch ();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:34

Date: / / Page No.

# Binary Conversion

Trick ⇒ write the powers of two backwards (reverse)

512 256 128 64 32 16 8 4 2 1

अब हमे जिस नंबर का binary करना है, हम यह देखेंगे की वह नंबर 2 की पावर'स मे से किन-किन नंबरो का योग है।

जैसे 25 25 is the sum of 16, 8, 1

अब हमारा दिया गया नंबर जिन नंबर's का sum है, उनके नीचे हम 1, लिखेंगे और बाकी बीच मे या पीछे आ रहे numbers के नीचे 0 लिखेंगे।

| 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|
| | | | | 1 | 1 | 0 | 0 | 1 |

$(25)_{10} = (11001)_2$

| | 512 | 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|
| 61 | | | | | 1 | 1 | 1 | 1 | 0 | 1 |
| 92 | | | | 1 | 0 | 1 | 1 | 1 | 0 | 0 |
| 83 | | | | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| 33 | | | | | 1 | 0 | 0 | 0 | 0 | 1 |

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:24

Date: / / Page No.:

# Functions

1. Function is block of code which has some name for identification

```
f1()
{
  code
}
```

← Function definition

2. Function names must be unique in program

3. Function definition, function call and function declaration are three different things.

```
f1();
```

← Function call

4. Any C program can have any number of functions, but minimum one function must be there.
If there is only one function in the program then its name must be main().

5. One function name must be main in the program. main() function is entry point of your program. program execution always begin from main function.

6. Function naming rules are same as of variables.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:34

practice

```
int x, y;
scanf("%d %d", &x, &y);
```

Scanf() मे अगर दो variable का use हम कर रहे है तो दो बार हम %d formate spacifier use करते है। formate spacifier के बीच मे हम (,) नही लगाते है आप चाहे तो space दे सकते है और आप चाहे तो (\n) भी लगा सकते है पर (,) कभी नही। then (,) लगाकर पहले variable का adress लिखे फिर (,) second variable का।

Now - पहले code मे हमने variable की value fix की हुयी थी अब हम चाहते है कि variable की value user के द्वारा inter की जाए।

```
main()
{
    int x, y;
    scanf("%d %d", &x, &y);
    printf(" %d + %d = %d", x, y, x+y);
    getch();
}
```

अब जब Program run होगा तब blank screen दिखेगी (क्योकि जैसे getch() के कारण screen रुक रही थी वैसे ही scanf() के कारण भी रुकेगी क्योकि ये ~~एक~~ दोनो input instruction है।

हमे जो data देना है वो हम type करेंगे फिर हम enter / space / tabe मे से कुछ भी दबाएंगे तब जाकर हमारा पहला data complete होगा और फिर दूसरा data लिखिये। आखिरी data के बाद enter press करना होगा।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:23

# Output instruction

जो normal user होगा, जिसको हमारा code नहीं दिख रहा है और उसे समझ मे भी नहीं आता है, उसके पास के सिर्फ हमारी .exe file है।

तो वो blank screen को देखकर confuse होगा। उसके लिए हम एक mesege print कराते हैं। जिसके लिए हम code मे एक लाइन printf() {Pridefined function} output instruction लिखेंगे।

```
main()
{
  int a,b ;
  printf (" Enter two numbers") ;
  scanf ("%d %d", &a, &b);
  printf ("%d + %d = %d", a , b , a+b);
  getch () ;
}
```

Now जब program run होगा तब सबसे पहली instruction (output) चलेगी और हमे screen पर वह message लिखा हुआ दिखेगा जिसे हमने printf() मे लिखा है।

If we want third variable then we can write

```
main()
{
  int a, b, c ;
  printf (" Enter two number") ;
  scanf ("%d %d", &a , &b) ;
  c = a+b ;
  printf ("Sum is %d", c) ;
  getch ()
}
```

or printf ("%d + %d = %d", a,b

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:23

Input instruction

scanf() predefined function

scanf() की help से keyboard से data जो इंटर होता है उसे हम एक variable मे पहुचाते है।

# getch() की help से हम एक charector देते है। but

# scanf() की help से हम एक data देते है। जो कि एक से ज्यादा keys से मिलकर बनता है।
जैसे आपकी age 19

⇒ scanf() की help से data input लेना और उसे variable मे लाना।

```
int x ;
scanf ("%d" ,&x );
```

here we can use only formate spacifier only in double codes. not other message is allow as we use in printf()

& ⇒ adress of (&), it is an m persent symbol

we use it in only scanf(), not in printf()

scanf() से जो key press की जाती है वो screen पर दिखते है
getch() से press की गयी key दिखाई नही देती screen पर

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:22

Switch case, goto control

1. the constant after the case can not be same for two. we can say that duplicate case is not possible.
Constant after each case should be unique.

2. it is not necessary that the constant after case will be in sequance, means 1, 2, 3 is not necess.

3. We can not use real constant after case
means case 4.3 error
case 4 ✓ int
case 'A' ✓ char

4. It is not neccessary to write default: in last we can write where ever we want

5. also write constant after case, we can not use variable.
we can use expration that only includes constants (e.g. 40+3)
be we can't (40+y)
(3>0) → it's result will be

6. we can also use continuty here, means if when you enter 1-10 then case execute for it

case 1 ... 10 ; → elipsis or continuty operator

case 'a' ... 'z' case 'A' ... 'Z'

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:32

Date: / / Page No.:

6. 7 atthease we can not write values by using,
Switch ( ) → we can write here veriable, expration
constant( integer )

```
switch ( expression)
{
   case constant :
        code
   case constant :
        code
   case constant :
        code
}
```

7.

for default :
we simply add

```
default :
code
```

```
default :
printf ("invailid input");
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

```
main()
{
    int x;
    printf ("Enter a number");
    scanf ("%d", &x);
    switch (x)
    {
      case 43 :
        printf ("Sunday");
      case -23 :
        printf ("Monday");
      case 0 :
        printf ("Tuesday");
      case 2 :
        printf ("Wednesday");
    }
    getch();
}
```

if we want to stop our program after any case we can use **break** after the code

output
Wednesday

if x is entered 2 then

if x is intered -23 then Monday Tuesday Wednesday

if x is intered 53 then screen will remain blank and after getting a charactor because of getch() program will end.

For this problem we use **default:**

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

default : it is a keyword.

if the value is not suitable for any case then the screen remains blank but here we want to print something appropriate

जिसका कोई case नहीं होता उसका default; होता है।

- we can't use continue, it is used only in loop.
  we can use break in switch case and also in loop.

elipsis or continuty operator.

