

We will miss the Marvels of Mr C

ESC101: Fundamentals of Computing

Purushottam Kar

Announcements

- Joint tutorial for B1 and B5, Friday Nov 16th 12PM in L19
- End-sem Theory exam Nov 25th 9AM – 12noon
- AT final presentations Nov 26th – 29th (all welcome)
- Proration Alert
 - All students getting prorated would've received an email last night. If you applied for proration but didn't receive an email, **contact me now!**
 - Only medical reasons entertained for proration of theory exams, major quizzes, lab exams – as per announced policy on website
 - Students dropping AT prorated according to announced policy
 - Students unable to attend replacement lab due to the Prutor crash of Sep 11 prorated



Working in Linux Environment

3



Working in Linux Environment

3

Can store your stuff in files



Working in Linux Environment

3

Can store your stuff in files

File names have *extensions* telling us what kind of a file it is



Working in Linux Environment

3

Can store your stuff in files

File names have *extensions* telling us what kind of a file it is

Files that store C programs often have names that end with .c for example hello.c, pointers.c etc



Working in Linux Environment

3

Can store your stuff in files

File names have *extensions* telling us what kind of a file it is

Files that store C programs often have names that end with .c for example hello.c, pointers.c etc

Files that store images often have names that end with .jpg or .png or .bmp



Working in Linux Environment

3

Can store your stuff in files

File names have *extensions* telling us what kind of a file it is

Files that store C programs often have names that end with .c for example hello.c, pointers.c etc

Files that store images often have names that end with .jpg or .png or .bmp

Files that store videos often have names that end with .avi or .mp4



Working in Linux Environment

3

Can store your stuff in files

File names have *extensions* telling us what kind of a file it is

Files that store C programs often have names that end with .c for example hello.c, pointers.c etc

Files that store images often have names that end with .jpg or .png or .bmp

Files that store videos often have names that end with .avi or .mp4

Files that store PHP programs (another language) end with .php



Working in Linux Environment

3

Can store your stuff in files

File names have *extensions* telling us what kind of a file it is

Files that store C programs often have names that end with .c for example hello.c, pointers.c etc

Files that store images often have names that end with .jpg or .png or .bmp

Files that store videos often have names that end with .avi or .mp4

Files that store PHP programs (another language) end with .php

Files that store SQL commands (another language) end with .sql



Working in Linux Environment

3

Can store your stuff in files

File names have *extensions* telling us what kind of a file it is

Files that store C programs often have names that end with .c for example hello.c, pointers.c etc

Files that store images often have names that end with .jpg or .png or .bmp

Files that store videos often have names that end with .avi or .mp4

Files that store PHP programs (another language) end with .php

Files that store SQL commands (another language) end with .sql

.c, .jpg, .avi, .php are the extensions that tell us what that file stores



Working in Linux Environment

3

Can store your stuff in files

File names have *extensions* telling us what kind of a file it is

Files that store C programs often have names that end with .c for example hello.c, pointers.c etc

Files that store images often have names that end with .jpg or .png or .bmp

Files that store videos often have names that end with .avi or .mp4

Files that store PHP programs (another language) end with .php

Files that store SQL commands (another language) end with .sql

.c, .jpg, .avi, .php are the extensions that tell us what that file stores

Files arranged in folders/directories



Working in Linux Environment

3

Can store your stuff in files

File names have *extensions* telling us what kind of a file it is

Files that store C programs often have names that end with .c for example hello.c, pointers.c etc

Files that store images often have names that end with .jpg or .png or .bmp

Files that store videos often have names that end with .avi or .mp4

Files that store PHP programs (another language) end with .php

Files that store SQL commands (another language) end with .sql

.c, .jpg, .avi, .php are the extensions that tell us what that file stores

Files arranged in folders/directories

Directories can contain files as well as other directories



Working in Linux Environment

3

Can store your stuff in files

File names have *extensions* telling us what kind of a file it is

Files that store C programs often have names that end with .c for example hello.c, pointers.c etc

