Let's Start

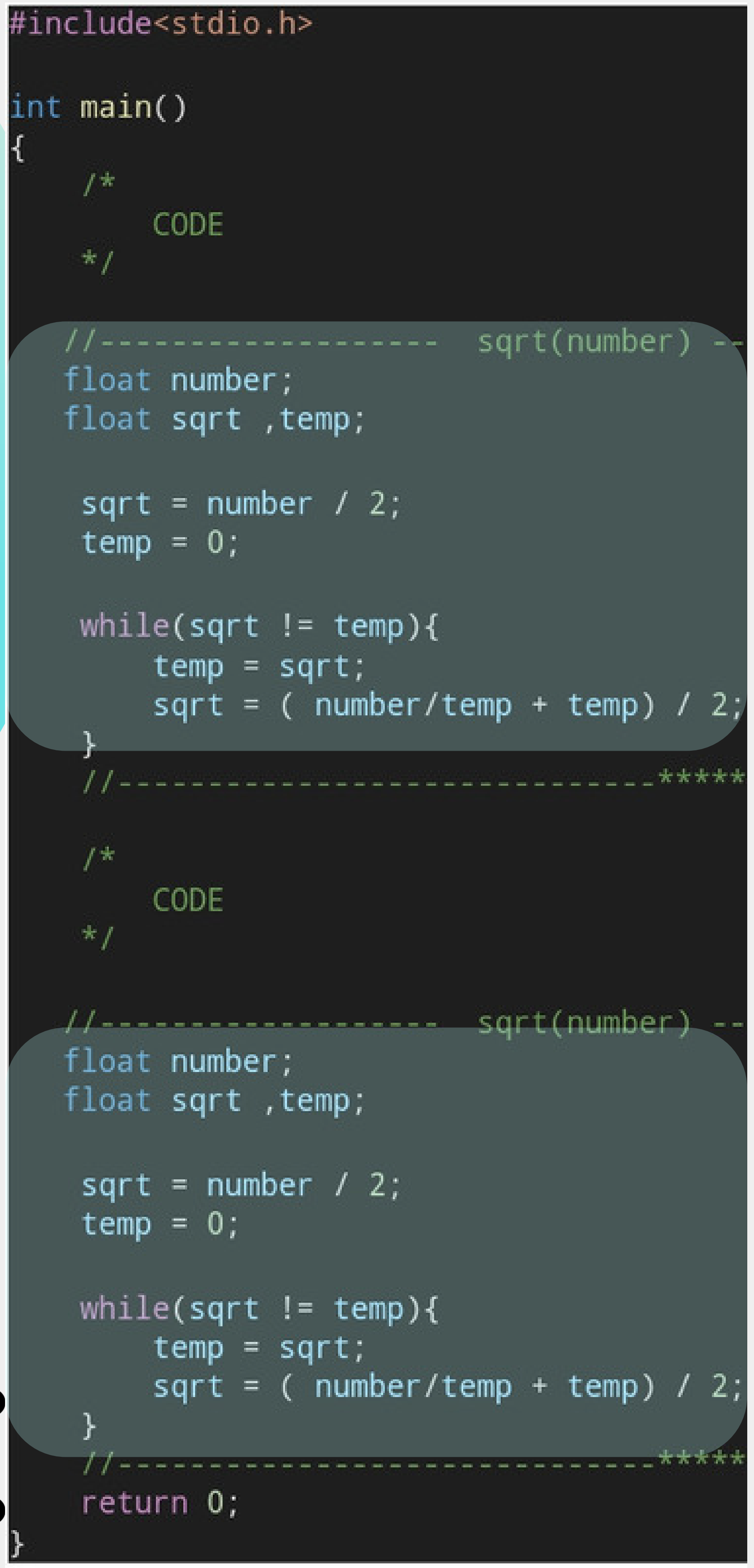
Topic <libraries>

KOSS Task

Karan K. Punjabi

21IM30010

INDEX


```
#include<stdio.h>

int main()
{
    /*
        CODE
    */

   //------------------   sqrt(number) --
   float number;
   float sqrt ,temp;

    sqrt = number / 2;
    temp = 0;

    while(sqrt != temp){
        temp = sqrt;
        sqrt = ( number/temp + temp) / 2;
    }
    //----------------------------*****

    /*
        CODE
    */

   //------------------   sqrt(number) --
   float number;
   float sqrt ,temp;

    sqrt = number / 2;
    temp = 0;

    while(sqrt != temp){
        temp = sqrt;
        sqrt = ( number/temp + temp) / 2;
    }
    //----------------------------*****
    return 0;
}
```

Assume you're creating a programme that requires you to take the square root of a number MULTIPLE TIMES