```
main ()
{ int x ;
  printf ("Enter a number");
  scanf ("%d", &x);
  switch (x)
  {
   case 1 ... 10 :
   printf ("Sunday");
   case -23 :
   printf ("Monday");
   case 25 :
   printf ("Tuesday");
  default:
   printf (" invailid input");
  }
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

```
# include <stdio.h>
main ()
{
  int x, ~~y=0~~ ;
while(1)
{ printf (" Enter a charactor ");
  fflush (stdin);
  Scanf ("%c", &x);

  switch (x)
  {
    case 'a'...'z' :
      printf ("Small alphabet");
      break;
    case 'A'...'Z' :
      printf (" capital alphabet");
      break;
    default :
      printf (" invailid input");
  } }
  getch();
}
```

- It's a simple program of continuity operator.
- But if we use here while loop then firstly we have to write # include <stdio.h> and in between fflush (stdin);

here while (1)

this means non zero value so the program will never end.

for ending this program can understand in another program

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

Date: / / Page No.:

6.

```
main()
{
    int x ;
    while (1)
    {
        printf ("Enter a number or 0 to end");
        scanf ("%d", &x);
        switch (x)
        {
            case 1 ... 10 :
                printf ("1 to 10");
                break;
            case 11 ... 20 :
                printf ("11 to 20");
                break;
            case 0 :
                printf ("zero");
                exit (0);
        }
    }
    getch ();
}
```

# exit is a predefined function, which directly ends the program

We write exit in two type

1. exit (0); (normally used)
   it means normal termination of program.
2. exit (1);
   it means we have end our program forcefully because of any abnormality.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

Date: / / Page No.:

goto control instruction

सबसे बेकार चीज़ है।
never-ever use it
इससे logics poor होते है
आपके Sylabus में है
इसलिए पढ़ना है।

goto is a keyword

```
function()
{
    code
    goto a1;
    code
    code
a1:
    code
    code
```

goto is for jumping. jumping control from one place to another place and places can be identified ~~by~~ with this name (eg. a1:) this is called lable.

lable → a1 : , we use colon
it have no declaration. we can use various lables in our program
By the use of goto we can move in forward direction as well as in backward direction

Whenevere we write goto we use it with if()

```
function()
{
    code
    if()
    goto a1;
    code
    code
a1:
    code
    code
    if()
    goto a1;
    code
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:33

Date: / / Page No.

6. 1 Write a program which takes month number as an input and display number of days in that month.

```
main()
{ int mn;
printf ("Enter the month number");
scanf ("%d", &mn);

switch (mn)
{
case 1 :
case 3 :
case 5 :
case 7 :
case 8 :
case 10 :
case 12 :
printf (" 31 days");
break;
case 2 :
printf (" 29/28 days");
break:
case 4:
case 6:
case 9:
case 11:
} printf (" 30 days");
} getch();
```

2. Write unique

Date: / / Page No.:

1. C is a block structured programing Language
2. A C program can have any number of bloks
3. you can have inner blocks too
4. outer blocks are usually functions
5. Function is a block of code, which has some name

```
f1()
{
}
```

here f(1) is a name of function

do not put semicolon here (;)

6. No keyword is a function.
7. A C program can have any number of functions
8. For naming a function we can use the same rules which we apply for naming a variable
9. In a C program, all function must have unique name
10. Even in the smalles C program, there must be one function, and its name must be main()
11. Declaration statements can be written either inside a block or outside a block.
12. Action statement can only written inside the block.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

Date: 9/12/21 Page No.:

Instructions - Action statement

1. Input output Instruction
2. Arithmetic instruction
3. Control instruction

⊙ Output instruction ⇒ Program में लिखी गई वह instruction जिसके exicute होने से keyboard से enter किया गया data, हमारे program के अंदर variable में आता है, वह Output instruction है।

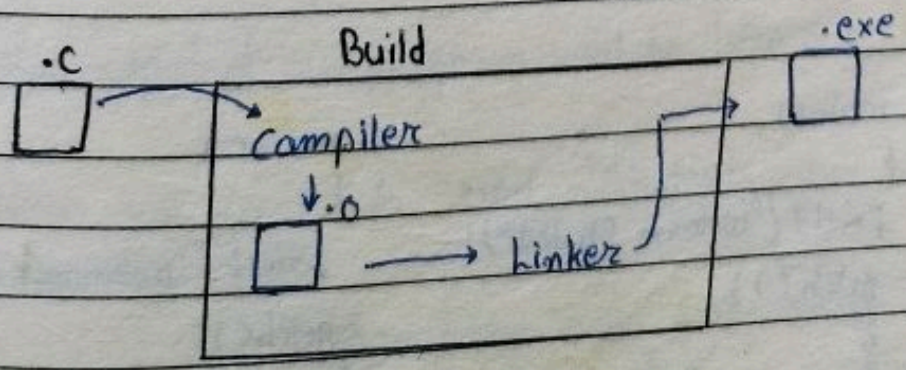
printf() Predefined Function

We use it
1. to print text on the screen
2. to print value of a variable or expression.

Ist program

```
main()
{ printf("WELCOME");
}
```

if we run this program by dable cleaking on .exe file this program ends very quick. we can see this program's output in our code bloks only. for this we will use getch().




Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

Date: / / Page No.:

getch() is a predefined function

its work is to take a charactor from the keyboard.
We use this as a trick - to hold the screen

Now ⇒

```
main()
{
    printf("WELCOME");
    getch();
}
```

11/12/21

Now - we want to print welcome student on our screen.

अगर हम उपर वाले program को modify करके बिना build किए हुए Run करेंगे तो, system हमे पहले से बनी हुई .exe file को Run करके दिखाएगा।

और अगर हम नया function बना रहे है तो इसे Run करने पर यह अपने आप पहले build होगा फिर Run होगा।

```
main()
{
printf("Welcome");
Printf("Students");
getch();
}
```

```
main()
{
printf("Welcome Students");
getch();
}
```

```
main()
{
printf("welcomestudents");
getch();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

Date: / / Page No.:

इन तीनो programs का output क्रमशः ⇒ WelcomeStudents
Welcome students
WelcomeStudents

अब अगर इस program के output को बारीकी से समझना चाहते है तो हम getch() का उपयोग program में कहीं भी कर सकते है।

हमारे program के output मे कोई भी word वहीं से print होता है जहाँ कर्सर होता है।

```
main()
{
    getch();
    printf("Welcome");
    getch();
    printf("students");
    getch();
}
```

if we try to understand the process of programming - firstly there will be a blank screen, here programs first action statment is working, when we type a charector it will end and the second statment will work and it will print welcome, then we type a charector, and students will be print and then we type a charector and program will be end.