Files that store images often have names that end with .jpg or .png or .bmp

Files that store videos often have names that end with .avi or .mp4

Files that store PHP programs (another language) end with .php

Files that store SQL commands (another language) end with .sql

.c, .jpg, .avi, .php are the extensions that tell us what that file stores

Files arranged in folders/directories

Directories can contain files as well as other directories

Directories are used to organize your files neatly



Moving Around Directories

4



Moving Around Directories

4

pwd (print working directory) – print name of the directory you are currently sitting inside



Moving Around Directories

4

pwd (print working directory) – print name of the directory you are currently sitting inside

ls (list) – print all files and directories inside current directory



Moving Around Directories

4

pwd (print working directory) – print name of the directory you are currently sitting inside

ls (list) – print all files and directories inside current directory

ls -al (list all with nice list format) – same as **ls** but with more details and a more neat format



Moving Around Directories

4

pwd (print working directory) – print name of the directory you are currently sitting inside

ls (list) – print all files and directories inside current directory

ls -al (list all with nice list format) – same as **ls** but with more details and a more neat format

cd <dirname> (change directory) – move to the directory with name *dirname*



Moving Around Directories

4

pwd (print working directory) – print name of the directory you are currently sitting inside

ls (list) – print all files and directories inside current directory

ls -al (list all with nice list format) – same as **ls** but with more details and a more neat format

cd <dirname> (change directory) – move to the directory with name *dirname*

cd .. (change to parent directory) – move to the directory that contains the current directory



Moving Around Directories

4

pwd (print working directory) – print name of the directory you are currently sitting inside

ls (list) – print all files and directories inside current directory

ls -al (list all with nice list format) – same as **ls** but with more details and a more neat format

cd <dirname> (change directory) – move to the directory with name *dirname*

cd .. (change to parent directory) – move to the directory that contains the current directory

cd ~ (jump to home directory) – jump directly to home directory no matter what the current working directory



Directories and Files

5



Directories and Files

5

mkdir <newdir> (make directory) create a directory with name *newdir*. Make sure a directory with this name does not already exist



Directories and Files

5

mkdir <newdir> (make directory) create a directory with name *newdir*. Make sure a directory with this name does not already exist

rmdir <olddir> (remove directory) delete the directory with name *olddir*. Make sure *olddir* exists and is empty (i.e. does not contain any files or folders)



Directories and Files

5

mkdir <newdir> (make directory) create a directory with name *newdir*. Make sure a directory with this name does not already exist

rmdir <olddir> (remove directory) delete the directory with name *olddir*. Make sure *olddir* exists and is empty (i.e. does not contain any files or folders)

You can create .c files using your favorite editor e.g. gedit, vi, emacs, vim, nano, kate etc



Directories and Files

5

mkdir <newdir> (make directory) create a directory with name *newdir*. Make sure a directory with this name does not already exist

rmdir <olddir> (remove directory) delete the directory with name *olddir*. Make sure *olddir* exists and is empty (i.e. does not contain any files or folders)

You can create .c files using your favorite editor e.g. gedit, vi, emacs, vim, nano, kate etc

rm <filename> remove the file with name *filename*



Directories and Files

5

mkdir <newdir> (make directory) create a directory with name *newdir*. Make sure a directory with this name does not already exist

rmdir <olddir> (remove directory) delete the directory with name *olddir*. Make sure *olddir* exists and is empty (i.e. does not contain any files or folders)

You can create .c files using your favorite editor e.g. gedit, vi, emacs, vim, nano, kate etc

rm <filename> remove the file with name *filename*

rm -r <dirname> remove the directory *dirname* (and all that it contains), recursively, even if the directory is not empty. WARNING: use with care.