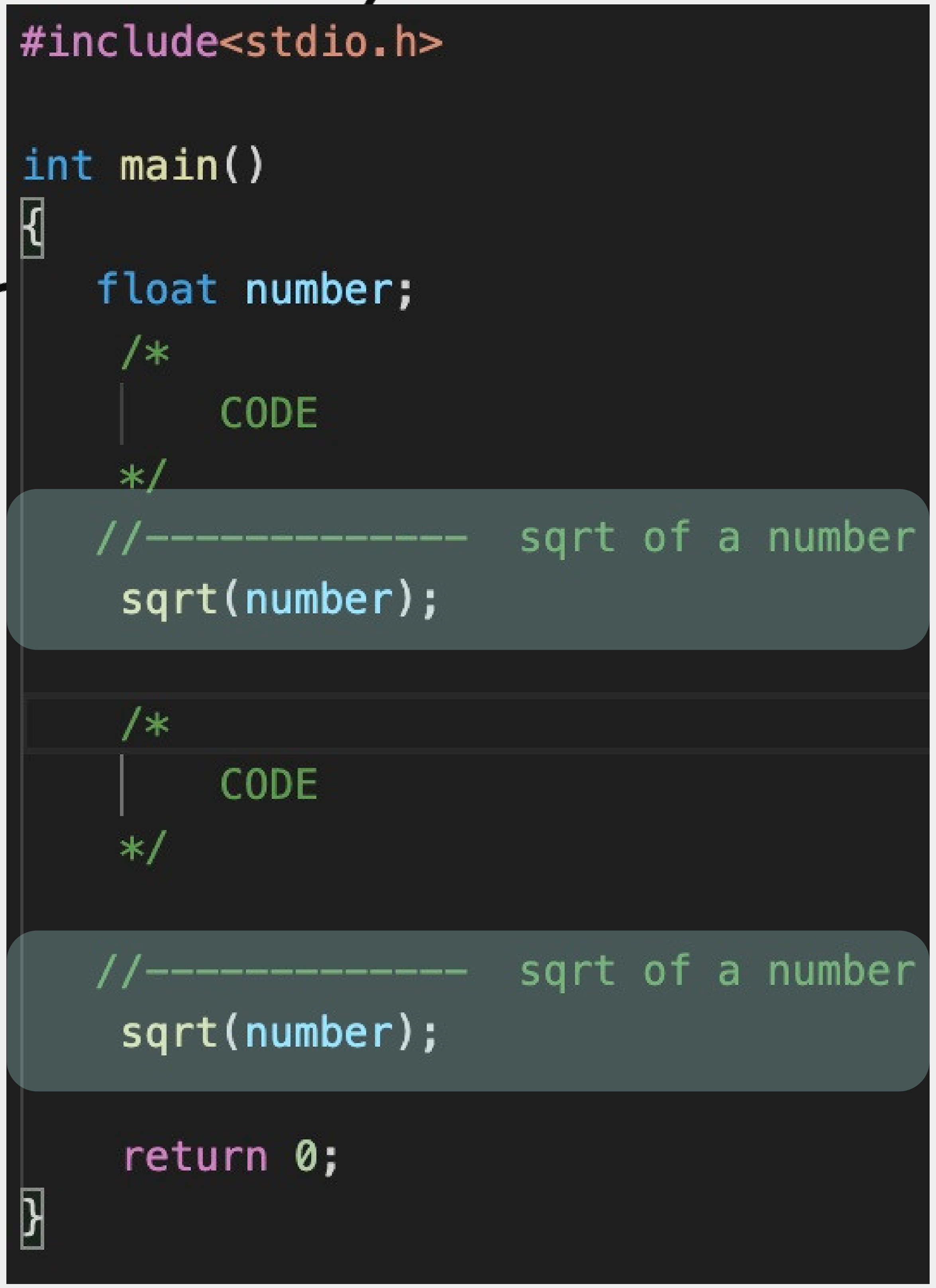
Code to compute Square root of a number

REDUNDANCY of Code

- Long Code
- Ctrl C / V
- No Reusability

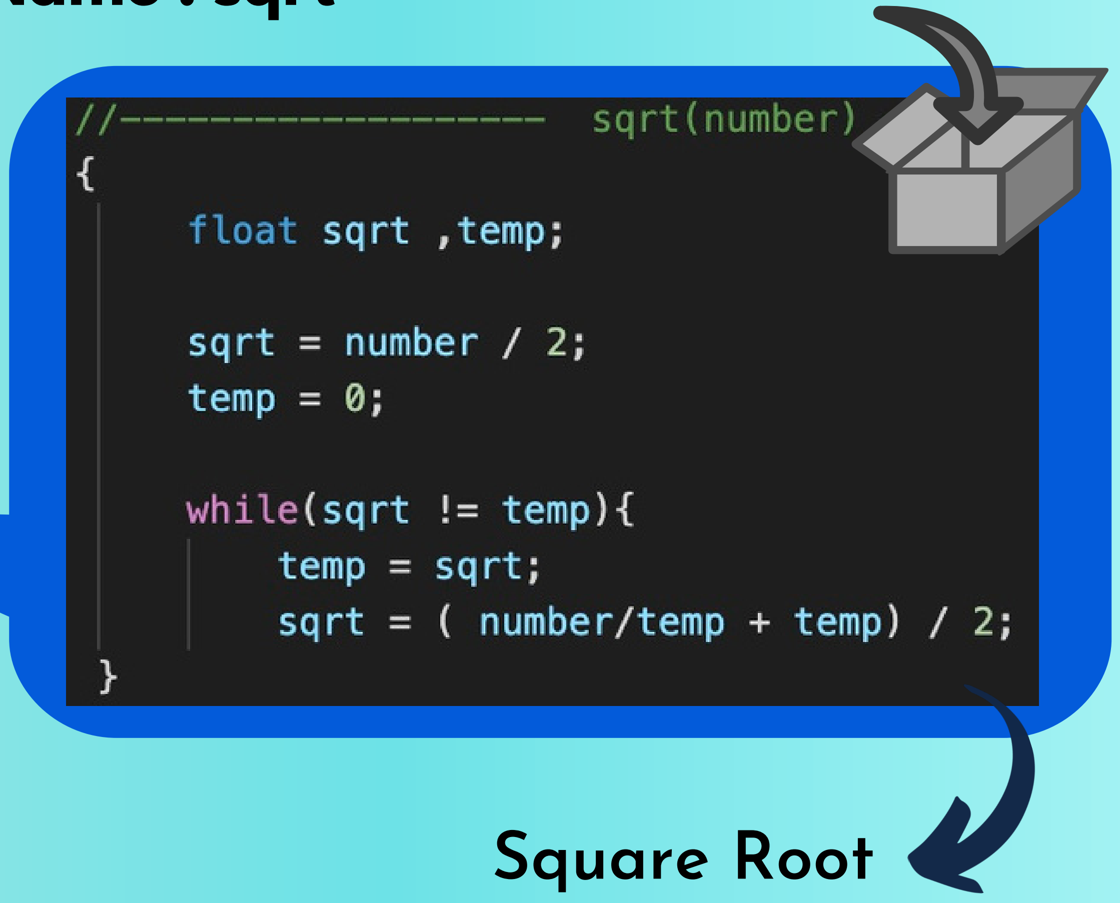
Something Efficient
Can be done ‼

But WHAT & HOW??



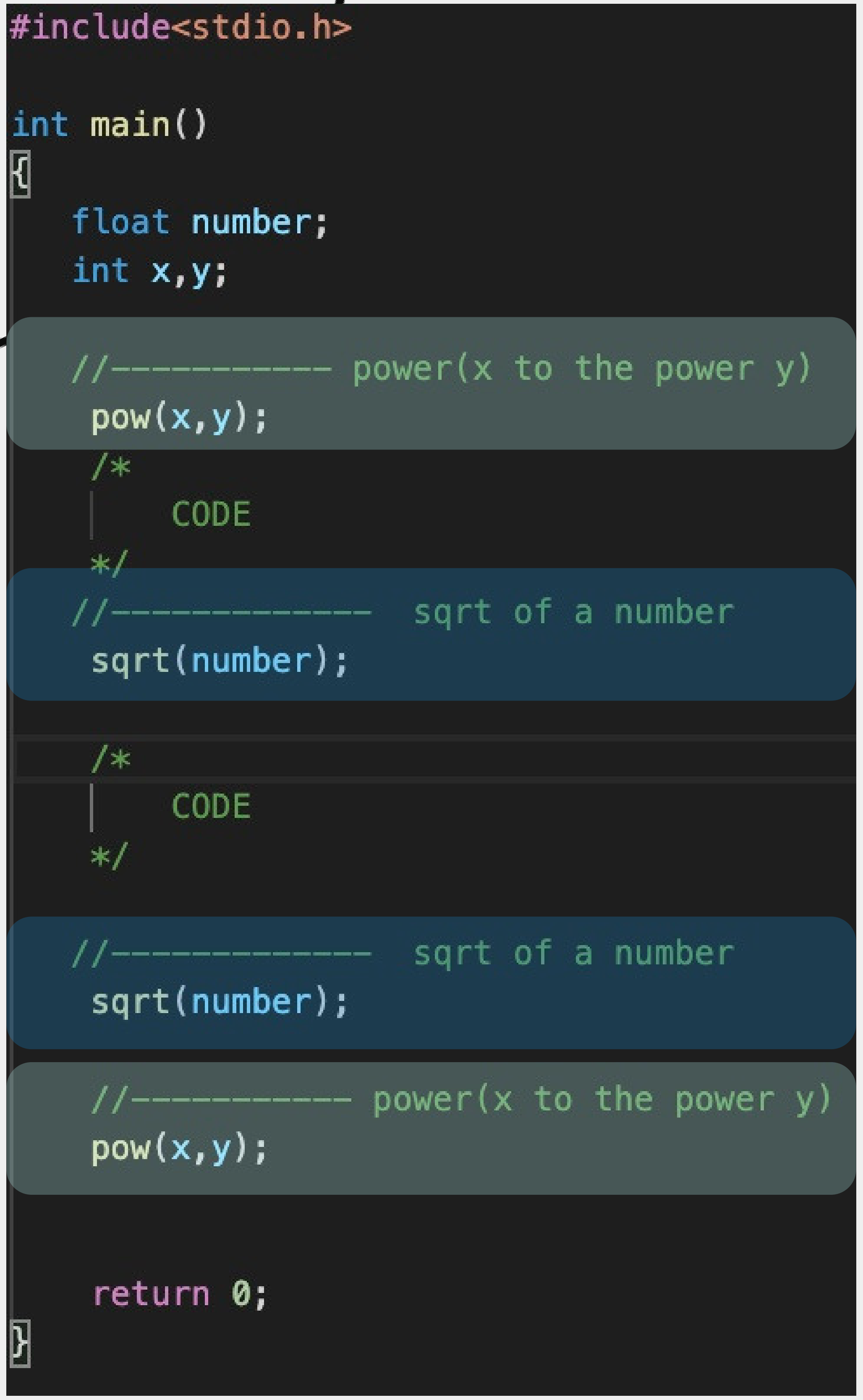
```c
#include<stdio.h>

int main()
{
    float number;
     /*
         CODE
     */
    //-------------   sqrt of a number
     sqrt(number);

     /*
         CODE
     */

    //-------------   sqrt of a number
     sqrt(number);

     return 0;
}
```

Say A Book
Name : sqrt



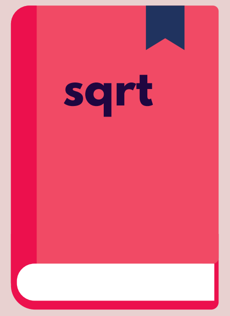
```c
//-------------------   sqrt(number)
{
    float sqrt ,temp;

    sqrt = number / 2;
    temp = 0;

    while(sqrt != temp){
        temp = sqrt;
        sqrt = ( number/temp + temp) / 2;
}
```

