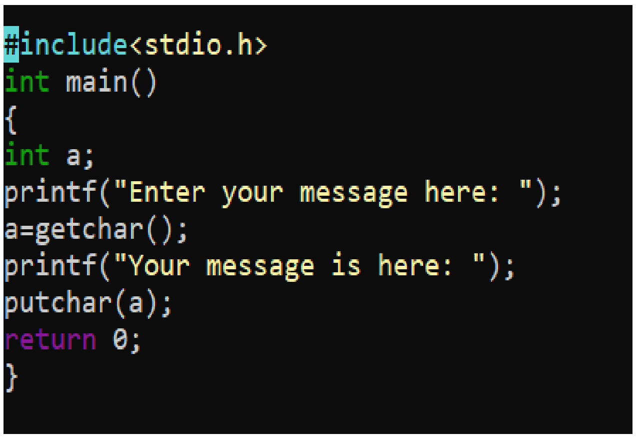
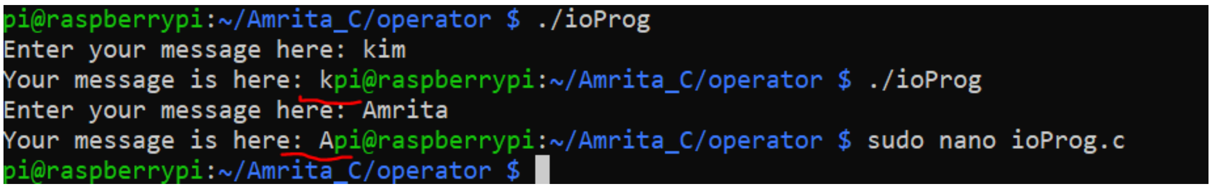
Input-Output:

getchar() &putchar():

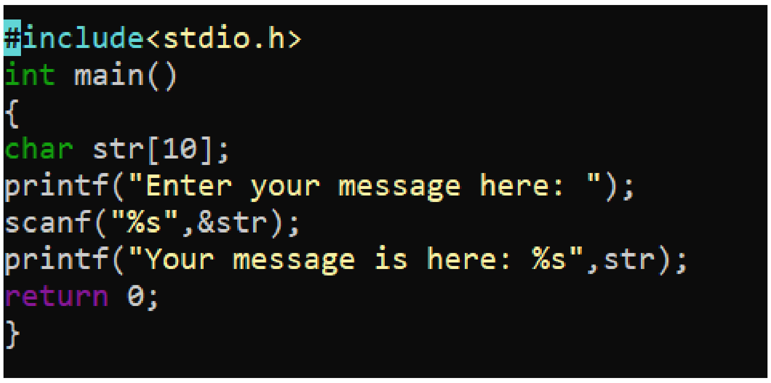


```
#include<stdio.h>
int main()
{
int a;
printf("Enter your message here: ");
a=getchar();
printf("Your message is here: ");
putchar(a);
return 0;
}
```

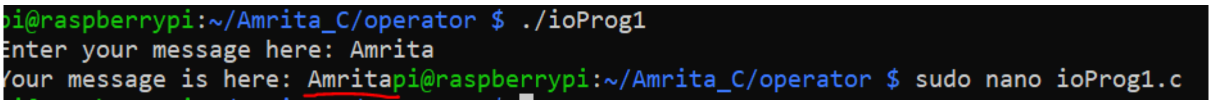


```
pi@raspberrypi:~/Amrita_C/operator $ ./ioProg
Enter your message here: kim
Your message is here: kpi@raspberrypi:~/Amrita_C/operator $ ./ioProg
Enter your message here: Amrita
Your message is here: Api@raspberrypi:~/Amrita_C/operator $ sudo nano ioProg.c
pi@raspberrypi:~/Amrita_C/operator $
```

Use of scanf and printf:

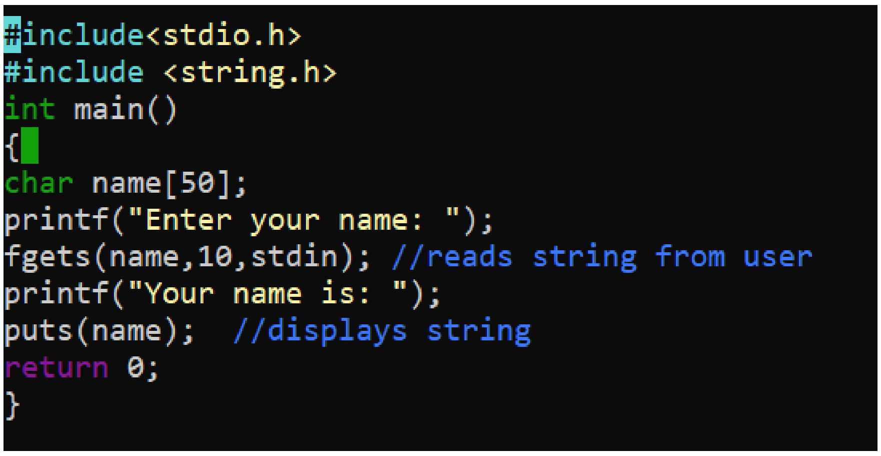


```
#include<stdio.h>
int main()
{
char str[10];
printf("Enter your message here: ");
scanf("%s",&str);
printf("Your message is here: %s",str);
return 0;
}
```