And the output of our program will be like this. ⇒ WelcomeStudents

But Now we want to print 'Students' and 'Welcome' in separate lines.

So here we use new line character

back slash n ⇒ \n

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

we use it as printf ("\n") ~~for~~

When we use it \n ~~print~~ do not print either we see its effect - and the cursor will move to the starting of the new line.

```
main()
{ getch();
  printf("Welcome");
  printf("\n");
  printf("Students");
  getch();
}
```

(or)

```
main()
{
printf("welcome\n");
printf("students");
}
```

(or)

```
main()
{
  Printf("Welcome\n Students");
  getch();
}
```

The output of above 3 codes will be same

~~Escape Seq~~

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:22

Date: / / Page No.:

Escape Sequences — Escape sequences are those charectors which starts with \n.

These are spacial charectors

| | |
|---|---|
| \n | New line |
| \t | Tab space |
| \b | Backspace |
| \r | Carriage Return |
| \\ | print \ |
| \" | print " |
| \' | print ' |

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:22

printf() can be used to print value of a variable or expression

```
main()
{
    int a = 5 ;
    printf("a");
    getch();
}
```

here we have write a is "" quats and the output will be → a so this is not ~~wr~~ right

```
main()
{
    int a = 5 ;
    printf(a);
    getch();
}
```

→ This is also wrong.

Now what is ~~wr~~ right

```
main()
{
    int a = 5 ;
    printf("%d", a);
    getch()
}
```

Format Specifier → %d int
%c char
%f float
%H double

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:22

```
main()
{
    int i, j ;

  for ( i=1 ; i<=4 ; i++)
{
   for ( j=1 ; j<=5 , j++ )
   {
      if()
       printf ("*");
      else
         printf ("$");
   }
      printf ("\n");
  }
    getch();
}
```

| i | j | i=2 | i=3 |
|---|---|---|---|
| 1<=4 | 2<=4 | 3<=4 | |
| 1<=5.... | 1<=4.... | 1<=4.... | |
| if (j<=1) | if(1<=2) | if(1<=3) | |

```
*
* *  -  -
* *  *  -
* *  *  *
```

i ii

4<=4
1<=4
if (1<=4)

## Star pattern

j colomn

| | | 1 | 2 | 3 | 4 |
|---|---|---|---|---|---|
| | 1 | * | | | |
| i | 2 | * | * | | |
| rows | 3 | * | * | * | |
| | 4 | * | * | * | * |

| i | j |
|---|---|
| 1 | 1 |
| 2 | 1 2 |
| 3 | 1 2 3 |
| 4 | 1 2 3 4 |

| i | j |
|---|---|
| 1 | j<=1 |
| 2 | j<=2 |
| 3 | j<=3 |
| 4 | j<=4 |

$\boxed{j <= i}$

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:32

Date: / / Page No.:

now we want to print - a = 6

// a = 6

```
main ()
{
    int a = 6 ;
  printf ( "a = %d" , a ) ;
  getch ( ) ;
}
```

now we want to print - // value of a is 6 and b is 7

```
main ( )
{
    int a = 6 , b = 7 ;
  printf ( " value of a is %d and b is %d", a, b) ;
  getch ( ) ;
}
```

now // 6+7 = 13

here a+b is expression

```
main ( )
{
    int a = 6 , b = 7 ;
  printf ( " %d + %d = %d", a, b, a+b ) ;
  getch ();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:22

```
main()
{
    int i, j ;

    for(i=1; i<=4; i++)
{
    for ( j=1 ; j<=4 ; j++)
    {
     if ( j<=i)
       pritf ("*");
     else
          printf (" ");
     }
      printf ("\n");
    }

    getch();
}
```

```
main()
{   int i, j, lines;
    printf ("Enter the number of lines");
    scanf ("%d", &lines);
 for (i=1, i<=lines; i++)
 {
  for (j=1; j<=lines; j++)
  {
   if ( j<=i)
     printf ("*");
   else
        printf (" ");
   }
     printf("\n");
  }  getch();
 }
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:32

Date: 9/12/2021 Page No.:

char m = 'A';

# Character Encoding technique → ASCII

(American standard code for information interchange)

| | | |
|---|---|---|
| | | 0 |
| space → | ' ' | 32 |
| | '0' | 48 |
| | '1' | 49 |
| | '2' | 50 |
| | ⋮ | |
| | '9' | 57 |
| | '@' | 64 |
| | 'A' | 65 |
| | 'B' | 66 |
| | ⋮ | |
| | 'Z' | 90 |
| | | |
| | 'a' | 97 |
| | 'b' | 98 |
| | 'z' | 122 |
| | | |
| | | 255 |

Single ~~qod~~ qoutes is used only on the character contant.
Single qoutes is not used on ASCII code

char m = 'A';
⇒ char m = 65;

int a = 65;
int a = 'A';

# Compiler will consider 'A' as 65
That's why we can rewrite our declaration statement as given above.

Total charecter constants = 256
& ASCII codes are 0 to 255

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

Date: __/__/____ Page No.:

# Star Pattern

```
main()
{ printf("*");
  getch();
}
```

```
main()
{ printf("*****");
  getch();
}
```

```
main()
{ int i=1;
  while (i<=5)
  { printf("*****");
    i++;
  }
  getch();
}
```

```
main()
{ int j;
  for (j=1; j<=5; j++)
  { printf("*");
  }
  getch();
}
```

```
main()
{ int i, j;

  for (i=1; i<=4; i++)
  {
    for (j=1; j<=5; j++)
    { printf("*");
    }
    printf("\n");
  }
  getch();
}
```

```
* * * * *
* * * * *
* * * * *
* * * * *
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:31

Date: / / Page No.

Keywords = Reserved words = 32 in C

| | | | |
|---|---|---|---|
| auto | double | if | sizeof |
| break | enum | long | short |
| case | else | return | typedef |
| char | extern | register | union |
| continue | for | static | unsigned |
| const | float | struct | void |
| do | goto | signed | volatile |
| default | int | switch | while |

Declaration Statement - Data type declaration instruction

instruction = Statement = Command

1 Declaration Statement

Data type ⇒ int
Primitive char
float
double
void

C is a case sensitive language. At the end of each statment we have to write ;

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:20

Date: / / Page No.:

```
main()
{
    int a, b, c;
    printf ("Enter three numbers");
    scanf ("%d %d %d", &a, &b, &c);
if (a>=b && a>=c)
    printf ("%d", a);
    else
    {   if (b>=a && b>=c)
        printf ("%d", b);
        else
        printf ("%d", c);
    }
  getch();
}
```