// Function () {}

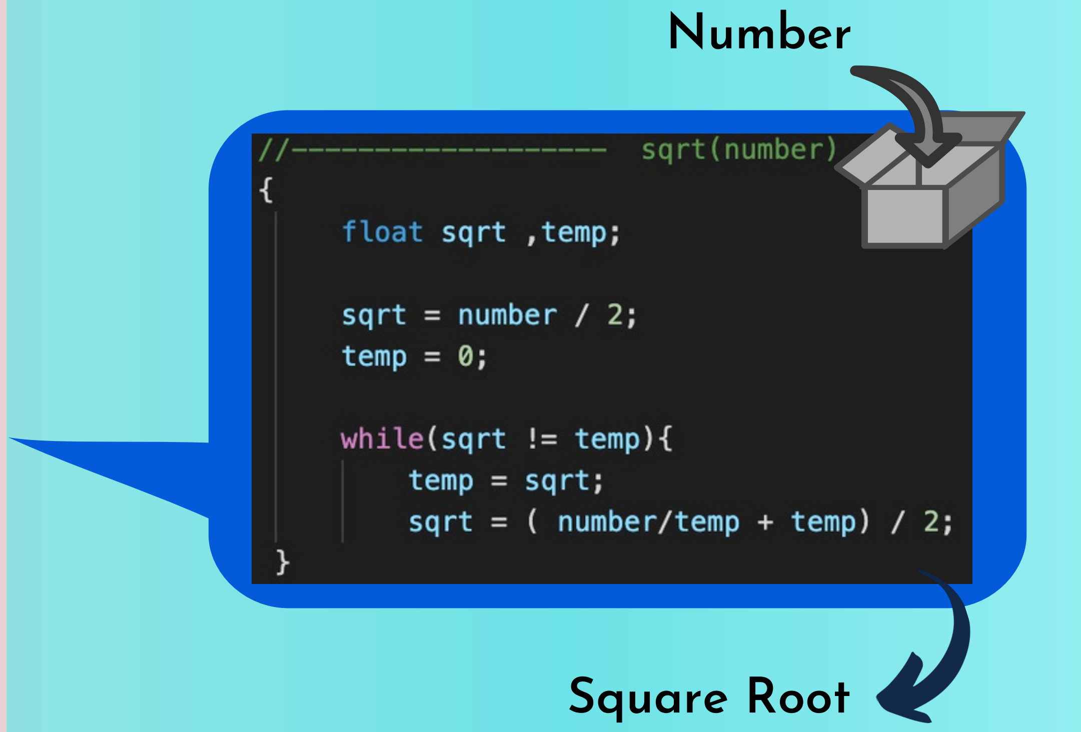


```c
#include<stdio.h>

int main()
{
   float number;
   int x,y;
   //----------- power(x to the power y)
    pow(x,y);
    /*
        CODE
    */
   //-------------  sqrt of a number
    sqrt(number);

    /*
        CODE
    */
   //-------------  sqrt of a number
    sqrt(number);
    //----------- power(x to the power y)
    pow(x,y);


    return 0;
}
```