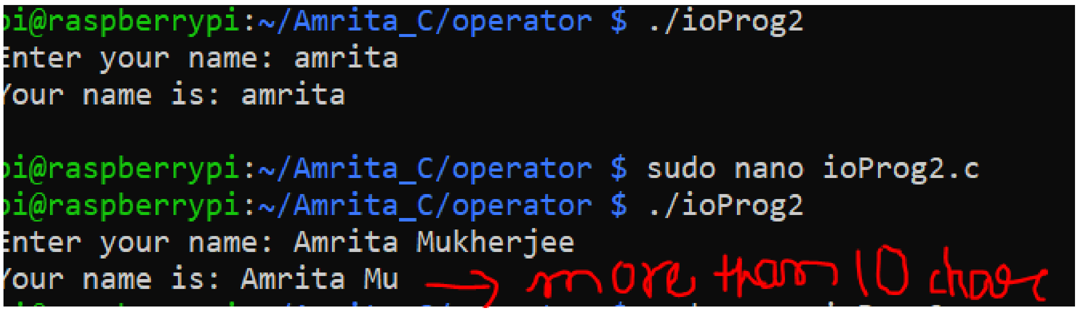


```
i@raspberrypi:~/Amrita_C/operator $ ./ioProg1
nter your message here: Amrita
our message is here: Amritapi@raspberrypi:~/Amrita_C/operator $ sudo nano ioProg1.c
```

Use of fgets() and puts():



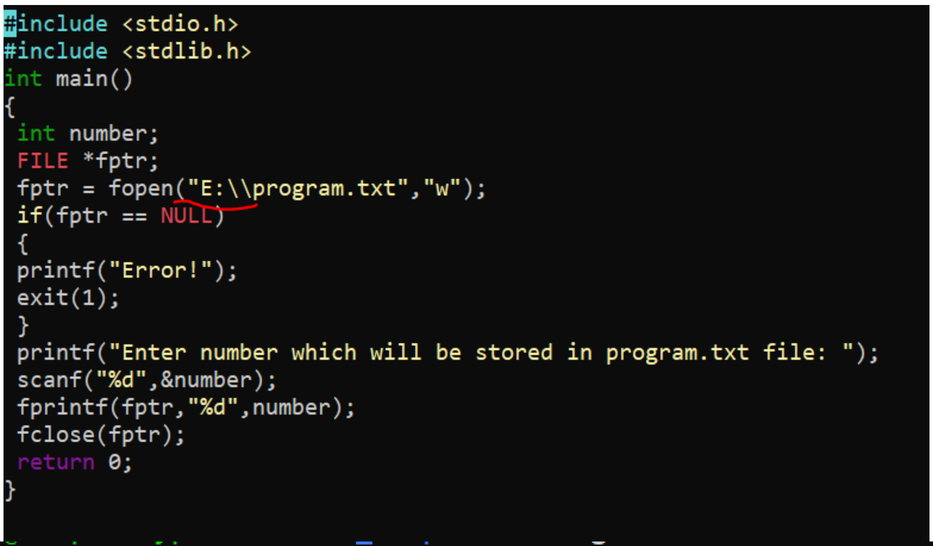
```c
#include<stdio.h>
#include <string.h>
int main()
{
char name[50];
printf("Enter your name: ");
fgets(name,10,stdin); //reads string from user
printf("Your name is: ");
puts(name);  //displays string
return 0;
}
```



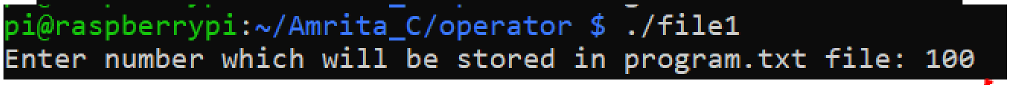
```
i@raspberrypi:~/Amrita_C/operator $ ./ioProg2
nter your name: amrita
our name is: amrita

i@raspberrypi:~/Amrita_C/operator $ sudo nano ioProg2.c
i@raspberrypi:~/Amrita_C/operator $ ./ioProg2
nter your name: Amrita Mukherjee
our name is: Amrita Mu
```

→ more than 10 char

File input output: Created a text file in E drive and store a number in it that is 100.