```
main()
{   int a, b, c;
    printf ("Enter three numbers");
    scanf ("%d %d %d", &a, &b, &c);
  if (a>=b && a>=c)
    printf ("%d", a);
else { if (b>=c)
        printf ("%d", b);
      else
        printf ("%d", c);
    }
  getch();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:28

1 byte = 8 bits

int a, b ; a, b is variable and a, b is int type variable

char m ; m is variable and m is char type

float k ; k is a variable and k is float type variable

double d1 ; d1 is a variable and its double type variable

in int type variable we can store

int type variable ⇒ 4 bytes → integer constant

char type variable ⇒ 1 bytes → Character constant

float type variable ⇒ 4 bytes → Real constant

double type variable ⇒ 8 bytes → Real constant

float is single precision
double is double precision
double is more accurate

float k = 0.7 ;
double d1 = 0.7 ;

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:21

```
main()
{   int a,b,c;
    printf ("~~%d %d~~ Enter three numbers");
    scanf ("%d %d %d", &a, &b, &c);
    if ( a>=b && a>=c)
      printf ("%d", a);
    else
        { if ( b>c)
            printf ("%d", b);
          else
            printf ("%d", c);
        }
    getch();
}
```

```
main()
{ int a,b,c;
  printf ("Enter three numbers");
  scanf ("%d %d %d", &a, &b, &c);
  if (a>b)
    { if (a>c)
        pritf ("%d", a);
      else
        printf ("%d", c);
    }
    else
    { if (b>c)
        printf ("%d", b);
      else
        printf ("%d", c);
    } getch();
}
```

```
main()
{ int a,b,c;
  printf ("Enter three numbers");
  scanf ("%d %d %d, &a, &b, &c);
  if (a>b)
    a>c ? printf("%d" a): printf("%d", c);
  else
    b>c ? printf("%d", b) : printf("%d", c);
  getch();
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:28

Date: __/__/__ Page No.:

```
main()
{
    int a,b,c;
    printf ("Enter three numbers);
    scanf ("%d %d %d", &a, &b, &c);

    a>b ? a>c ? printf("%d",a) : printf("%d",c): b>c ? printf("%d",b): printf(%d,c);
    getch();
}
```

```
main()
{
    int a,b,c;
    printf ("Enter three numbers");
    scanf ("%d %d %d", &a, &b, &c);

printf ("%d", a>b ? a>c ? a:b : b>c ? b:c);
    getch();
}
```

**Preprocessor** :- when we build our program firstly preprocessor program runs and read .c file. in our .c file line that have starting symbol # will be read by preprocessor

**# preprocessor directive**

whatever preprocessor does is not a part of c language. c language starts where compiler strats its work

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:28

```
main ( )
{ int i = 1 ;
  while ( i <= 10 )
  { printf ("MySirG");
  i += 2
  }
```

1 <= 10 , 3 <= 10 , 5 <= 10 , 7 <= 10 , 9 <= 10

5 times this will be true and after it this will be false

```
main ( )
{ int i = 1 ; //initialization
  while ( i <= 10)   // condition
  { printf ("MySirG");
  i += 2;            // flow
  }
}
```

Date: / / Page No.

## break

1. break is a keyword.
2. it can be used only in the body of loop or switch
3. it terminates the execution of loop

```
while ( condition )
{
    code
    code
    if ( condition )
        break ;
    code
}
```

## Continue

1. continue is a keyword.
2. It can only be used in the body of loop
3. it transfers the control to next iteration.

```
while ( condition )
{
    code
    code
    if ( condition )
        continue ;
    code
}
```

# the continue statement skip some code inside the loop and continues with the next iteration.

# It is mainly used for a condition so that we can skip some lines of code for a particular condition

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:29

Entry Control Loop

while

```
while (condition)
{
  code
  code
}
```

Exit Control loop

do while

```
do
{
    code
    code
} while (condition);
```

```
int i = 1;
while (i <= 10)
{
  printf ("%d", i);
  i++;
}
```

```
int i = 1;
do
{
    printf ("%d", i);
    i++;
} while (i <= 10);
```

9 times

| | | |
|---|---|---|
| Condition cheak | 11 | 10 |
| Condition true | 10 | 9 |
| Condition false at | 11 <= 10 | 11 <= 10 |
| output will be | printed 1-10 | printed 1-10 |

The difference between while and do while is ⇒

if the condition is false
- while's code will not run even once.
- do while's code will run once in every situation

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:29

Date: / / Page No.:

# Entry Control Loop

## for

```
for ( ; ; )
{
  code
  code
}
```

```
for ( initialization; condition; flow)
{
  code
  code
}
```

```
int i;
for ( i=1 ; i<=10 ; i++ )
{
  printf ("%d", i);
}
```

```
for (i=1 ; i<=10 ; i++)
{
  printf ("%d", i);
}
```

T F

we can write ⇒

```
int i=1;
for ( ; ; )
{
  if (i==11)
  break;
printf ("%d", i);
i++;
}
```

infinite loop

```
int i=1;
for( ; ; )
{
  printf ("%d", i);
  i++;
}
```

we can write ⇒

```
while ( )
{
  for ( ; ; )
  {
  }
}
```

# you can write any loop inside any loop.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:30

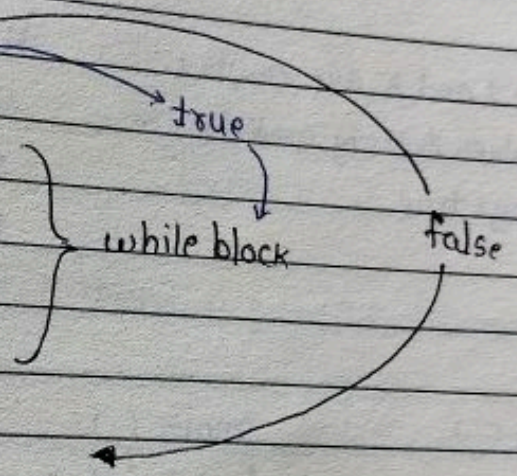
while

Iterative Control instruction | loop

methods of implementation ⇒ 1) while
2) do while
3) for

while is a keyword