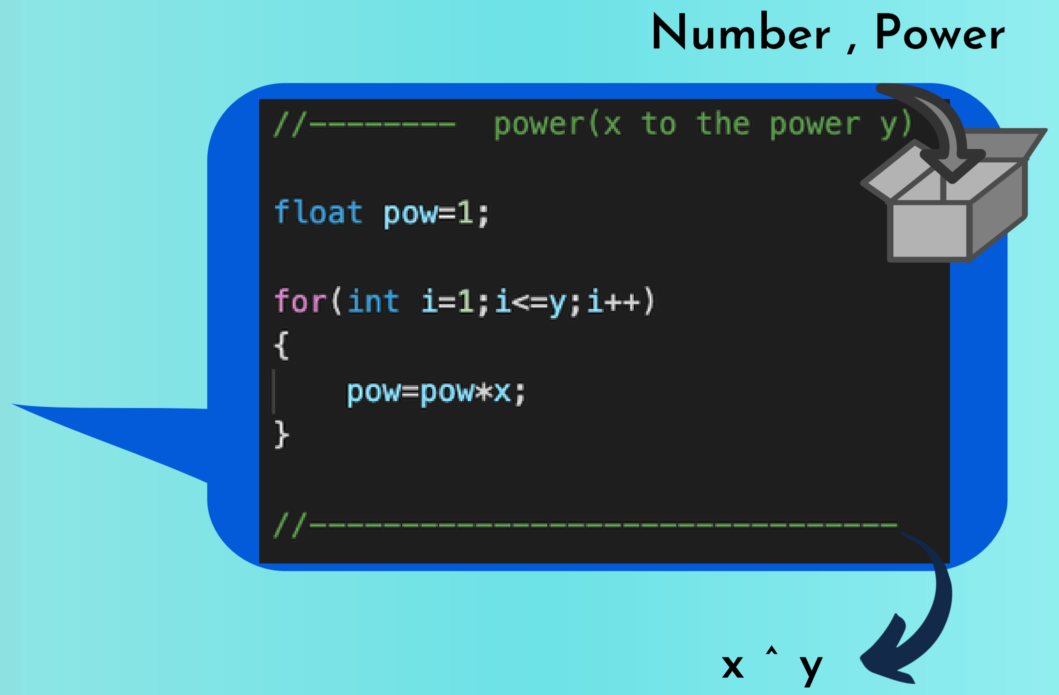
// <libraries>



<math>

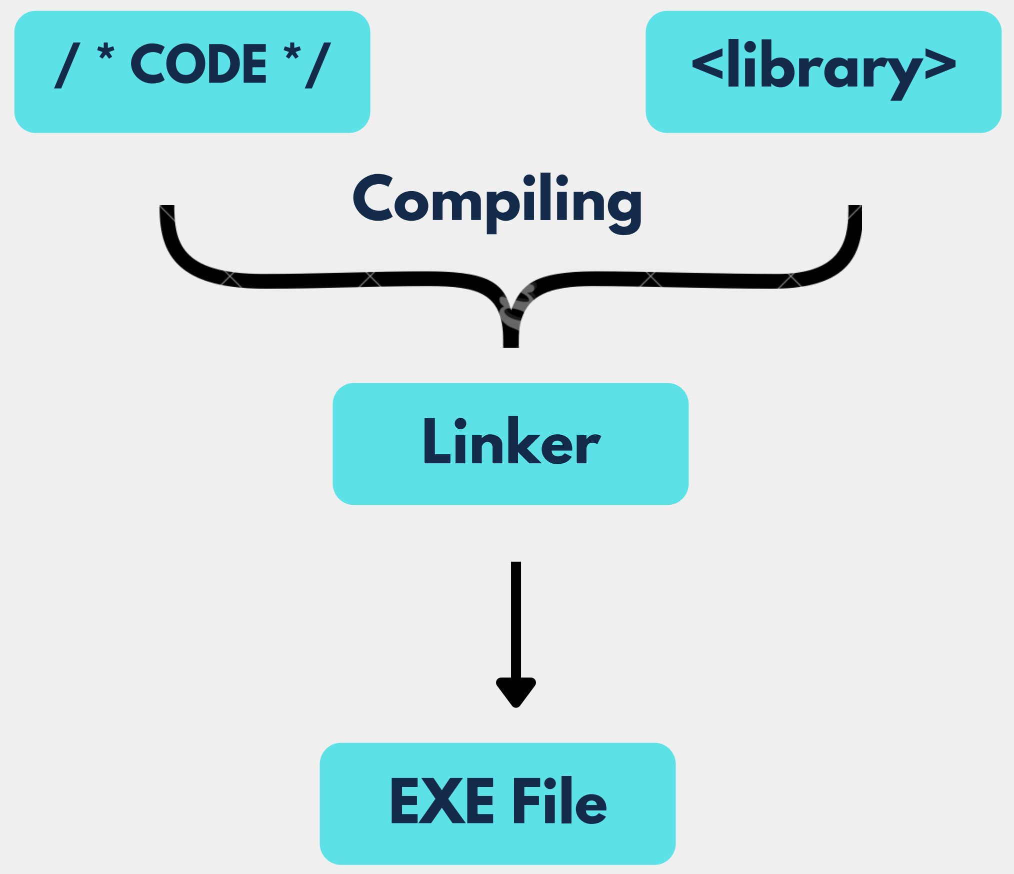


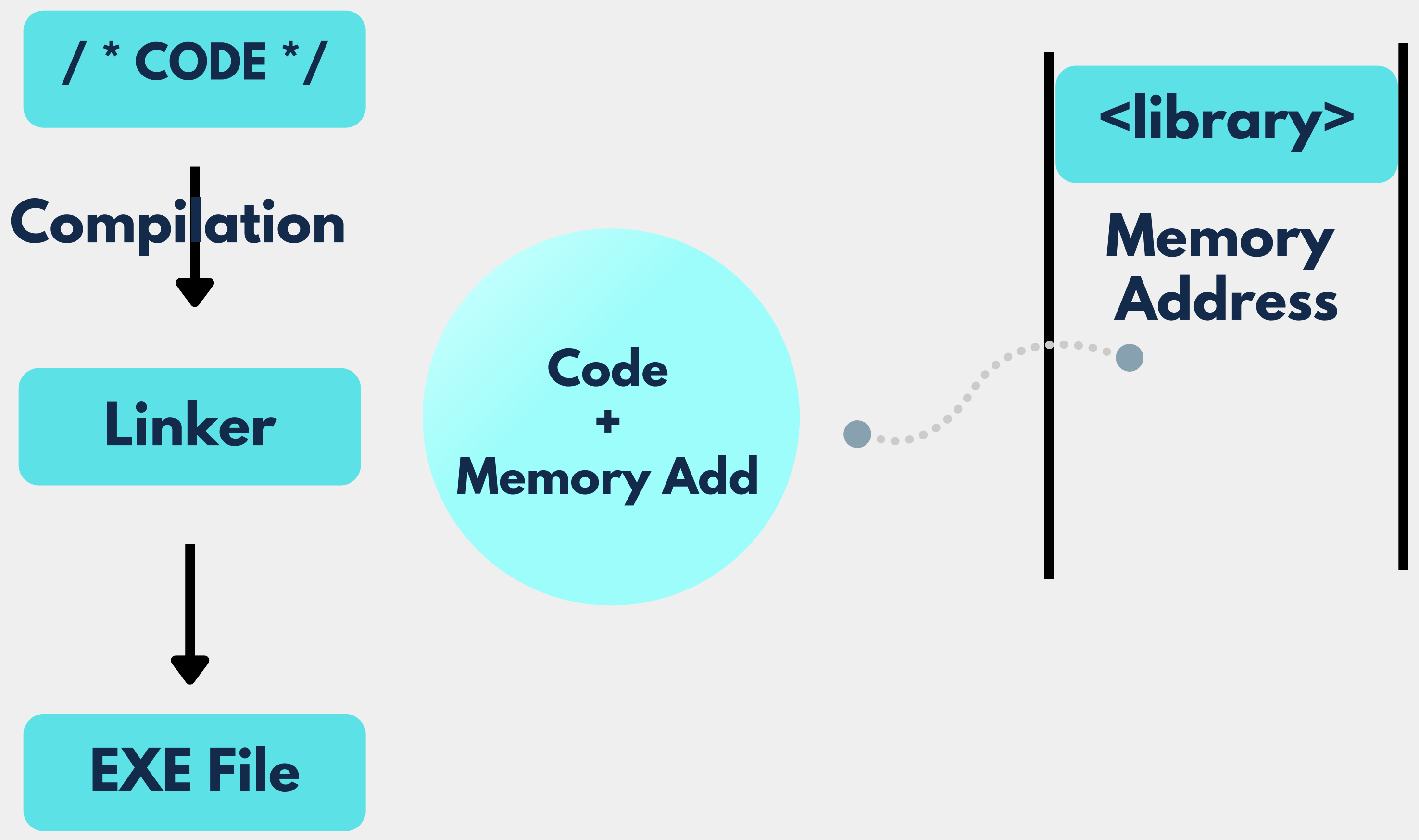
```c
//------------------  sqrt(number)
{
    float sqrt ,temp;

    sqrt = number / 2;
    temp = 0;

    while(sqrt != temp){
        temp = sqrt;
        sqrt = ( number/temp + temp) / 2;
 }
```



```c
//--------  power(x to the power y)

float pow=1;

for(int i=1;i<=y;i++)
{
    pow=pow*x;
}

//----------------------------------
```

In what ways can you call/Link a <library> !?





