**جلسه اول - مقدمه درس**

**کارایی برنامه:**

در درس الگوریتم و ساختمان داده مورد بررسی قرار می‌گیرد. با موازی سازی تا حدی می توان کارایی را افزایش داد.

خوانایی برنامه (Readability)

کاربردپذیری

نگهداری برنامه (Maintenance)

امنیت برنامه

مقیاس‌پذیری (scalability) که در درس مهندسی اینترنت مورد بررسی است.

درستی عملکرد

کتابخانه‌های آماده

قابلیت تغییر برنامه

به عنوان مثال استفاده از global variable چندان توصیه نمی شود.

دستور Go to که در زبان های قدیمی تر موجود بود و کنترل اجرای برنامه را پیچیده می کرد.

تست نرم افزار را برای لحظه آخر نگه ندارید.

**تاکید درس: نوشتن برنامه های خوب**

**Correctness: Testing, Debugging**

**Maintainability: Object orientation, coding style**

**Reusability: object orientation**

از قطعات نرم‌افزار بتوانیم مجددا استفاده کنیم.

زبان اصلی: C++

JAVA هم کاربردی است.

زبان ها: C++، C#، Python، MATLAB، JAVA، R

C++ کارایی C (تاکید روی سرعت و استفاده بهینه از منابع) را با شی‌گرایی و انعطاف پذیری ترکیب کرده است.

کتاب:

Deitel: C++ How to program

Google Search

Stackoverflow.com

Cplusplus.com

Ramtung.ir/apnotes/html

**جلسه دوم - مفاهیم مقدماتی زبان ++C**

**سلام دنیا:** یک برنامه‌ی بسیار ساده در زبان سی‌پلاس‌پلاس که صرفاً یک رشته را در خروجی می‌نویسد.



Iostream هدرفایلی است که امکانات ورودی و خروجی C++ را در اختیار ما می‌گذارد.

قاعده قرار دادن .h قدیمی است و تقریبا منسوخ شده است.

Std مخفف استاندارد است و تمام امکانات استاندارد C++ در این فضای نام قرار دارد.

فضاهای نام برای پرهیز از تداخل بین کتابخانه‌های نرم افزاری مختلف مورد استفاده قرار می‌گیرند.

اگر خط دوم را ننویسم بایستی هربار بنویسیم std::cout

**امصا تابع: نام تابع، نوع پارامتر ورودی، نوع پارامتر خروجی**

در مورد خط چهارم فعلا بدانیم که امکان پاس کردن آرگومان به تابع main از طریق command line وجود دارد.

در C++ بلافاصله بعد از امضای تابع و در خط بعدی، آکلاد را باز می کنیم. این کار در جاوا درست بعد از امضای تابع و در همان خط انجام می گیرد.

در داخل بدنه‌ی تابع، یک تب (TAB) قرار می‌دهیم.

Cout مخفف console out است که اصطلاح اطلاق شده به همان ترمینال یا command line می باشد.

G++ کامپایلر C++ در لینوکس. فایل اجرایی در لینوکس پسوند .out و در ویندوز پسوند .exe دارد.

بازگرداندن صفر اطمینان از موفقیت آمیز بودن اجرای برنامه را تضمین می کند.

**تایپ رشته - خواندن از ورودی:** این مثال استفاده اولیه از تایپ رشته و نحوه‌ی خواندن از ورودی را نشان می‌دهد.

string هدرفایلی است که امکان کار روی رشته‌ها را به ما می‌دهد.

Name شی (object) از نوع string است.

تابع cin فقط کلمه اول از ورودی را می گیرد. کلمه دنباله ای از کاراکترها است که به فضای سفید محدود می‌شود.

**فضای سفید: space، TAB، Enter**

Cin کاراکترهای غیر فضای سفید را می‌خواند.

خواندن کل یک خط با تابع getline انجام می‌گیرد.

**خواندن چند قلم از ورودی:** در این مثال چند مقدار به دنبال هم از ورودی خوانده می‌شود. دقت کنید که اگر رشته‌ای از cin خوانده شود، یک کلمه از ورودی خوانده شده در آن متغیر قرار می‌گیرد. مثلاً اگر ورودی Gholam 29 در ورودی تایپ شود مقدار name برابر Gholam و مقدار age برابر 29 خواهد بود.



Endl ثابتی که در هدرفایل iostream تعریف شده است و با کاراکتر ‘\n’ مترادف است.