```
while (condition)
{
    code
    code
}
```




condition of while is true → Control while block के अंदर आएगा → while block के अंदर का code execute होगा → while की body end होने पर → Control वापस while के पास पहुंचेगा → then again condition check होगी → फिर से true है तो फिर से block के अंदर आएगा → जब condition false हो जाएगी तो loop end हो जाएगा।

iterative means repeat

repeatation हो रहा है इसलिए ही इसे loop कहते हैं।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:30

3

Date: / / Page No.:

1) Read and understand question
2) Test case
3) Dry run

1) Read and understand question.
2) imagine output screen.
3) take a example

# if you are watching a solution of a problem then that problem is worthless for you.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:19

Date: / / Page No.:

If you are not born with a silver spoon then skillset is something that will help you improve your life & becoming independent.

Be very impatient and tought to yourself
Be very patient and kind to others.

Without discipline, you are nothing
Embrace ideas, however always insist on actions and executions.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:19

Date: / / Page No.:

Kernel ⇒ kernel is a program that directly interact with hardware. The kernal is an important part of the operating system.

System software ⇒ It is also a program. System software are device drivers. System software interacts with kernal.

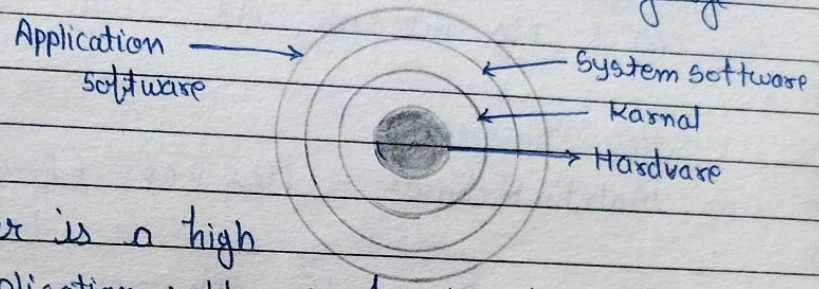
Application software ⇒ Calculator, Microsoft office, point, word, zoom,

Application software's code interact with system software then system software's code/program interact with kernal and then kernal's code/program interact with hardware

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:19

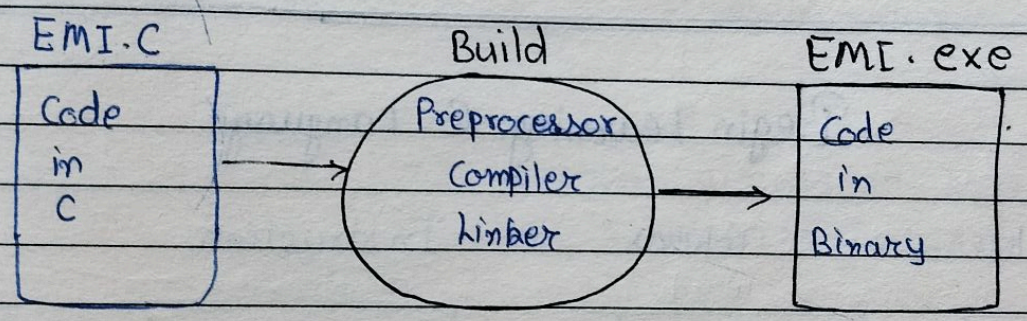
Date: 27/11/21 Page No.:

C is middle level language = High level + Low level language




High level :- Calculater is a high level application software. In this the coder has no mean with the hardware.

Low level :- for karnal and system software's coading low level lenguage is required.




Source File

Executable file (Software)

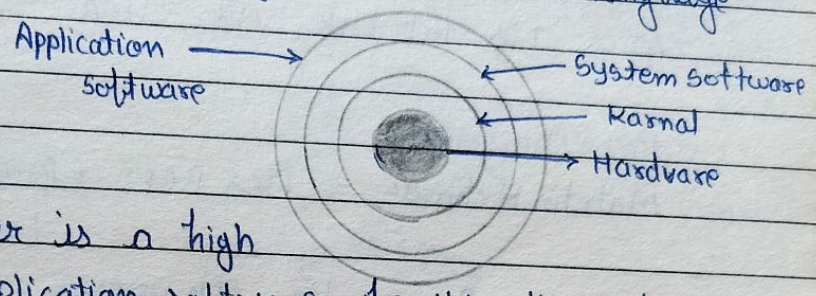
1. Create a source file and write a source code in C lengu
2. Build software

IDE = Integrated Development Enviroment

Code Editor
Build Tools (Preprocessor, compiler, linker, Debugging Tool

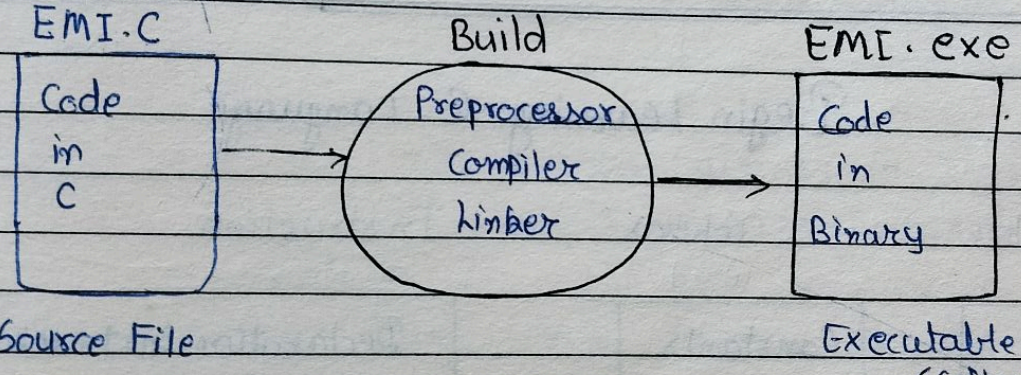
Date: 27/11/21 Page No.:

C is middle level language = High level + Low level language




High level :- Calculater is a high level application software. In this the coder has no mean with the hardware.

Low level :- for karnal and system software's coading low level language is required.




1. Create a source file and write a source code in C lengu
2. Build software

IDE = Integrated Development Enviroment

Code Editor
Build Tools (Preprocessor, compiler, linker, Debugging Tool

Date: 30/11/21 Page No.:

# History of C language

AT & T's Bell Lab, NJ, USA

Ken Thompson
Martin Richards - 1966 BCPL (Basic Combined Programing Language)

1969 - B - Ken Thompson
1972 - C - Dennis Ritchie

Unix opreting system

---

# Begin Learning C Language

| Basic Letter | Tokens (Word) | Instruction (Sentence) |
|---|---|---|
| a to z<br>A to Z<br>0 to 9<br>% & ( [ ] \<br>: " , ? +<br>= , all keys of keybord | Constants<br>Variables<br>Keywords<br>Function<br>Macro<br>Label | - Declaration instruction<br>- Action statement |

syntact - grammar
error - mistake

syntactical error - grammatical mistake

Program

(Set of instructions)

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:20

Date: / / Page No.:

Constants = Data = Constants = Information

| Primary Constants | Secondary Constants |
|---|---|
| - Integer - 23, 95, -234, 0 | Array. |
| - Real - 3.4, -0.06, 4.0 | String. |
| -Character - 'single letter' | Pointer |
| | structure |
| | union |
| | enumerator |

Integer = 5, -30, 10000, 9131044717, 8574104986 12
0, -1, 12, 3, - 9954957863

Real - 3.0, .0, -7.3, 0.23, 666.25

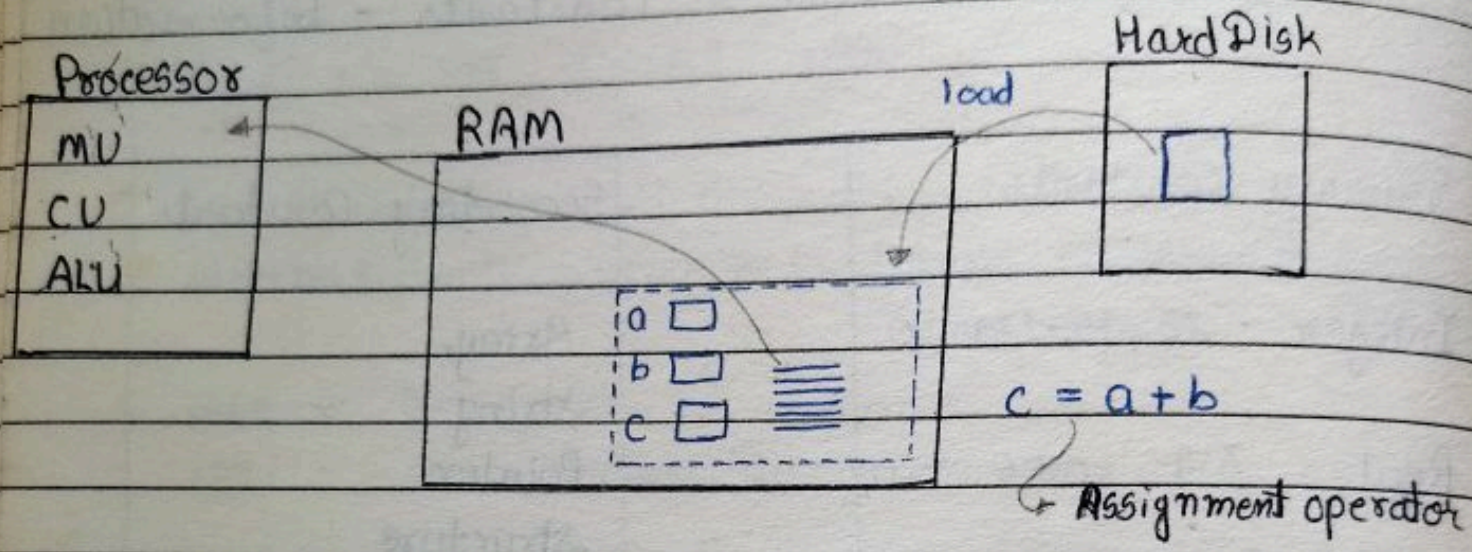
Character - ' ', '_', '?', '5', 'A', 'z', 'a'
'*', '/', '.'

String - "Bhopal" "4.3"

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:20

Date: 2/12/21 Page No.:

# Variables




Variables are memory locations where we store our data.

Variable name is any combination of alphabets, digits and underscore No other symbol is allowed.
Variable name can not start with a digit

A, a3, x-4, da, da_23_ab

x-1 ⇒ it cannot be a variable name because we have used hyfen (-) and in variable name no other symbol is allowe expect underscore.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:20

Date: / / Page No.:

Now we can also write ~~it as~~ ⇒ this code as ⇒