Static Linking

Library Prelinked into EXE (executable) file

Static lib

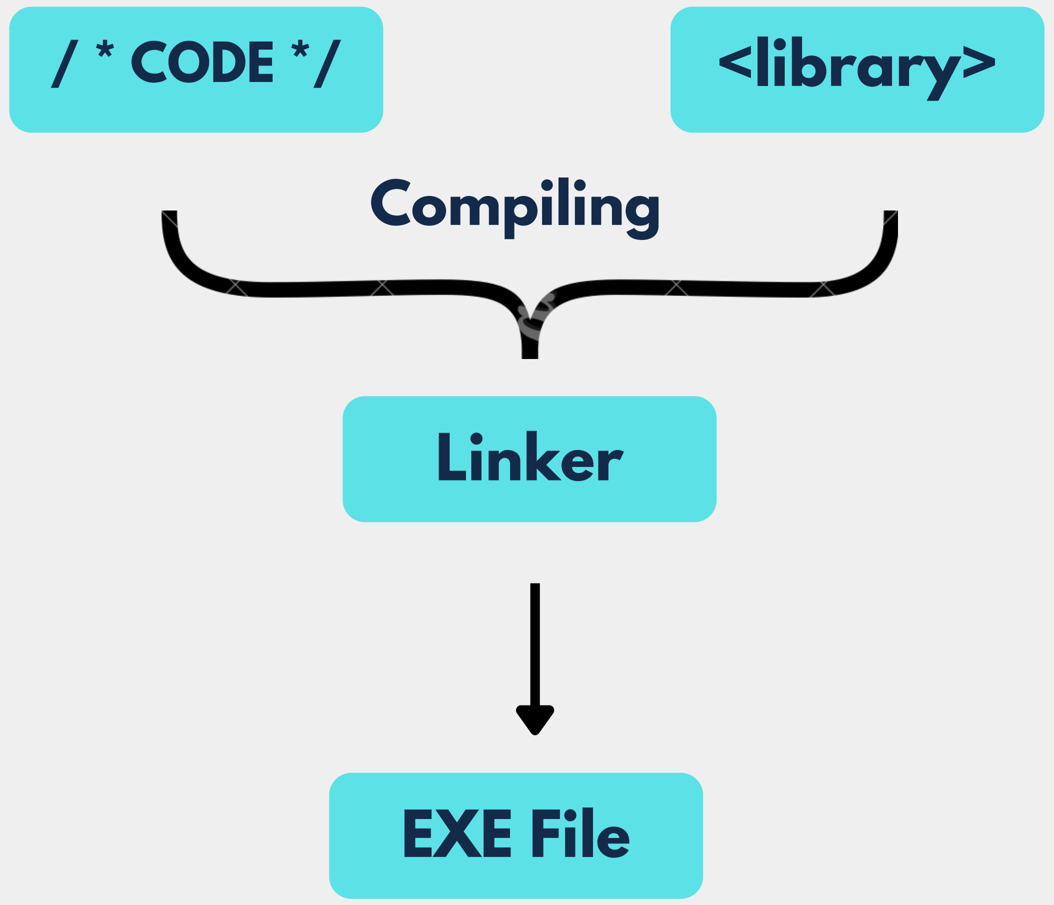
Dynamic Linking

Library links during runtime via memory address

Dynamic lib

Static Library

Library Prelinked into EXE (executable) file



Everything in ONE File implies Higher Memory Space

In case of any updation everything needs to be recompiled

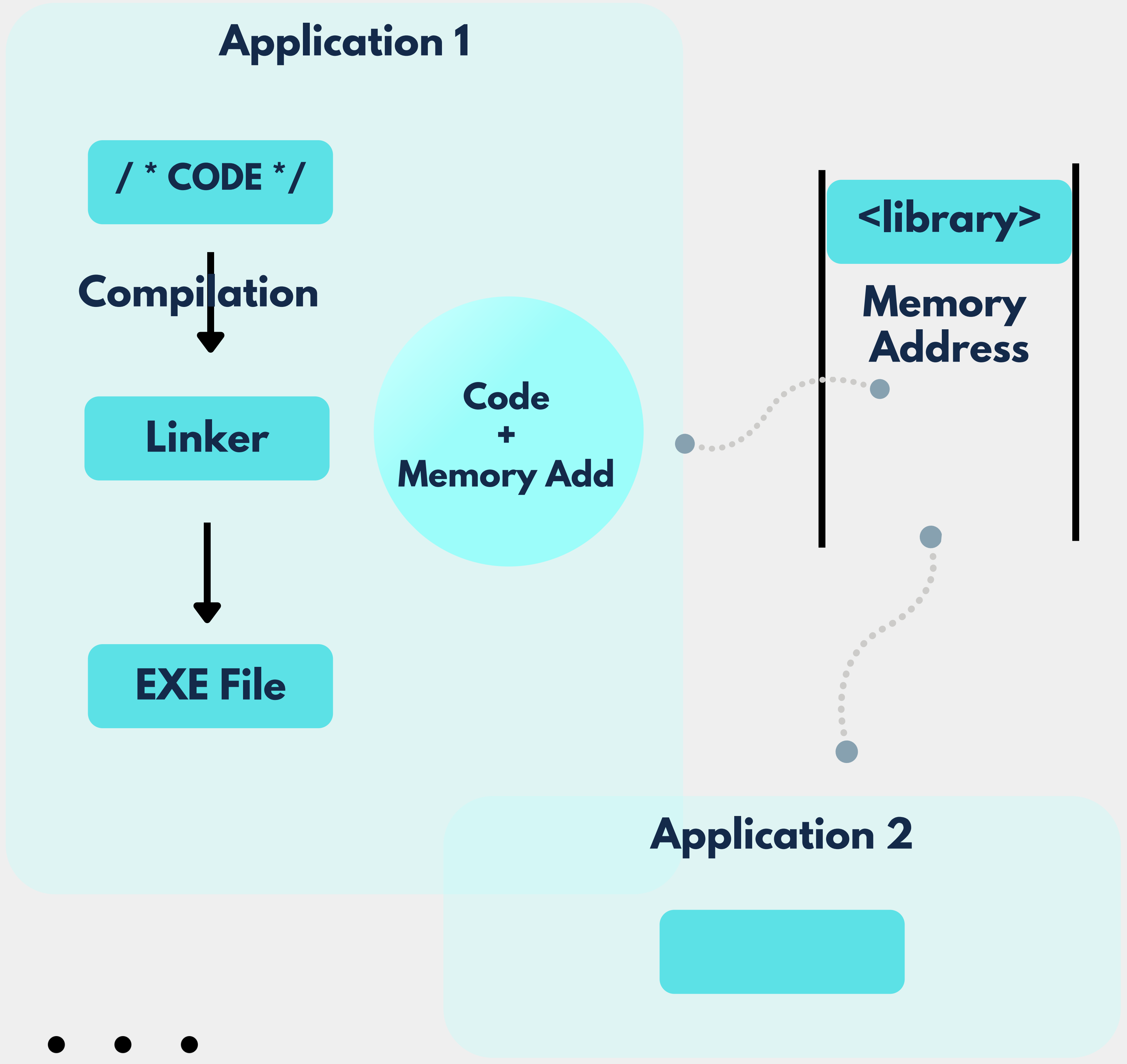
Easy to Distribute and install as everything is in ONE file.

No Dependency on external files

No Compatibility Issues

Dynamic Library

Library links during runtime via memory address



Dependent on external files

Compatibility Issues if the library is removed

Uses less memory space

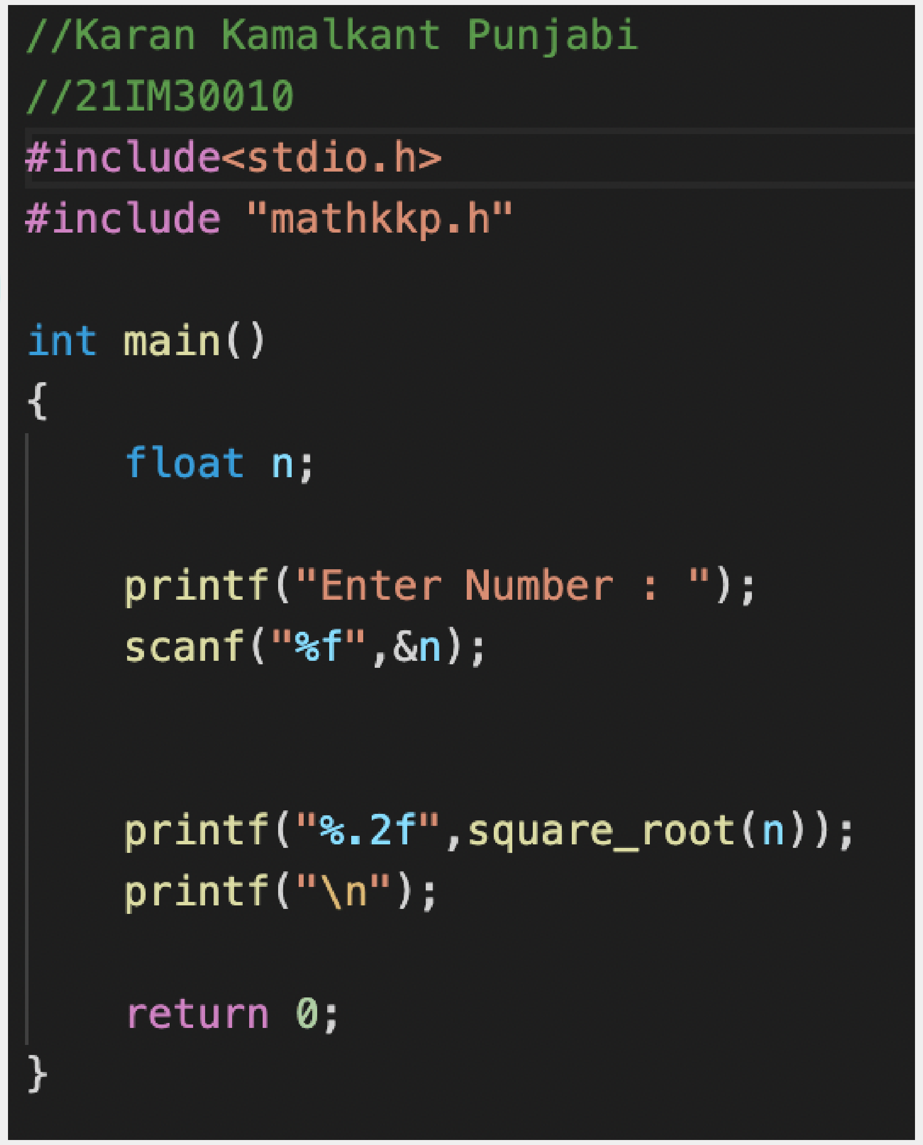
One Library could be used in multiple applications

Easy Updation as exe file is the same

Faster compilation process

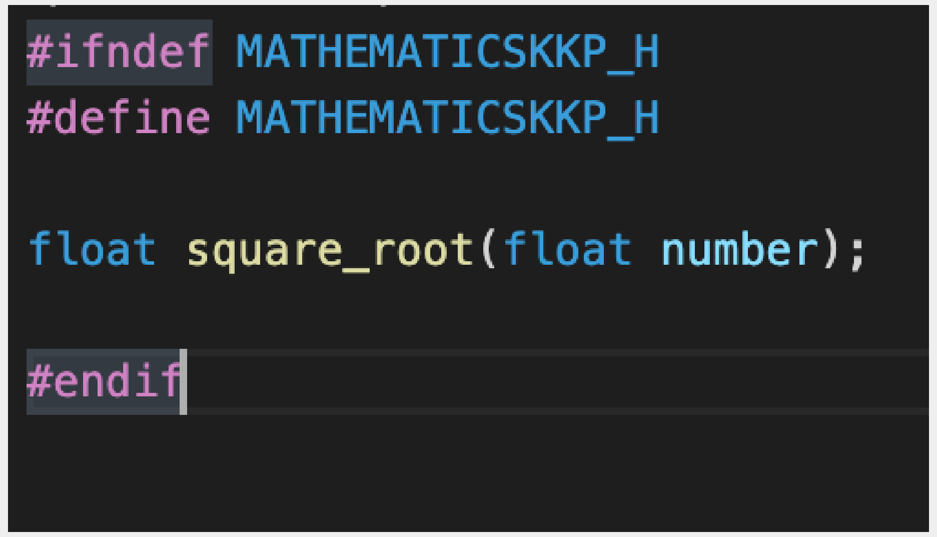
Linking Static Libraries

mainfile.c



```c
//Karan Kamalkant Punjabi
//21IM30010
#include<stdio.h>
#include "mathkkp.h"

int main()
{
    float n;

    printf("Enter Number : ");
    scanf("%f",&n);


    printf("%.2f",square_root(n));
    printf("\n");

    return 0;
}
```