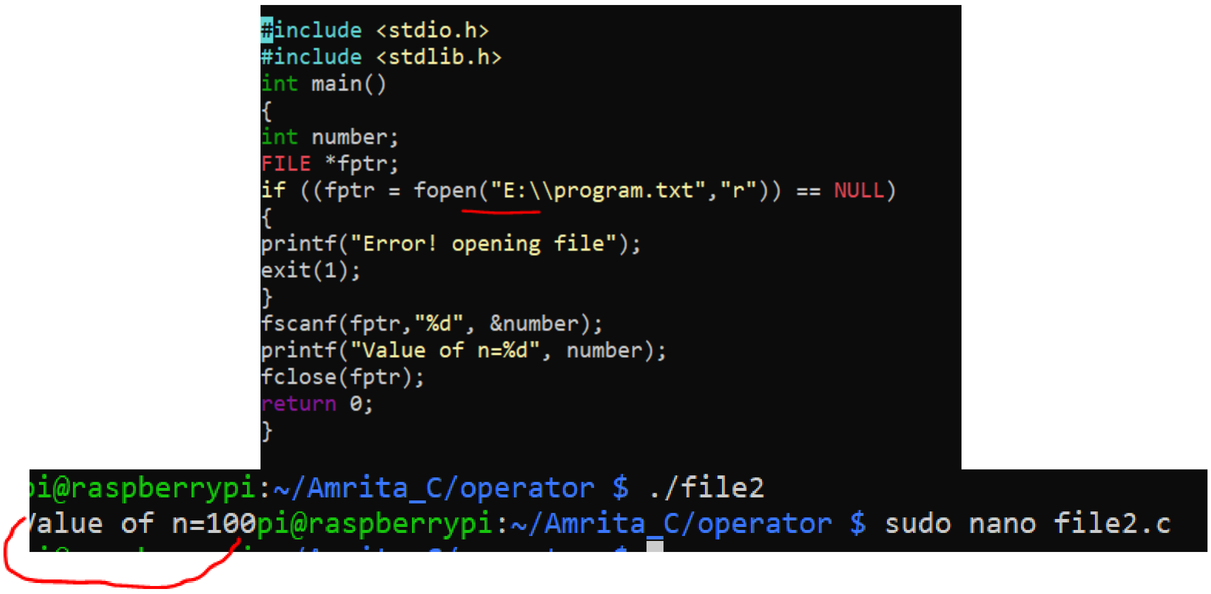


```c
#include <stdio.h>
#include <stdlib.h>
int main()
{
 int number;
 FILE *fptr;
 fptr = fopen("E:\\program.txt","w");
 if(fptr == NULL)
 {
 printf("Error!");
 exit(1);
 }
 printf("Enter number which will be stored in program.txt file: ");
 scanf("%d",&number);
 fprintf(fptr,"%d",number);
 fclose(fptr);
 return 0;
}
```



```
pi@raspberrypi:~/Amrita_C/operator $ ./file1
Enter number which will be stored in program.txt file: 100
```

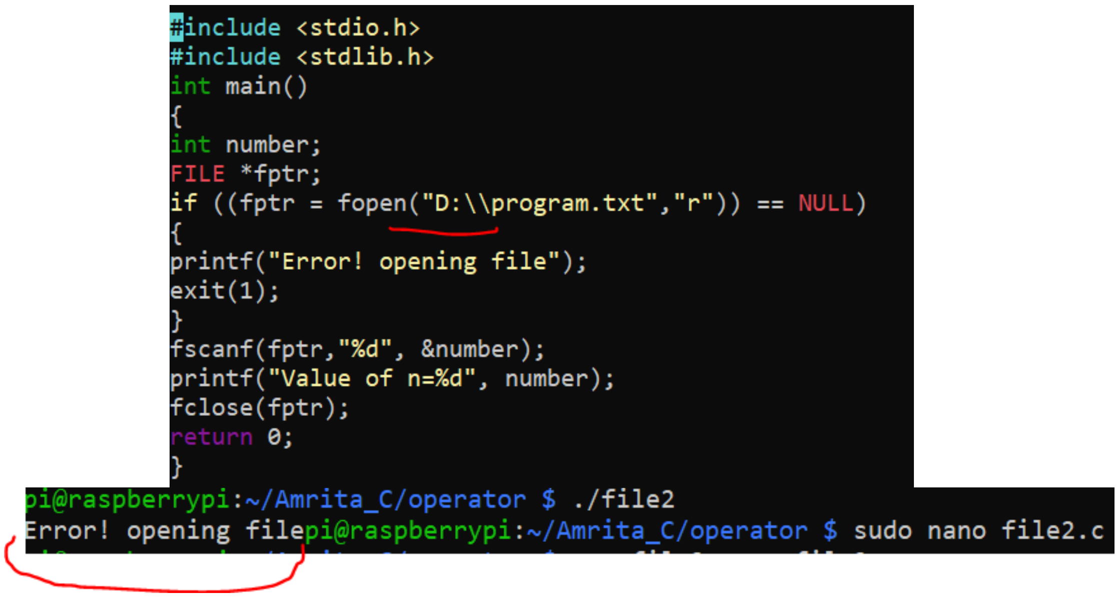
Read the number already stored in it that is 100.



```c
#include <stdio.h>
#include <stdlib.h>
int main()
{
int number;
FILE *fptr;
if ((fptr = fopen("E:\\program.txt","r")) == NULL)
{
printf("Error! opening file");
exit(1);
}
fscanf(fptr,"%d", &number);
printf("Value of n=%d", number);
fclose(fptr);
return 0;
}
```

```
pi@raspberrypi:~/Amrita_C/operator $ ./file2
Value of n=100pi@raspberrypi:~/Amrita_C/operator $ sudo nano file2.c
```

If we change the location from E drive to D drive then it produce error.