```
main ()
{ int year;                                  // 1997, 2020
  printf ("Enter year");
  scanf ("%d", &year);

  if ( year % 400 == 0 || year % 100 && year % 4 == 0)
     printf ("Leap year");
  else
     printf ("Non leap year");

  getch ();
}
```

Write a program to find greater among three numbers

```
main ()
{   int a, b, c;
    printf ("Enter three numbers");
    scanf ("%d %d %d", &a, &b, &c);
   if (a >= b && a >= c)
   printf ("%d", a);
   else
       { if (b >= a && b >= c)
           printf ("%d", b);
         else
             { if (c >= a && c >= b)
                 printf ("%d", c);
             }
       }
}
```

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:28

we can also write it as

```
main()
{ int n ;
  printf ("Enter the number");
  scanf ("%d", &n);
printf ( n > 0 ? "Positive" : "Non positive");
  getch ();
}
```

else if lader

```
if (condition)
    code
else
    {   if (condition)
            code
        else { if (condition)
                code
               else { if (condition)
                       code
                    else
                       code
                    }
             }
    }
```

else if lader

```
if (condition)
  code
else if (condition)
  code
else if (condition)
  code
else if (condition)
  code
else
  code
```

we can write it as (because if else is one action statment)

The style of writting the code in this way is called else if ladder.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:26

year

Sentuary → ( % 400 == 0 ) → Leap year
Sentuary → not divisible → Noneleapyear

Non sentury → ( % 4 == 0 ) → Leap
Non sentury → not divisible → Non leap

**nested if else** ⇒ if के अंदर if else
else के अंदर if else

```
main()
{
   int year;
   printf ("enter year");
   scanf ("%d", &year);
if (year % 100 == 0)
  {
    if ( year % 400 == 0)
     printf ("Leap year");
    else
     printf ("Non leap year");
  }
else
  { if (year % 4 == 0)
     printf ("Leap year");
    else
     print ("Non leap year");
  }
 getch();