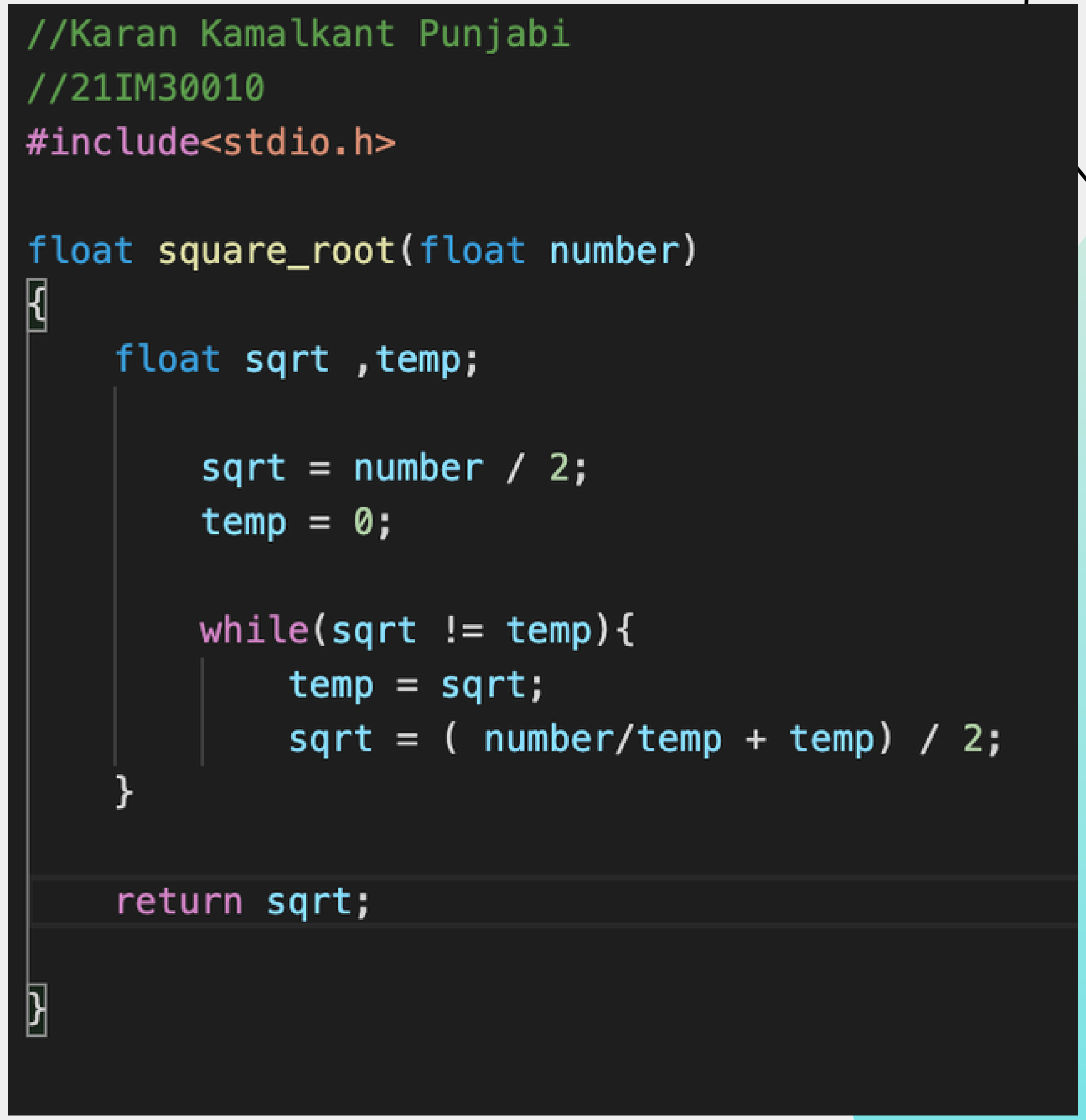
mathkkp.h



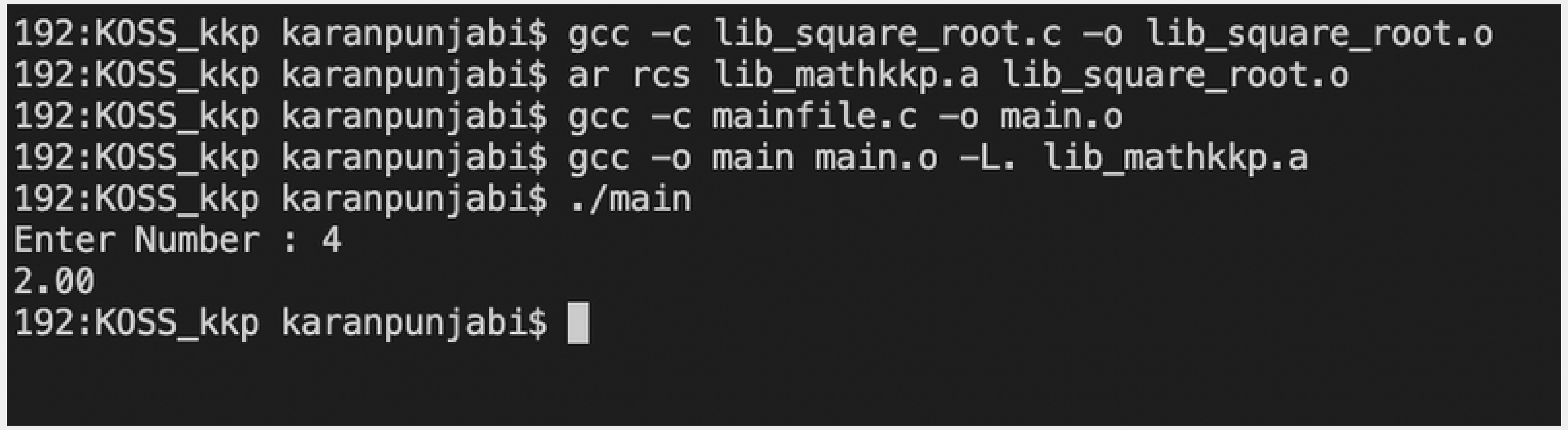
```c
#ifndef MATHEMATICSKKP_H
#define MATHEMATICSKKP_H

float square_root(float number);

#endif
```

Header File storing function declarations

lib_square_root.c



```c
//Karan Kamalkant Punjabi
//21IM30010
#include<stdio.h>

float square_root(float number)
{
    float sqrt ,temp;

        sqrt = number / 2;
        temp = 0;

        while(sqrt != temp){
            temp = sqrt;
            sqrt = ( number/temp + temp) / 2;
    }

    return sqrt;

}
```



```
192:KOSS_kkp karanpunjabi$ gcc -c lib_square_root.c -o lib_square_root.o
192:KOSS_kkp karanpunjabi$ ar rcs lib_mathkkp.a lib_square_root.o
192:KOSS_kkp karanpunjabi$ gcc -c mainfile.c -o main.o
192:KOSS_kkp karanpunjabi$ gcc -o main main.o -L. lib_mathkkp.a
192:KOSS_kkp karanpunjabi$ ./main
Enter Number : 4
2.00
192:KOSS_kkp karanpunjabi$ █
```