**تایپ ده انگشتی**

چنانچه return 0 را برنگردانید، C++ خود به صورت پیش‌فرض مقدار صفر را برمی‌گرداند.

البته این ویژگی چندان خوب نیست. لذا سعی کنید همواره بصورت explicit مقدار صفر را برگردانید و چنانچه تابع قرار است مقدار خروجی نداشته باشد، نوع خروجی را void تعریف کنید.

**خواندن از ورودی در حلقه:** این برنامه تعدادی کلمه را از ورودی می‌خواند و در صورتی که کلمه‌ای تکرار شود این موضوع را با نمایش پیغامی اطلاع می‌دهد.

خواندن ورودی تا آنجا ادامه می‌یابد که کاربر با ctrl-d (یا ctrl-z در ویندوز) خاتمه ورودی را مشخص کند.



در برنامه بالا تا زمانی که اینتر نزنیم، C++ به خواندن از ورودی ادامه می‌هد و پس از آن پردازش خط بعد را آغاز می‌کند.

خواندن ورودی از فایل، پاس کردن به فایل اجرایی در لینوکس و نوشتن خروجی در فایل

a.out<masalan.txt>output.txt

نمایش مختویات فایل text در لینوکس

Cat output.txt

پاس کردن خروجی برنامه به تابع sort لینوکس ([piping in linux](https://www.google.com/search?q=piping+in+linux&rlz=1C1GCEA_enFR801DE802&oq=piping+in+linux&aqs=chrome..69i57.5678j0j8&sourceid=chrome&ie=UTF-8))

a.out<masalan.txt|sort

از مشکلات آرایه این است که اندازه آن باید ثابت باشد. جهت حل این مشکل کتابخانه استاندارد C++ وکتورها را در اختیار ما قرار می دهد. تایپ vector جزئی از کتابخانه‌ی استاندارد سی‌پلاس‌پلاس است که کار با دنباله‌ای از عناصر را **بدون دغدغه‌های مدیریت حافظه** مهیا می‌کند.

**محاسبه‌ی میانگین و میانه:** در این برنامه تعدادی عدد اعشاری (که نماینده‌ی دما هستند) از ورودی گرفته می‌شود و میانگین و میانه‌ی آنها در خروجی نوشته می‌شود. تعداد این اعداد در زمان نوشتن برنامه نامعلوم است.



در سر فایل algorithm عده‌ای از توابع همچون sort که در برنامه بالا در خط 18 مورد استفاده قرار گرفته، تعریف شده‌اند.

**مقداردهی اولیه‌ی بردارها:** این مثال نشان‌دهنده‌ی مقداردهی اولیه‌ی اندازه و عناصر بردار است.



در مثال بالا اگر خط 9 را کامنت نکنیم با RUNTIME ERROR مواجه خواهیم شد. زیرا در لحظه‌ی تعریف، vector شامل صفر عنصر است و با هر pushback حافظه تخصیص می‌یابد.

Segmentation fault

خطای دسترسی غیرمجاز به حافظه در لینوکس است.

در خط 6 یک بردار ساخته‌ایم که دارای 6 عنصر integer است و مقدار اولیه‌ی هریک از این عناصر صفر است.

خط 26 با خطای کامپایلر روبرو می‌گردد. چون 99 رشته نیست. اما "99" یا ‘99’ رشته است.

در خط 28، برداری ساخته‌ایم که 1000 عنصر اولیه آن با مقدار اعشاری 2/1 پر شده است. خط 29 هم با خطای segmentation (دسترسی غیرمجاز به حافظه) روبرو است.

بردار دو بعدی:

Vector<vector<int>>twod(3);

Twod[0].push\_back(12);

Twod[1].push\_back(4);

Twod[2].push\_back(65);

Cout<< Twod[1][0]<<endl;

Vector<vector<int>>twod(3,vector<int>(4));

وکتوری با 3 عنصر اولیه بساز و هرکدام از عناصر اولیه‌ی آن را با یک بردار چهارتایی از نوع مقدار صحیح، مقدار اولیه بده.