Creating and Running C programs 6



Creating and Running C programs 6

Write C programs exactly as we did on Prutor



Creating and Running C programs 6

Write C programs exactly as we did on Prutor

Have to work with the gcc compiler now (unless you install the clang compiler yourself)



Creating and Running C programs 6

Write C programs exactly as we did on Prutor

Have to work with the gcc compiler now (unless you install the clang compiler yourself)

Warning: gcc does not give as helpful warning messages



Creating and Running C programs 6

Write C programs exactly as we did on Prutor

Have to work with the gcc compiler now (unless you install the clang compiler yourself)

Warning: gcc does not give as helpful warning messages

Some differences may exist between gcc and clang



Creating and Running C programs 6

Write C programs exactly as we did on Prutor

Have to work with the gcc compiler now (unless you install the clang compiler yourself)

Warning: gcc does not give as helpful warning messages

Some differences may exist between gcc and clang

Major difference: need to give output after running code (in Prutor, input was given before executing code)



Creating and Running C programs 6

Write C programs exactly as we did on Prutor

Have to work with the gcc compiler now (unless you install the clang compiler yourself)

Warning: gcc does not give as helpful warning messages

Some differences may exist between gcc and clang

Major difference: need to give output after running code (in Prutor, input was given before executing code)

Your microprocessor cannot run C programs directly. It can only run what is known as an *executable file*



Compilation and Execution

7



Compilation and Execution

7

Suppose your C program is in the file test.c



Compilation and Execution

7

Suppose your C program is in the file `test.c`

To compile the C program type **`gcc test.c`**



Compilation and Execution

7

Suppose your C program is in the file `test.c`

To compile the C program type **`gcc test.c`**

It will create an executable file called `a.out`



Compilation and Execution

7

Suppose your C program is in the file `test.c`

To compile the C program type **`gcc test.c`**

It will create an executable file called `a.out`

Can execute that file by typing **`./a.out`**



Compilation and Execution

7

Suppose your C program is in the file `test.c`

To compile the C program type **`gcc test.c`**

It will create an executable file called `a.out`

Can execute that file by typing **`./a.out`**

If you want to give the executable a nice name, use the following command **`gcc -o myname test.c`**



Compilation and Execution

7

Suppose your C program is in the file `test.c`

To compile the C program type **`gcc test.c`**

It will create an executable file called `a.out`

Can execute that file by typing **`./a.out`**

If you want to give the executable a nice name, use the following command **`gcc -o myname test.c`**

An executable called *myname* will get created



Compilation and Execution

7

Suppose your C program is in the file `test.c`

To compile the C program type **`gcc test.c`**

It will create an executable file called `a.out`

Can execute that file by typing **`./a.out`**

If you want to give the executable a nice name, use the following command **`gcc -o myname test.c`**

An executable called *myname* will get created

Can execute that file by typing **`./myname`**



C files and Executable Files

8



C files and Executable Files

8

```
#include <stdio.h>
int main(){
    int a = 2, b = 3, c;
    c = a + b;
    printf("%d\n", c);
    return 0;
}
```



C files and Executable Files

8

```
#include <stdio.h>
int main(){
    int a = 2, b = 3, c;
    c = a + b;
    printf("%d\n", c);
    return 0;
}
```

```
400526: 55          push %rbp
400527: 48 89 e5     mov  %rsp,%rbp
40052a: 48 83 ec 10   sub  $0x10,%rsp
40052e: c7 45 f4 02 00 00 00 movl $0x2,-0xc(%rbp) ; Mr C is doing a = 2 here (in hexadecimal)
400535: c7 45 f8 03 00 00 00 movl $0x3,-0x8(%rbp) ; Mr C is doing b = 3 here (in hexadecimal)
40053c: 8b 55 f4     mov  -0xc(%rbp),%edx
40053f: 8b 45 f8     mov  -0x8(%rbp),%eax
400542: 01 d0       add  %edx,%eax ; Mr C is doing c = a + b here
400544: 89 45 fc     mov  %eax,-0x4(%rbp)
400547: 8b 45 fc     mov  -0x4(%rbp),%eax
40054a: 89 c6       mov  %eax,%esi
40054c: bf f4 05 40 00 mov  $0x4005f4,%edi
400551: b8 00 00 00 00 mov  $0x0,%eax
400556: e8 a5 fe ff ff callq 400400 <printf@plt> ; Mr C is calling the printf function here
40055b: b8 00 00 00 00 mov  $0x0,%eax
400560: c9         leaveq
400561: c3         retq
400562: 66 2e 0f 1f 84 00 00 nopw %cs:0x0(%rax,%rax,1)
400569: 00 00 00
40056c: 0f 1f 40 00 nopl 0x0(%rax)
```