Created an object file for lib_square_root.c

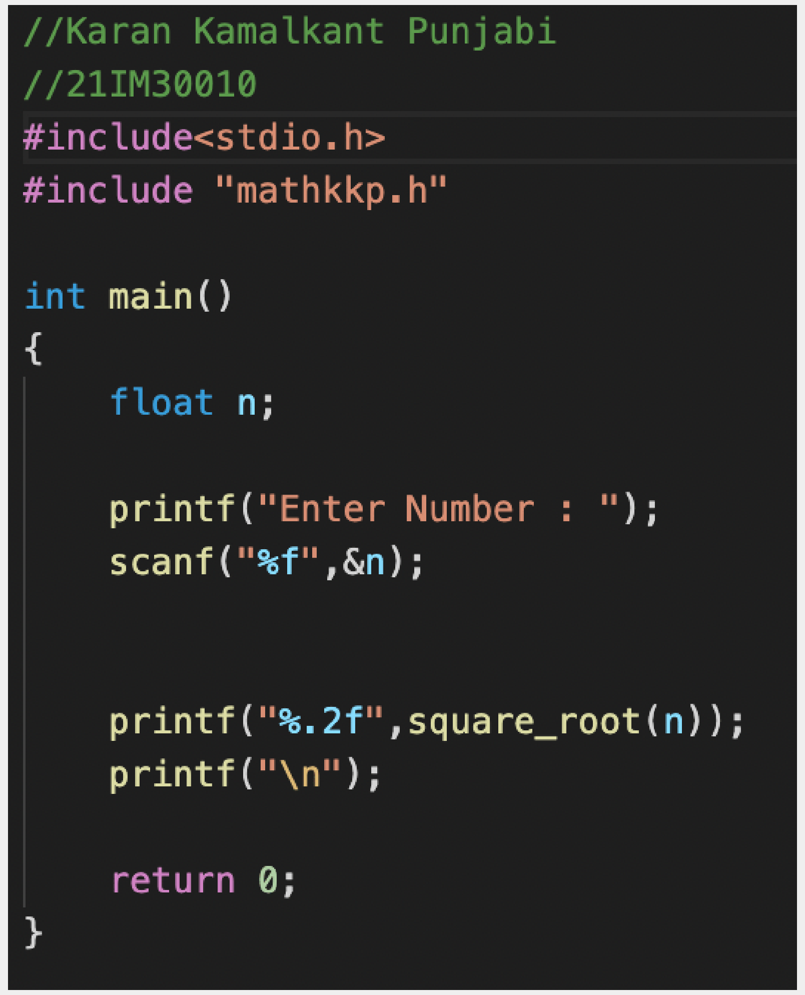
archive command to create static lib by grouping all the .o files into .a static lib file

creating object file for mainfile as main.o

Linking main.o with static lib mathkkp.a and outputing final file as main

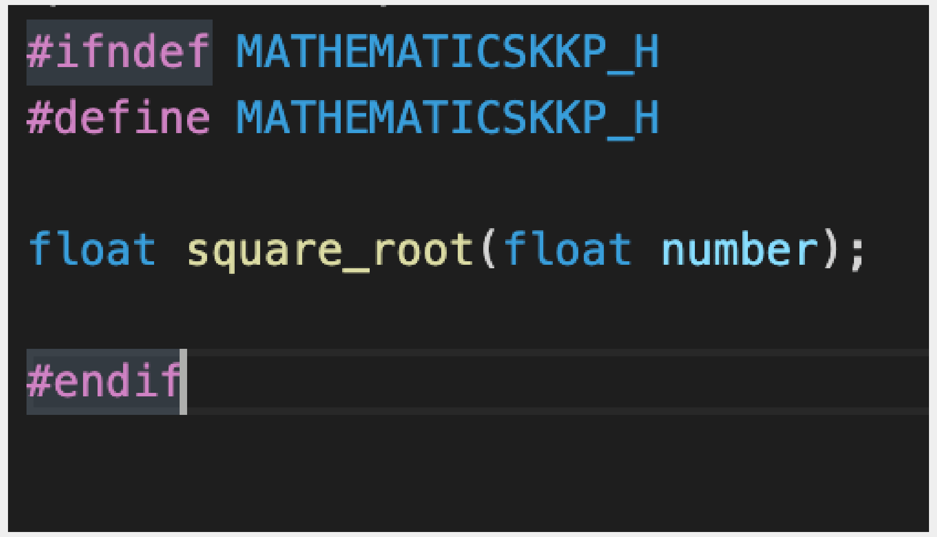
Linking Dynamic/Shared Libraries

main.c



```c
//Karan Kamalkant Punjabi
//21IM30010
#include<stdio.h>
#include "mathkkp.h"

int main()
{
    float n;

    printf("Enter Number : ");
    scanf("%f",&n);


    printf("%.2f",square_root(n));
    printf("\n");

    return 0;
}
```