}
```

This type of code ↑ is called nested if else.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:27

Date: / / Page No.:

30 sep. 2022

Test for certificate

Objective type test

each quation has 4 options 1 is right option

no nagative marking . in certificate marks will not display

have to clear test ⇒ passing marks 50 %

for downloading certificate ⇒ click on certificates then after clearing the test there will be available a download button

if fail in test → then you must have to reset the test

test reset request ⇒ go on profile ⇒ support ⇒ Raise ticket

in description write → please reset my test
Batch code SB
subject → issue in test
category → other technical issue

Then click on Raise Ticket

it will take 24 hours after it you will get notified

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:18

Date: / / Page No.:

| | 512 | 256 | 128 | 64 | 32 | 16 | 8 | 4 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|
| 33 → | | | | | 1 | 0 | 0 | 0 | 0 | 1 |
| 83 → | | | | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| | | | | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

int x ;
x = 33 & 83 ;

output = $(1)_{10}$

x = 55|91

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 55 → | | 1 | 1 | 0 | 1 | 1 | 1 |
| 91 → | 1 | 0 | 1 | 1 | 0 | 1 | 1 |
| | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

output $(127)_{10}$

x = 62^39

| | | | | | | |
|---|---|---|---|---|---|---|
| 62 → | 1 | 1 | 1 | 1 | 1 | 0 |
| 39 → | 1 | 0 | 1 | 0 | 0 | 1 |
| | 0 | 1 | 0 | 1 | 1 | 1 |

output $(23)_{10}$

x = ~55

| | | | | | | |
|---|---|---|---|---|---|---|
| 55 → | 1 | 1 | 0 | 1 | 1 | 1 |
| | 0 | 0 | 1 | 0 | 0 | 0 |

output $(8)_{10}$

x = 75>>2

75 → 1 0 0 1 0 1 1

output = $(18)_{10}$

>>2 → 0 0 1 0 0 1 0 1 1

x = 12 << 3

1 1 0 0

1 1 0 0 0 0 0

output = $(96)_{10}$

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:25

Obs ⇒ screen recorder

Date: / / Page No.:

Home key - will move your cursor to the beginning line

End key - will move your cursor to the end of line.

Delete key - The letters of the text will be erased from the right side of the cursor.

ctrl + Back space - will erase complete word

windows + V ⇒ all your previous copyed thing will be here.
↳ you have to enable it than it will work

windows + [→] ⇒ for opening two windows side by side.
right arrow key
you can do this thing by your mouse cursur just catch the window and put it on right side it will automatic give you the option

if in your tars bar ⇒ crome, fiEx, VS code - - - -
(in this condition if you want to open crome then press ⊞ + 1
if VS code then press ⊞ + 3

Windos + D ⇒ your all windows will be minimize and you will see your desktop / you can do it through mose right side extreme lower

Windows + tabe ⇒ you can see all your open windows

alt + tabe ⇒ you can open any your previous open window

I am a good boy!

if your cursur is on end point and you press Shift + End then all your text will be selected.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:18

Date: __/__/__ Page No.:

# Relational Operators / Comparision operators

| | | |
|---|---|---|
| < | less than | inequality operators — $I^{st}$ priority |
| > | greater than | |
| <= | less than or equal to | |
| >= | greater than or equal to | |
| == | equal to | equality operators — $II^{nd}$ priority |
| != | not equal to | |

Result of Relational operators → 0 or 1 , 0 if false relation
1 if true relation

| | output |
|---|---|
| x = 5 < 4 ; | 0 |
| x = 3 != 4 ; | 1 |
| x = 4 < 4 ; | 0 |
| x = 5 > 4 > 3 ; | 0 → x = 1 > 3 ; fals → 0 |
| x = 0 != 0 < 1 ; | 1 |

# Logical Operators

| | |
|---|---|
| ! | Not operator ( Unary ) Top prioriit > Rela., Bitwi. Arith etc. |
| && | And |
| \|\| | OR |

! Not operator ⇒ every non zero value is true

!0 → 0 → false
!1 → 1 → true
!-1 → 1 → true
!5 → 1 → true

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:25

Date: / / Page No.:

I am a good / boy.

if your cursure ↗ is here and you press Shift + Home i am a good will be selected.

if yo press Shift + End boy. will be selected.

Windows + P ⇒ in case (😐) you have two connected monitor you can control it.

windows + L ⇒ For locking your pc

mouse whele. you can press it and open any search result in which tabe you wcant.

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:19

Date: / / Page No.:

dry run

$x = !5 > -2$ ; $x = 1 > -2$ ; true output 1

output = 1

$x = !(5 > -2)$ ;

And && $x > 0$ && $y > 0$

exp1 && exp2

T && T = T

T && F = F

F && X = F

OR || exp1 || exp2

F || F = F

F || T = T

T || X = T

Assignment operators - = it prints the right side value in left side

a = 4

assignment operator में left side में हमेशा variable होता है।

memory location / Container

int x = 5;

x = x+3;

content

4 = a ; error

3 = 4 ; error

variable =

memory location

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:25

Date: / / Page No.:

# Compound Assignment Operation

+= -= *= /= %= >>= &=

x += 3 ; x = x+3

x* = 5 ; x = x*5

x& = 8 ; x = x&8

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:25

Date: / / Page No.:

## Control instruction ⇒

We have written a program of 7 lines and we want that after our 3rd line, 7th line of our code will run. means we do not want to run 4th, 5th and 6th line of our code.

Why - e.g. ATM machine. The code have written once for this software.

In this software there is a instruction written for withdraw money. But if you do not want to withdraw money and you just want to cheak your balance.

may be in futur you want to withdraw money but currently you just want to cheak balance.

That is why. control instructions are important

Control instruction -

1. Decision Control
2. Iterative Control
3. Switch case control
4. goto control

Decision Control instruction or Selection Control instruction

if

if else

Conditional operator

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:26

Keyword and keyword is never called function

if ( ) Expration — condition (T or F)

```
{
    code
    code
}
```

if block

if true → if block will run

if false → if block will not run ✗