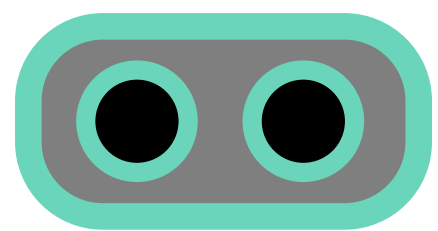


C files and Executable Files

8

```
#include <stdio.h>
int main(){
    int a = 2, b = 3, c;
    c = a + b;
    printf("%d\n", c);
    return 0;
}
```

```
400526: 55          push %rbp
400527: 48 89 e5     mov  %rsp,%rbp
40052a: 48 83 ec 10   sub  $0x10,%rsp
40052e: c7 45 f4 02 00 00 00 movl $0x2,-0xc(%rbp) ; Mr C is doing a = 2 here (in hexadecimal)
400535: c7 45 f8 03 00 00 00 movl $0x3,-0x8(%rbp) ; Mr C is doing b = 3 here (in hexadecimal)
40053c: 8b 55 f4     mov  -0xc(%rbp),%edx
40053f: 8b 45 f8     mov  -0x8(%rbp),%eax
400542: 01 d0       add  %edx,%eax ; Mr C is doing c = a + b here
400544: 89 45 fc     mov  %eax,-0x4(%rbp)
400547: 8b 45 fc     mov  -0x4(%rbp),%eax
40054a: 89 c6       mov  %eax,%esi
40054c: bf f4 05 40 00 mov  $0x4005f4,%edi
400551: b8 00 00 00 00 mov  $0x0,%eax
400556: e8 a5 fe ff ff callq 400400 <printf@plt> ; Mr C is calling the printf function here
40055b: b8 00 00 00 00 mov  $0x0,%eax
400560: c9         leaveq
400561: c3         retq
400562: 66 2e 0f 1f 84 00 00 nopw %cs:0x0(%rax,%rax,1)
400569: 00 00 00
40056c: 0f 1f 40 00 nopl 0x0(%rax)
```



C files and Executable Files

8

```
#include <stdio.h>
int main(){
    int a = 2, b = 3, c;
    c = a + b;
    printf("%d\n", c);
    return 0;
}
```

```
400526: 55          push %rbp
400527: 48 89 e5     mov  %rsp,%rbp
40052a: 48 83 ec 10   sub  $0x10,%rsp
40052e: c7 45 f4 02 00 00 00 movl $0x2,-0xc(%rbp) ; Mr C is doing a = 2 here (in hexadecimal)
400535: c7 45 f8 03 00 00 00 movl $0x3,-0x8(%rbp) ; Mr C is doing b = 3 here (in hexadecimal)
40053c: 8b 55 f4     mov  -0xc(%rbp),%edx
40053f: 8b 45 f8     mov  -0x8(%rbp),%eax
400542: 01 d0       add  %edx,%eax ; Mr C is doing c = a + b here
400544: 89 45 fc     mov  %eax,-0x4(%rbp)
400547: 8b 45 fc     mov  -0x4(%rbp),%eax
40054a: 89 c6       mov  %eax,%esi
40054c: bf f4 05 40 00 mov  $0x4005f4,%edi
400551: b8 00 00 00 00 mov  $0x0,%eax
400556: e8 a5 fe ff ff callq 400400 <printf@plt> ; Mr C is calling the printf function here
```

On the right is a more human readable version of the executable file corresponding to the C program

```
400569: 00 00 00     nopl 0x0(%rax,%rax,1)
40056c: 0f 1f 40 00   nopl 0x0(%rax)
```