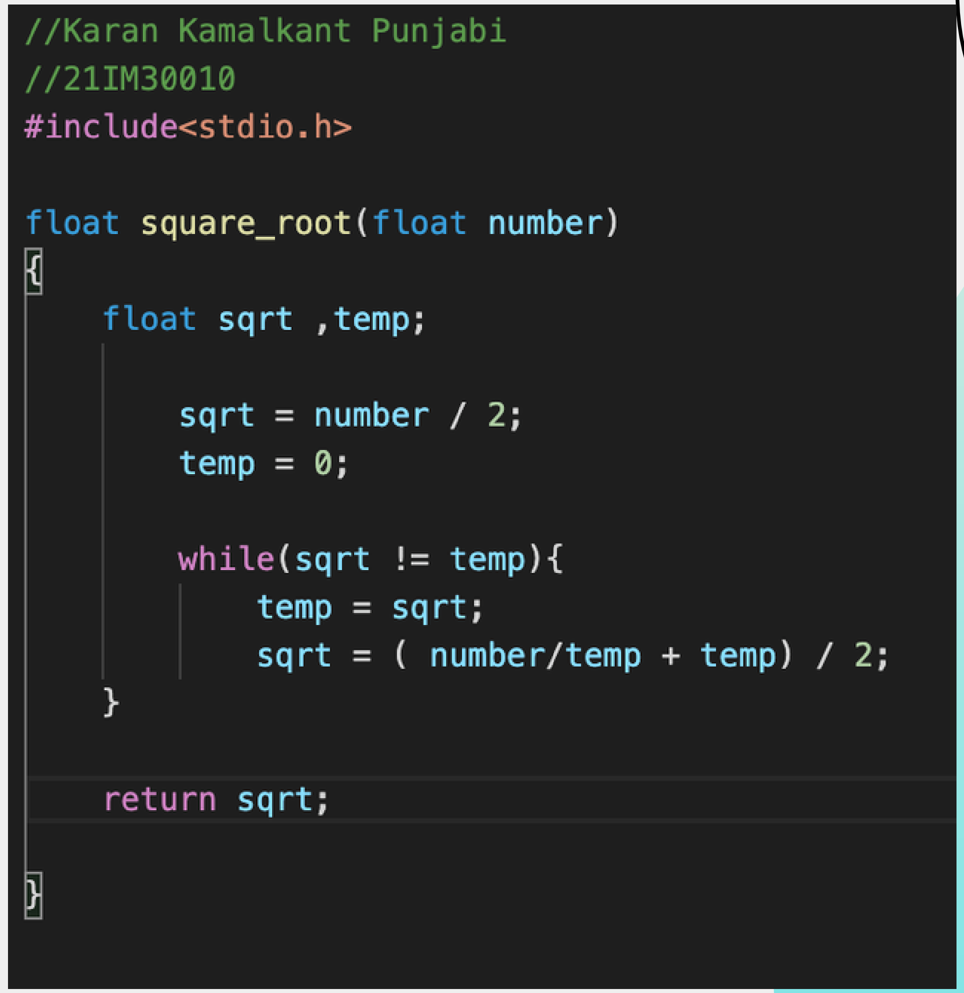
mathkkp.h



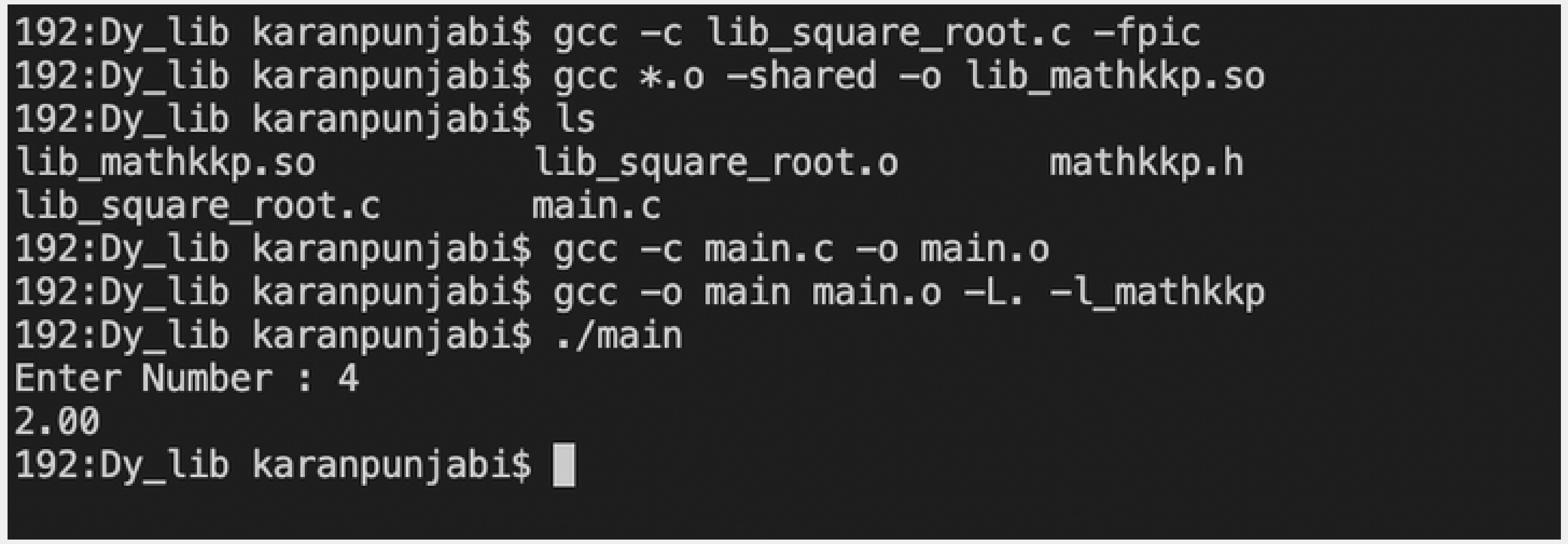
```c
#ifndef MATHEMATICSKKP_H
#define MATHEMATICSKKP_H

float square_root(float number);

#endif
```

Header File storing function declarations

lib_square_root.c



```c
//Karan Kamalkant Punjabi
//21IM30010
#include<stdio.h>

float square_root(float number)
{
    float sqrt ,temp;

        sqrt = number / 2;
        temp = 0;

        while(sqrt != temp){
            temp = sqrt;
            sqrt = ( number/temp + temp) / 2;
    }

    return sqrt;

}
```



```
192:Dy_lib karanpunjabi$ gcc -c lib_square_root.c -fpic
192:Dy_lib karanpunjabi$ gcc *.o -shared -o lib_mathkkp.so
192:Dy_lib karanpunjabi$ ls
lib_mathkkp.so          lib_square_root.o       mathkkp.h
lib_square_root.c       main.c
192:Dy_lib karanpunjabi$ gcc -c main.c -o main.o
192:Dy_lib karanpunjabi$ gcc -o main main.o -L. -l_mathkkp
192:Dy_lib karanpunjabi$ ./main
Enter Number : 4
2.00
192:Dy_lib karanpunjabi$ █
```

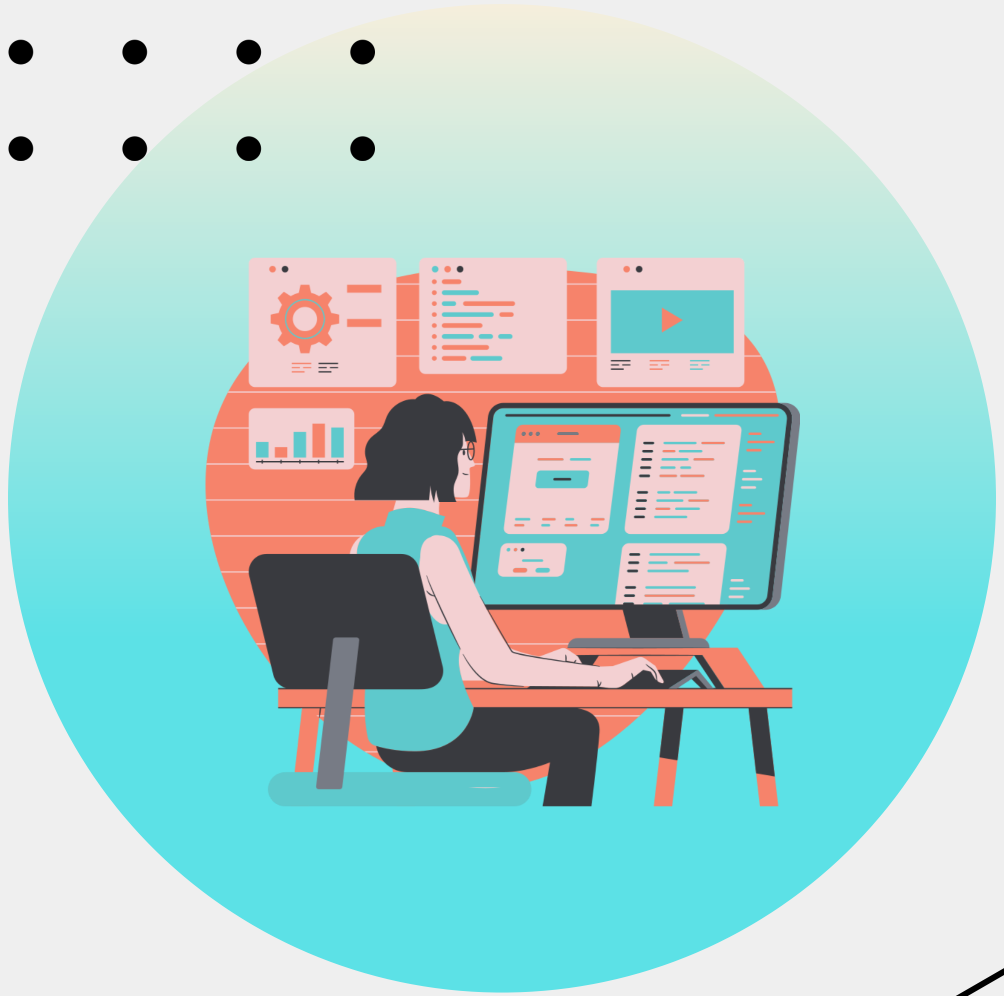
Created an object file for lib_square_root.c with the help of the command fpic (used to create postion independent code)

Creating an so file (extention for shared lib) with the help of share flag

creating object file for mainfile as main.o

Linking main.o with dynamic lib mathkkp.so and outputing final file as main

ld command



- ld command is the linkage command that combines all the object files into one output object file
- -l is the flag used as a shortcut to designate the -lname to libname.so/a
- -L indicates the linking of the library to the final executable file if lib present in the same directory the add a dot after -L and if present in another directory we need to specify its location after -L

Appendix

..
https://medium.com/@StueyGK/static-libraries-vs-dynamic-libraries-af78f0b5f1e4
https://www.linkedin.com/pulse/what-static-library-how-use-them-juan-david-tuta-botero/?trackingId=9ogvYX7jTMyzl%2FL%2FJfWW1Q%3D%3D
https://www.linkedin.com/pulse/differences-between-static-dynamic-libraries-juan-david-tuta-botero/
https://cu7ious.medium.com/how-to-use-dynamic-libraries-in-c-46a0f9b98270
https://www.youtube.com/watch?v=eW5he5uFBNM
https://www.youtube.com/watch?v=bzCuiX4lj0I
https://www.youtube.com/watch?v=3RmIVDgPmGkhttps://www.youtube.com/watch?v=Re5Z607jA0A
https://www.ibm.com/docs/en/aix/7.2?topic=l-ld-command

Presentation By

Karan Kamalkant Punjabi
21IM30010

End

Thank you

Do you have any questions?