```
main()
{
  line 1 ;
  line 2 ;
  line 3 ;
  line 4 ;
  line 5 ;
  line 6 ;
  line 7 ;
  line 8 ;
  line 9 ;
  line 10 ;
}
```

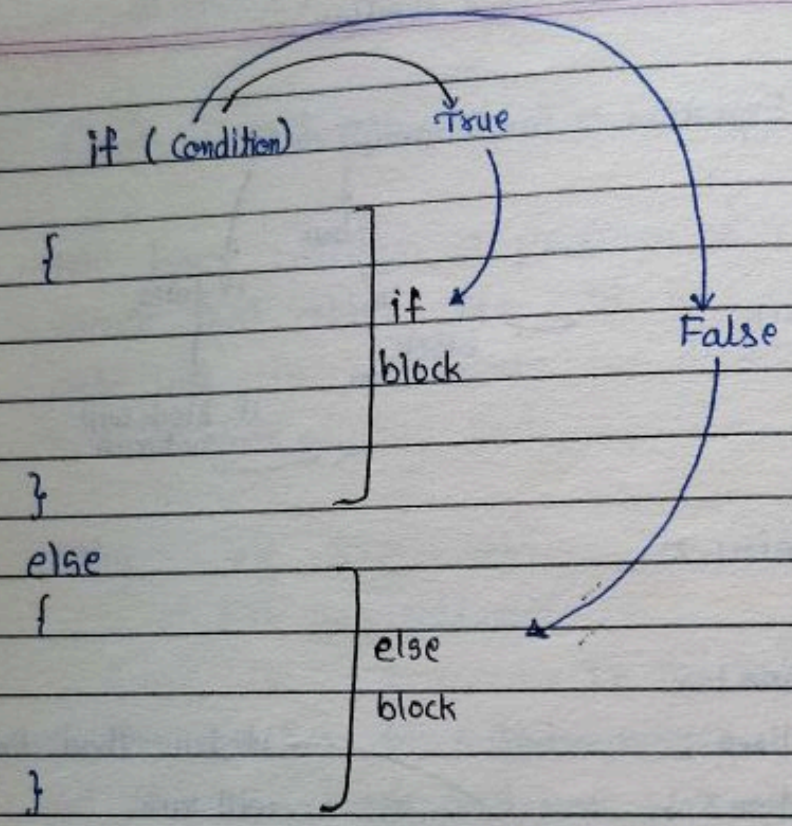
```
main()
{
  line 1 ;
  line 2 ;
  line 3 ;
  if (expration) condition)
  { line 4 ;
    line 5 ;
    line 6 ;
  }
  line 7 ;
  line 8 ;
  line 9 ;
  line 10 ;
}
```

if block

→ if true than line 4, 5, 6 will run

→ false (0) means line 4, 5, 6 will not run

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:26




else have no ( ), meanse it have no condition, it depends on if's (expration/condition).

if का block खत्म होते ही else लिखना है। हम बीच मे कोई और लाइन नही लिख सकते, it will show error.

else कभी अकेला use नही हो सकता क्योंकि उसका meaning ही नही है।

else, if दोनो ही keyword है।

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:26

: colon

Conditional operator ⇒ (? :) ⇒ (Ternary operator)

exp1 ? exp2 : exp3;

True

False

```
main()
{   int n;
    printf("Enter the number");
    scanf("%d", &n);
  if (n>0)
    printf("Positive");
  else
     printf("Non positive");
    getch();
  }
```

```
main()
{   int n;
    printf("Enter the number");
    scanf("%d", &n);
    n>0 ? printf("Positive") : printf("Nonpositive");
    getch();
  }
```

exp1 ? exp2 : exp3; it is one expression

if exp1 is true then exp2 will work

if exp1 is false then exp3 will work

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:26

23 Nov - 30 Jan - 30 Sep.

C language / Batch Code : SB  Date: 23/11/21 Page No.:

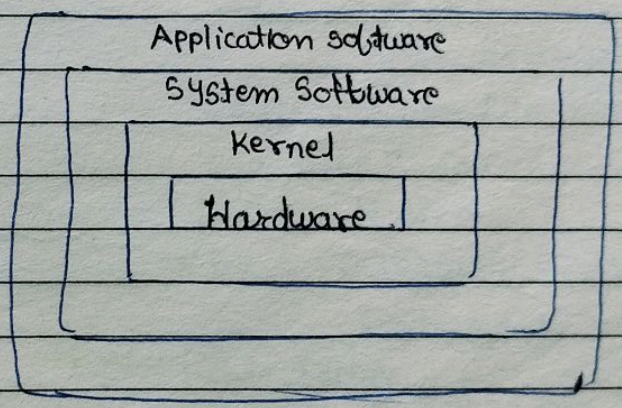
https://premium.mysirg.com ((create account here))

here - recording of Live classes
(shub01690@gmail.com) Assignments  nidhu@01690
- Certificate
- Discussion Forum
- support
- Account

https://coder.mysirg.com | user Name NIDHIYADAV
Pass. → 919799

---

Computer




Hardware ⇒ (tangible) RAM, hard disk, cache memory
Processor, Touchpad, USB, mouse etc

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:18

Date: / / Page No.:

| | | | |
|---|---|---|---|
| ✓ | | 1 | Orientation lecture |
| ✓ | | 2 | Fundamentals for C language |
| ✓ | | 3 | What is programing, Role of C, IDE |
| ✓ | | 4 | History, Begin learning C, Constants |
| ✓ | | 5 | Variables and keywords |
| ✓ | | 6 | Data type declaration instruction |
| ✓ | | 7 | float vs double, ASCII, block, Rules |
| ✓ | | 8 | Output instruction, printf, getch, build and run |
| ✓ | | 9 | Printf, scanf, escape sequence, formate specifier |
| ✓ | | 10 | Unary operators |
| ✓ | | 11 | Arithmetic Operators |
| ✓ | | 12 | Bitwise, relational, logical, assignment operators, if...else |
| ✓ | • | 13 | Conditional Operator |
| ✓ | | 14 | nested if else, else if ladder |
| ✓ | • | 15 | while |
| ✓ | • | 16 | break and continue |
| ✓ | • | 17 | Loop |
| ✓ | • | 18 | do while, for loop |
| ✓ | • | 19 | Doubt handling |
| ✓ | • | 20 | more on loops |
| ✓ | | 21 | star pattern logic |
| ✓ | | 22 | Switch case, goto control |
| ✓ | • | 23 | Functions introduction |
| ✓ | • | 24 | Ways to define a function |
| ✓ | | 25 | Recursion |
| ✓ | | 26 | Array introduction |
| ✓ | | 27 | arrays and functions |
| ✓ | | 28 | Arrays and functions |
| ✓ | | 29 | 2d Arrays |

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:52

Date: / / Page No.:

30 Strings
31 Strings and function
32 multiply strings, pointer, pointer's arithmetic
33 Applications of pointers, call by reference
34 structure
35 Vidio lesson 50
36 Union, enum, typedef, storage classes
37 Preprocessor, Last class

Assignment - 1 | Printf, scanf
Assignment - 2 | Operators
Assignment - 3 | Decision control
Assignment - 4 | while loop
Assignment - 5 | loop
Assignment - 6 | More on Loops
Assignment - 7 | Star pattern problems
Assignment - 8 | Switch
Assignment - 9 | functions
Assignment - 10 | Recursion
Assignment - 11 | Arrays
Assignment - 12 | Arrays and functions
Assignment - 13 | 2d arrays
Assignment - 14 | Strings
Assignment - 15 | strings and functions
Assignment - 16 | multiple strings
Assignment - 17 | Pointer, structure, DMA

Test for Certificate

Shot on OnePlus
Powered by Triple Camera 2023.11.05 11:52