C files and Executable Files

8

```
#include <stdio.h>
```

```
int main(){
```

```
    int a = 2, b = 3, c;
```

```
    c = a + b;
```

```
    printf("%d\n", c);
```

```
    return 0;
```

```
}
```

```
400526: 55          push %rbp
400527: 48 89 e5     mov  %rsp,%rbp
40052a: 48 83 ec 10   sub  $0x10,%rsp
40052e: c7 45 f4 02 00 00 00 movl $0x2,-0xc(%rbp) ; Mr C is doing a = 2 here (in hexadecimal)
400535: c7 45 f8 03 00 00 00 movl $0x3,-0x8(%rbp) ; Mr C is doing b = 3 here (in hexadecimal)
40053c: 8b 55 f4     mov  -0xc(%rbp),%edx
40053f: 8b 45 f8     mov  -0x8(%rbp),%eax
400542: 01 d0       add  %edx,%eax ; Mr C is doing c = a + b here
400544: 89 45 fc     mov  %eax,-0x4(%rbp)
400547: 8b 45 fc     mov  -0x4(%rbp),%eax
40054a: 50          %eax,%esi
40054b: 50          $0x4005f4,%edi
40054c: 50          $0x0,%eax
40054d: 00 40 00 <printf@plt> ; Mr C is calling the printf function here
40054e: 50          %eax
40054f: 50          $0x0(%rax,%rax,1)
400552: 00 00 00
400555: 0f 1f 40 00   nopl 0x0(%rax)
```

Your microprocessor cannot execute a C program so I convert your C program to an executable

On the right is a more human readable version of the executable file corresponding to the C program



Linux and Windows

9



Linux and Windows

9

On Linux systems, gcc compiler will always be available



Linux and Windows

9

On Linux systems, gcc compiler will always be available
Can install clang too if you own that machine



Linux and Windows

9

On Linux systems, gcc compiler will always be available

Can install clang too if you own that machine

On Windows, can install gcc via the Cygwin or the MinGW routes



Linux and Windows

9

On Linux systems, gcc compiler will always be available

Can install clang too if you own that machine

On Windows, can install gcc via the Cygwin or the MinGW routes

<https://sourceforge.net/projects/tdm-gcc/>



Linux and Windows

9

On Linux systems, gcc compiler will always be available

Can install clang too if you own that machine

On Windows, can install gcc via the Cygwin or the MinGW routes

<https://sourceforge.net/projects/tdm-gcc/>

Can install Clang on Windows too



Linux and Windows

9

On Linux systems, gcc compiler will always be available

Can install clang too if you own that machine

On Windows, can install gcc via the Cygwin or the MinGW routes

<https://sourceforge.net/projects/tdm-gcc/>

Can install Clang on Windows too

<http://releases.lldvm.org/download.html>



Linux and Windows

9

On Linux systems, gcc compiler will always be available

Can install clang too if you own that machine

On Windows, can install gcc via the Cygwin or the MinGW routes

<https://sourceforge.net/projects/tdm-gcc/>

Can install Clang on Windows too

<http://releases.llvm.org/download.html>

However, easier to install Visual Studio on Windows



Linux and Windows

9

On Linux systems, gcc compiler will always be available

Can install clang too if you own that machine

On Windows, can install gcc via the Cygwin or the MinGW routes

<https://sourceforge.net/projects/tdm-gcc/>

Can install Clang on Windows too

<http://releases.lldvm.org/download.html>

However, easier to install Visual Studio on Windows

<https://visualstudio.microsoft.com/vs/community/>



Pursue your Interest

10



Pursue your Interest

10

If interested in problem solving – take up competitive programming



Pursue your Interest

10

If interested in problem solving – take up competitive programming

<https://www.hackerrank.com/>



Pursue your Interest

10

If interested in problem solving – take up competitive programming

<https://www.hackerrank.com/>

<https://leetcode.com/>



Pursue your Interest

10

If interested in problem solving – take up competitive programming

<https://www.hackerrank.com/>

<https://leetcode.com/>

<https://www.codechef.com/>



Pursue your Interest

10

If interested in problem solving – take up competitive programming

<https://www.hackerrank.com/>

<https://leetcode.com/>

<https://www.codechef.com/>

<https://www.geeksforgeeks.org/>



Pursue your Interest

10

If interested in problem solving – take up competitive programming

<https://www.hackerrank.com/>

<https://leetcode.com/>

<https://www.codechef.com/>

<https://www.geeksforgeeks.org/>

<http://codeforces.com/>



Pursue your Interest

10

If interested in problem solving – take up competitive programming

<https://www.hackerrank.com/>

<https://leetcode.com/>

<https://www.codechef.com/>

<https://www.geeksforgeeks.org/>

<http://codeforces.com/>

Many application areas require heavy programming



Pursue your Interest

10

If interested in problem solving – take up competitive programming

<https://www.hackerrank.com/>

<https://leetcode.com/>

<https://www.codechef.com/>

<https://www.geeksforgeeks.org/>

<http://codeforces.com/>

Many application areas require heavy programming

Fluid dynamics (AE, CHE), Particle accelerators (PHY), Data Analysis (MTH, CSE), Wireless communications (EE), Optimization (IME, ECO, MTH, CSE)