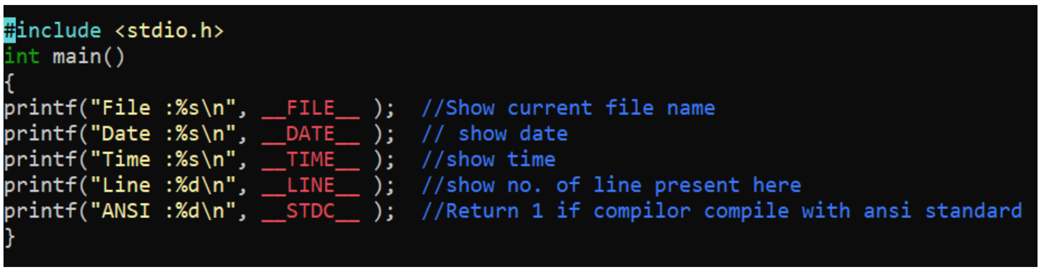


```c
#include <stdio.h>
#include <stdlib.h>
int main()
{
int number;
FILE *fptr;
if ((fptr = fopen("D:\\program.txt","r")) == NULL)
{
printf("Error! opening file");
exit(1);
}
fscanf(fptr,"%d", &number);
printf("Value of n=%d", number);
fclose(fptr);
return 0;
}
```

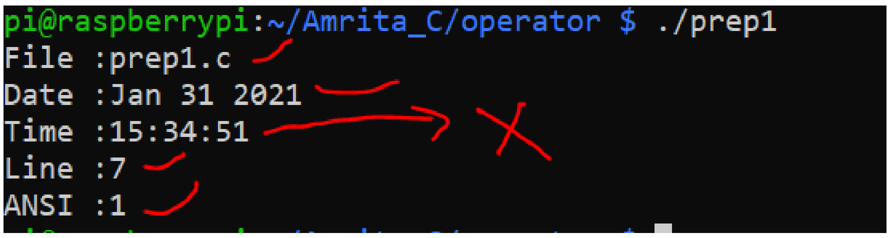
```
pi@raspberrypi:~/Amrita_C/operator $ ./file2
Error! opening filepi@raspberrypi:~/Amrita_C/operator $ sudo nano file2.c
```

Preprocessor:

Predefined macros:



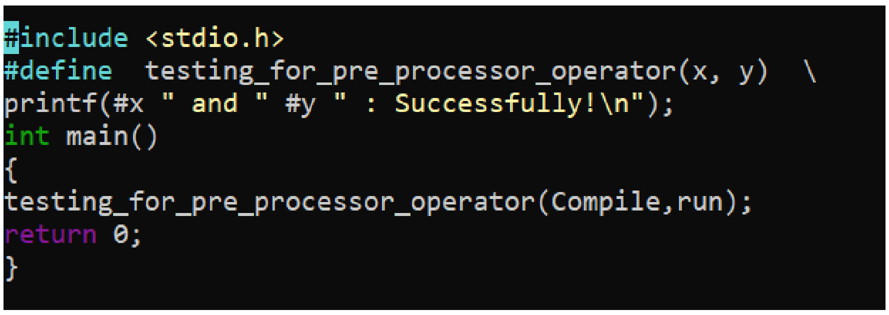
```
#include <stdio.h>
int main()
{
printf("File :%s\n", __FILE__ );  //Show current file name
printf("Date :%s\n", __DATE__ );  // show date
printf("Time :%s\n", __TIME__ );  //show time
printf("Line :%d\n", __LINE__ );  //show no. of line present here
printf("ANSI :%d\n", __STDC__ );  //Return 1 if compilor compile with ansi standard
}
```



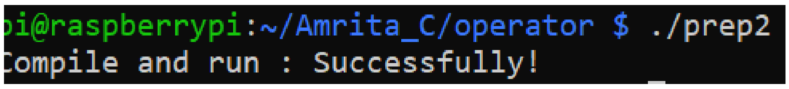
```
pi@raspberrypi:~/Amrita_C/operator $ ./prep1
File :prep1.c
Date :Jan 31 2021
Time :15:34:51
Line :7
ANSI :1
```

Preprocessor Operators:

Continuation and stringize(#) operator:

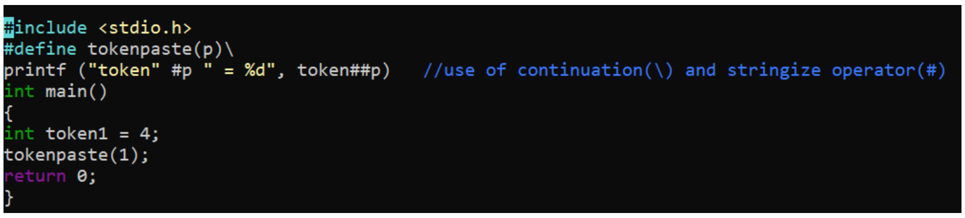


```
#include <stdio.h>
#define  testing_for_pre_processor_operator(x, y)  \
printf(#x " and " #y " : Successfully!\n");
int main()
{
testing_for_pre_processor_operator(Compile,run);
return 0;
}
```

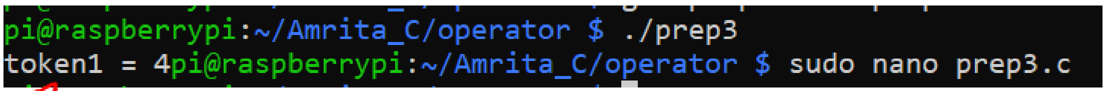


```
pi@raspberrypi:~/Amrita_C/operator $ ./prep2
Compile and run : Successfully!
```

Token paster operator:

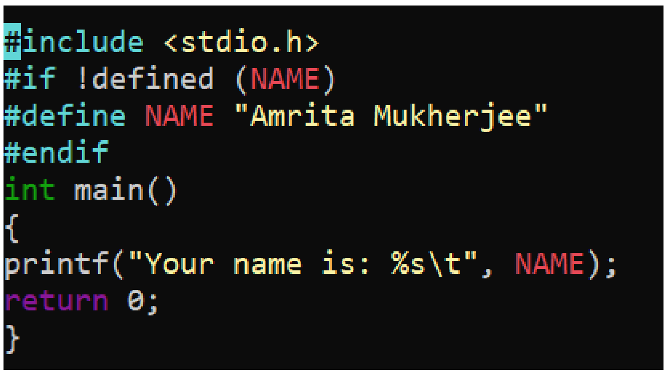


```
#include <stdio.h>
#define tokenpaste(p)\
printf ("token" #p " = %d", token##p)   //use of continuation(\) and stringize operator(#)
int main()
{
int token1 = 4;
tokenpaste(1);
return 0;
}
```

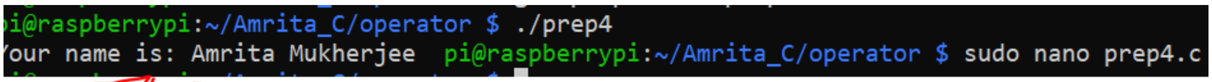


```
pi@raspberrypi:~/Amrita_C/operator $ ./prep3
token1 = 4pi@raspberrypi:~/Amrita_C/operator $ sudo nano prep3.c
```

Defined operator:

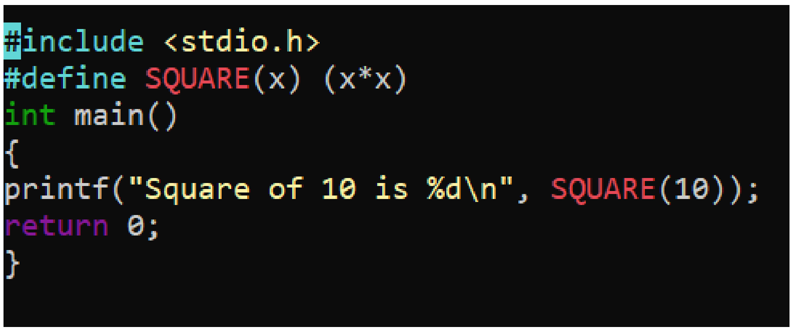


```
#include <stdio.h>
#if !defined (NAME)
#define NAME "Amrita Mukherjee"
#endif
int main()
{
printf("Your name is: %s\t", NAME);
return 0;
}
```

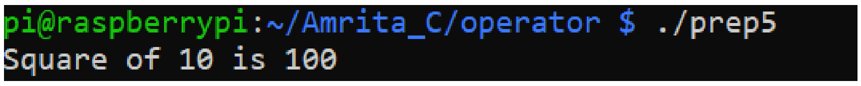


```
i@raspberrypi:~/Amrita_C/operator $ ./prep4
'our name is: Amrita Mukherjee  pi@raspberrypi:~/Amrita_C/operator $ sudo nano prep4.c
```

Parameterized Macros:

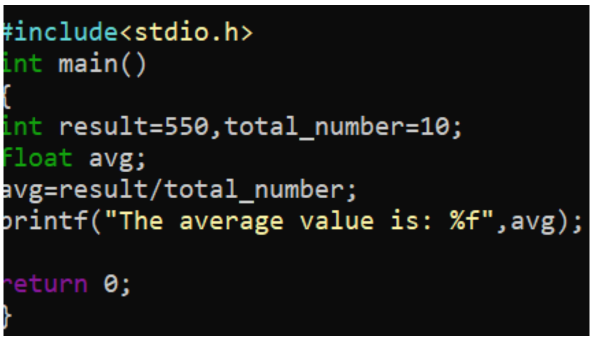


```
#include <stdio.h>
#define SQUARE(x) (x*x)
int main()
{
printf("Square of 10 is %d\n", SQUARE(10));
return 0;
}
```

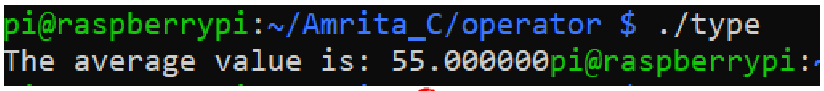


```
pi@raspberrypi:~/Amrita_C/operator $ ./prep5
Square of 10 is 100
```

Type casting: Here int first converted into float and then complete the operation and give float type data.

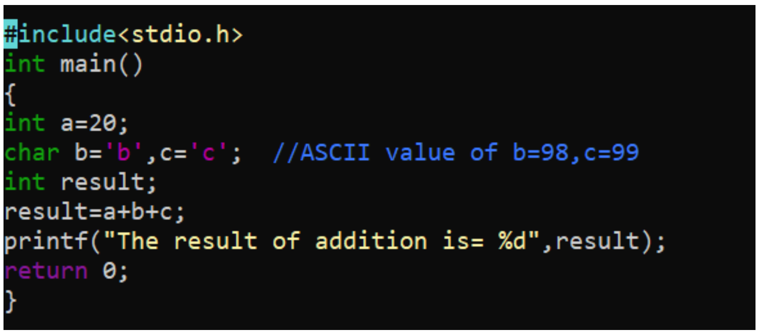


```
#include<stdio.h>
int main()
{
int result=550,total_number=10;
float avg;
avg=result/total_number;
orintf("The average value is: %f",avg);

eturn 0;
}
```

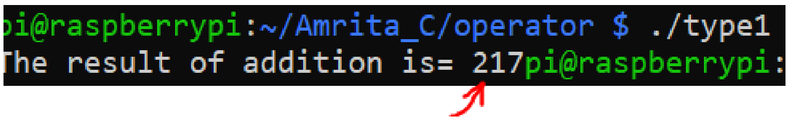


```
pi@raspberrypi:~/Amrita_C/operator $ ./type
The average value is: 55.000000pi@raspberrypi:
```

Integer promotion:



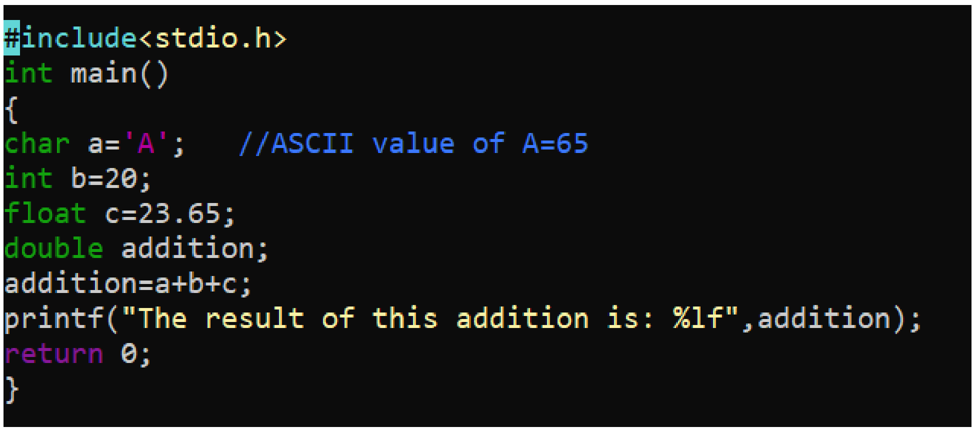
```
#include<stdio.h>
int main()
{
int a=20;
char b='b',c='c';  //ASCII value of b=98,c=99
int result;
result=a+b+c;
printf("The result of addition is= %d",result);
return 0;
}
```



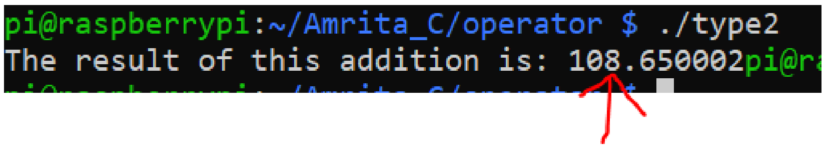
```
pi@raspberrypi:~/Amrita_C/operator $ ./type1
The result of addition is= 217pi@raspberrypi:
```

Arithmetic conversion: Here conversion will be like this:

Char→int-->float→double..and represent result as double. Cause double has highest place in hierarchy among all of them.

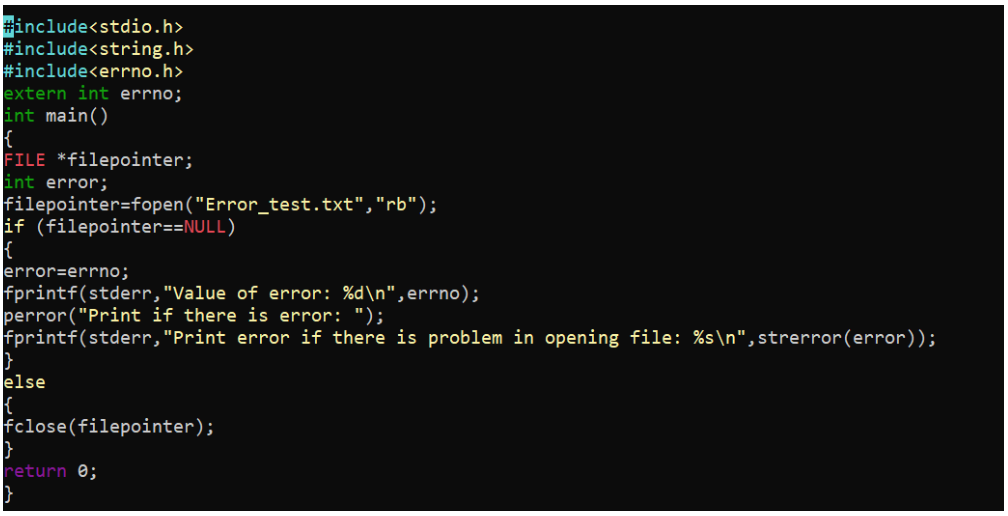


```
#include<stdio.h>
int main()
{
char a='A';   //ASCII value of A=65
int b=20;
float c=23.65;
double addition;
addition=a+b+c;
printf("The result of this addition is: %lf",addition);
return 0;
}
```