Pursue your Interest

10

If interested in problem solving – take up competitive programming

<https://www.hackerrank.com/>

<https://leetcode.com/>

<https://www.codechef.com/>

<https://www.geeksforgeeks.org/>

<http://codeforces.com/>

Many application areas require heavy programming

Fluid dynamics (AE, CHE), Particle accelerators (PHY), Data Analysis (MTH, CSE), Wireless communications (EE), Optimization (IME, ECO, MTH, CSE)

Several applications within CSE



Pursue your Interest

10

If interested in problem solving – take up competitive programming

<https://www.hackerrank.com/>

<https://leetcode.com/>

<https://www.codechef.com/>

<https://www.geeksforgeeks.org/>

<http://codeforces.com/>

Many application areas require heavy programming

Fluid dynamics (AE, CHE), Particle accelerators (PHY), Data Analysis (MTH, CSE), Wireless communications (EE), Optimization (IME, ECO, MTH, CSE)

Several applications within CSE

Architecture, Operating Systems, Compilers, Theory, Algorithm Design, Databases, Web Programming, Machine Learning and AI, and many more



There is no Mr C 😊

11



There is no Mr C 😊

11

We introduced the character of Mr C to make the discussion fun and engaging – no Mr C in reality



There is no Mr C 😊

11

We introduced the character of Mr C to make the discussion fun and engaging – no Mr C in reality

When we said Mr C, some times we were referring to the compiler, sometimes the runtime environment, and sometimes the Prutor application



There is no Mr C 😊

11

We introduced the character of Mr C to make the discussion fun and engaging – no Mr C in reality

When we said Mr C, some times we were referring to the compiler, sometimes the runtime environment, and sometimes the Prutor application

Please use the term “compiler” from now on. People will not understand you if you keep referring to Mr C



There is no Mr C 😊

11

We introduced the character of Mr C to make the discussion fun and engaging – no Mr C in reality

When we said Mr C, some times we were referring to the compiler, sometimes the runtime environment, and sometimes the Prutor application

Please use the term “compiler” from now on. People will not understand you if you keep referring to Mr C

Also, no clones of Mr C are born when functions get called



There is no Mr C 😊

11

We introduced the character of Mr C to make the discussion fun and engaging – no Mr C in reality

When we said Mr C, some times we were referring to the compiler, sometimes the runtime environment, and sometimes the Prutor application

Please use the term “compiler” from now on. People will not understand you if you keep referring to Mr C

Also, no clones of Mr C are born when functions get called

When functions are invoked, the control gets passed from one function to another along with arguments as input



No More Prutor after a few days

12



No More Prutor after a few days

12

Prutor logins for all of us will be disabled a few days after the grades are declared – will warn you beforehand!



No More Prutor after a few days

12

Prutor logins for all of us will be disabled a few days after the grades are declared – will warn you beforehand!

New logins will be created for your friends in A batch



No More Prutor after a few days

12

Prutor logins for all of us will be disabled a few days after the grades are declared – will warn you beforehand!

New logins will be created for your friends in A batch

Need to start using other compilers like gcc



No More Prutor after a few days

12

Prutor logins for all of us will be disabled a few days after the grades are declared – will warn you beforehand!

New logins will be created for your friends in A batch

Need to start using other compilers like gcc

Can keep using Clang as well (need to install it though) – gcc is always available on Linux computers



No More Prutor after a few days

12

Prutor logins for all of us will be disabled a few days after the grades are declared – will warn you beforehand!