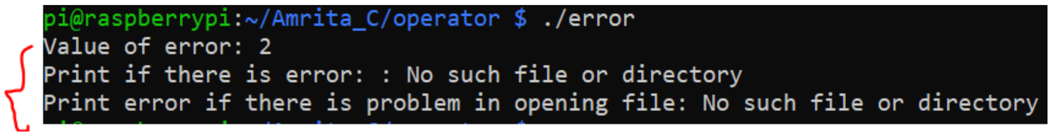


```
pi@raspberrypi:~/Amrita_C/operator $ ./type2
The result of this addition is: 108.650002pi@ra
```

Error handling:

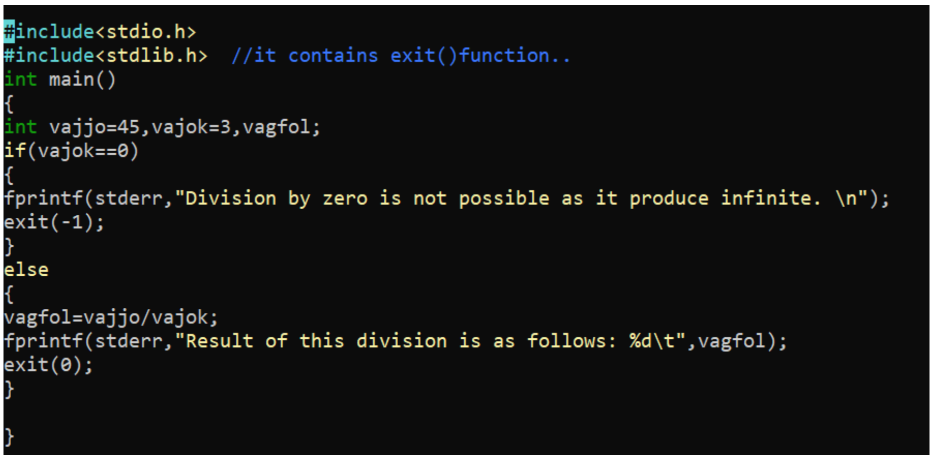


```c
#include<stdio.h>
#include<string.h>
#include<errno.h>
extern int errno;
int main()
{
FILE *filepointer;
int error;
filepointer=fopen("Error_test.txt","rb");
if (filepointer==NULL)
{
error=errno;
fprintf(stderr,"Value of error: %d\n",errno);
perror("Print if there is error: ");
fprintf(stderr,"Print error if there is problem in opening file: %s\n",strerror(error));
}
else
{
fclose(filepointer);
}
return 0;
}
```

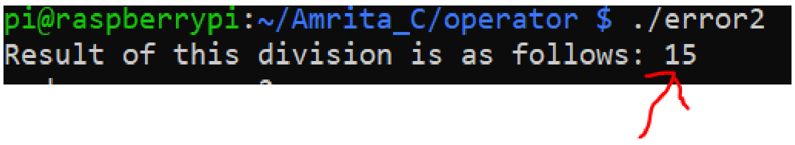


```
pi@raspberrypi:~/Amrita_C/operator $ ./error
Value of error: 2
Print if there is error: : No such file or directory
Print error if there is problem in opening file: No such file or directory
```

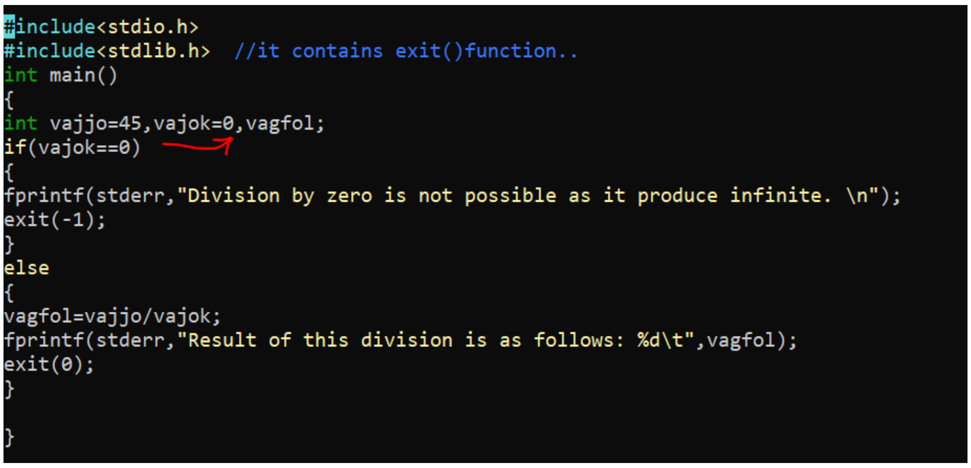
Divide by zero error:



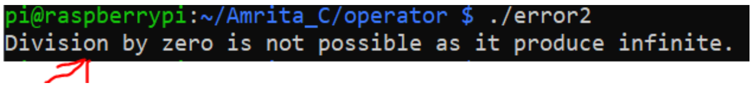
```c
#include<stdio.h>
#include<stdlib.h>  //it contains exit()function..
int main()
{
int vajjo=45,vajok=3,vagfol;
if(vajok==0)
{
fprintf(stderr,"Division by zero is not possible as it produce infinite. \n");
exit(-1);
}
else
{
vagfol=vajjo/vajok;
fprintf(stderr,"Result of this division is as follows: %d\t",vagfol);
exit(0);
}

}
```



```
pi@raspberrypi:~/Amrita_C/operator $ ./error2
Result of this division is as follows: 15
```

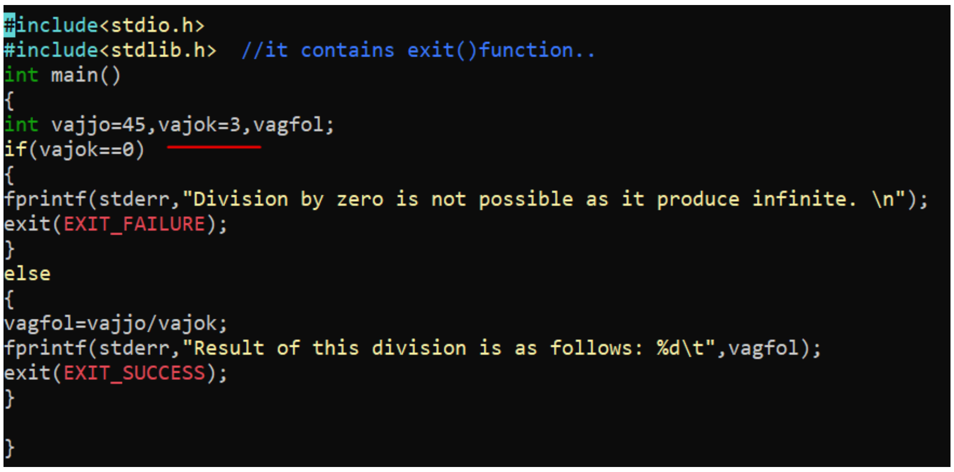


```c
#include<stdio.h>
#include<stdlib.h>  //it contains exit()function..
int main()
{
int vajjo=45,vajok=0,vagfol;
if(vajok==0)
{
fprintf(stderr,"Division by zero is not possible as it produce infinite. \n");
exit(-1);
}
else
{
vagfol=vajjo/vajok;
fprintf(stderr,"Result of this division is as follows: %d\t",vagfol);
exit(0);
}

}
```

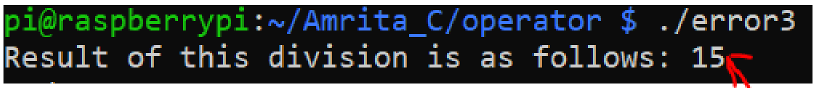


```
pi@raspberrypi:~/Amrita_C/operator $ ./error2
Division by zero is not possible as it produce infinite.
```

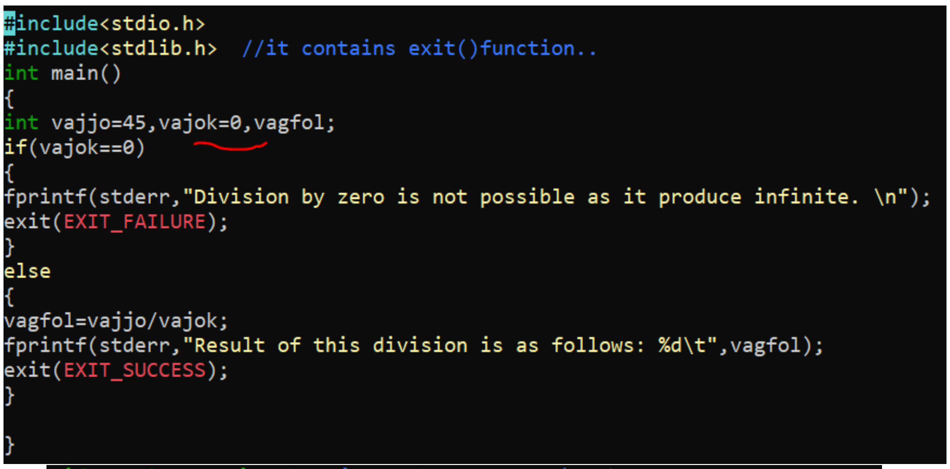
Program exit() status:



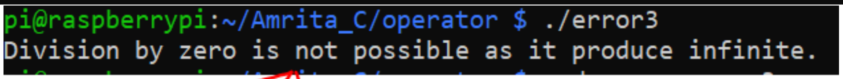
```c
#include<stdio.h>
#include<stdlib.h>  //it contains exit()function..
int main()
{
int vajjo=45,vajok=3,vagfol;
if(vajok==0)
{
fprintf(stderr,"Division by zero is not possible as it produce infinite. \n");
exit(EXIT_FAILURE);
}
else
{
vagfol=vajjo/vajok;
fprintf(stderr,"Result of this division is as follows: %d\t",vagfol);
exit(EXIT_SUCCESS);
}

}
```



```
pi@raspberrypi:~/Amrita_C/operator $ ./error3
Result of this division is as follows: 15
```



```c
#include<stdio.h>
#include<stdlib.h>  //it contains exit()function..
int main()
{
int vajjo=45,vajok=0,vagfol;
if(vajok==0)
{
fprintf(stderr,"Division by zero is not possible as it produce infinite. \n");
exit(EXIT_FAILURE);
}
else
{
vagfol=vajjo/vajok;
fprintf(stderr,"Result of this division is as follows: %d\t",vagfol);
exit(EXIT_SUCCESS);
}

}
```



```
pi@raspberrypi:~/Amrita_C/operator $ ./error3
Division by zero is not possible as it produce infinite.
```