New logins will be created for your friends in A batch

Need to start using other compilers like gcc

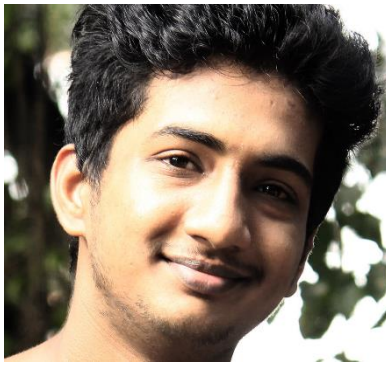
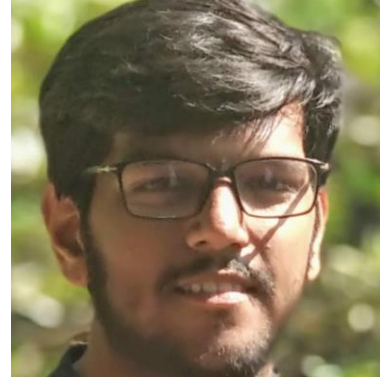
Can keep using Clang as well (need to install it though) – gcc is always available on Linux computers

Windows computers need to install a compiler (VS etc)



Acknowledgements - Tutors

13



Acknowledgements – TAs

14

THEORY TAs

AMIT CHANDAK
AVIDEEP MUKHERJEE
DESAI ANERI HIREN
DIXIT KUMAR
GOURAV TAKHAR
NIRJHAR ROY
PANKAJ SIWAN
PRAFULLA SAXENA
PRATEEK DWIVEDI
RAHUL SHARMA
SOUMYA BANERJEE
VIKAS MAURYA
WASEEM AKRAM

LAB TAs

ABHISHEK GUPTA
ABHISHEK KUMAR
ADITYA JAIN
AMIT KUMAR
ANKIT KUMAR
ANKIT SHARMA
ANKUR
APPU B
ARUNKUMAR VEDIAPPAN
ASHISH PAL
AVIJIT ROY
BIDYA SARKAR
DARSHAK CHHATBAR
DARSHIT SHAILESH VAKIL

DEEPAK YADAV
DEEPAK YADAV
HARIKRISHNAN BALAGOPAL
HARIOM
HARSHIT GUPTA
HEMANT PARIHAR
HIMANSHU SARAIYA
HIMANSHU SINGLA
I G PRASAD
JATIN ASWAL
JATIN DEV
JIMMY KUMAR
KAPIL KUMAR
KULDEEP SAHU
MANISH MAZUMDER

MAYANK RAWAT
MIRIYALA JEEVAN KUMAR
MOHIT MALHOTRA
NAMAN NARANG
NIKHIL SHAGRITHAYA
NITIN VIVEK BHARTI
NITISH MANGESH KALAN
PRABHATH NAMPALLY
PRANJAL JAIN
PRATEEK SAMAIYA
RAHUL KUMAR SINGH
RAJAT
SAGAR
SURAJIT GHOSH



Acknowledgements - Admins

15



**Umair Z.
Ahmed**

Prutor Admin
and Co-creator
of Prutor



**Prof. Amey
Karkare**

Head CC and
Co-creator of
Prutor



**Bhaskar
Mukhoty**

Workflow
Admin



**Prof. Piyush
Rai**

Faculty Tutor



Acknowledgements – Staff Members

CC staff members: Mr Gopesh Tiwari, Mr Soumitri Mishra

NCL staff members: Mr Awanish Kumar

CSE Office staff members: Mr Prashant Kumar Sahu, Mr Rajesh Kumar, Mr Ranjan Kumar, Mr Anubhav Kumar Arya, Mr Amit Kumar Bharti



Acknowledgements

17

Prutor Team: Prof. Karkare, Umair

Previous instructors: Prof. Sumit Ganguly, Prof. Nitin Saxena, Prof. Amey Karkare, Prof. Indranil Saha, Prof. Nisheeth Srivastava, Prof. Rajat Mittal (Hindi lectures), Prof. Vinay Namboodiri

Material in this offering was based on their material 😊

Piazza Team: helpful for discussions and clearing doubts

Gradescope Team: extremely vital for massive courses