**تمرین:** برنامه‌ای بنویسید که تعدادی کلمه را از ورودی بخواند و آن‌ها را در قالب یک لیست که با کاما جدا شده‌اند چاپ کند. به عنوان مثال اگر کلمات ورودی به ترتیب Aang و Kyoshi و Roku باشند خروجی چاپ شده باید دقیقاً به صورت [Aang, Kyoshi, Roku] باشد. توجه داشته باشید که تعداد کلمه‌ها مشخص نیست و تا انتهای ورودی بخوانید.



خط اول [سرفایل iostream](https://en.cppreference.com/w/cpp/header/iostream) را به سورس اضافه می‌کند. این سرفایل بخشی از [کتابخانه ورودی/خروجی](https://en.cppreference.com/w/cpp/io) است.

خط دوم [سرفایل sstream](https://en.cppreference.com/w/cpp/header/sstream) را به سورس اضافه می‌کند که آن نیز بخشی از کتابخانه ورودی/خروجی است.

[سرفایل string](https://en.cppreference.com/w/cpp/header/string) بخشی از [کتابخانه رشته‌ها](https://en.cppreference.com/w/cpp/string) است.

با قطعه کد

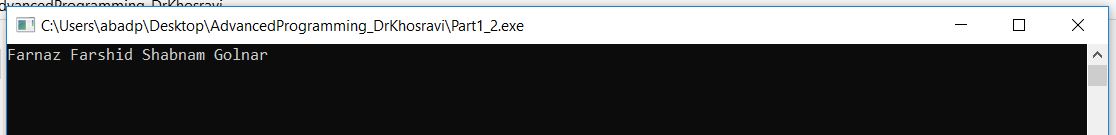
std::getline(std::cin, line);

ورودی را تا زمانی که کاربر Enter بزند، خوانده و در line ذخیره می‌کنیم.



**Get line from stream into string**

Extracts characters from *is* and stores them into *str* until the delimitation character *delim* is found (or the newline character, '\n', for *(2)*).



لذا وقتی کاربر Enter می‌زند، داریم:

Line = “Farnaz Farshid Shabnam Golnar”

سپس آنچه را در بافر ورودی است به نام stream می‌نامیم و با line مقدار دهی می‌کنیم.

std::istringstream stream{ line };

با کد

stream >> s;

کلمات (آنچه محصور بین دو فضای سفید است) را از بافر ورودی استخراج می‌کنیم و در s می‌ریزیم.

خطوط 11 و 12

stream >> s;

std::cout << "[" << s;

اجرا می‌شوند و داریم s = Farnaz و چاپ می‌شود:

[Farnaz

سپس وارد حلقه while می‌شویم. **چون کاراکتر >> فضای سفید را ندید می‌گیرد**، مقداری که در شروع حلقه در s ذخیره است عبارت است از s=Farshid و فاصله بین Farnaz و Farshid ندید گرفته شده است. لذا به دنبال [Farnaz کاراکتر ‘, ’ و بعد از آن Farshid نوشته می‌شود:

[Farnaz, Farshid

در دومین تکرار حلقه S=Shabnam و داریم:

[Farnaz, Farshid, Shabnam

و در سومین تکرار s=Golnar و داریم:

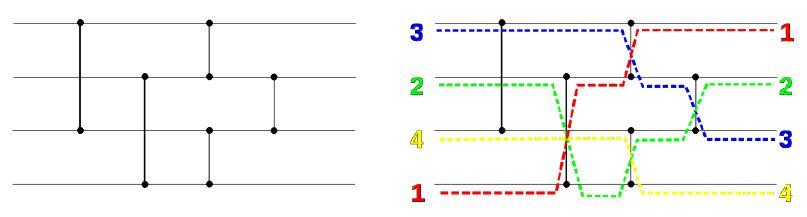
[Farnaz, Farshid, Shabnam, Golnar

دیگر وارد حلقه نمی‌شویم. چون در جریان ورودی stream از نوع istringstream(input string stream) چیزی باقی نمانده است. از حلقه خارج شده و به دنبال مواد بالا، ] نیز چاپ می‌شود.

[Farnaz, Farshid, Shabnam, Golnar]

**جلسه سوم - طراحی بالا به پایین**

**شبکه‌های مرتب‌سازی و sorting network:** شبکه‌ی مرتب‌سازی یک مدل انتزاعی ریاضی شامل شبکه‌ای از سیم‌ها و واحدهای مقایسه کننده است که برای مرتب‌سازی دنباله‌ای از اعداد ار آن استفاده می‌شود. هر مقایسه کننده دو سیم را به هم متصل می‌کند و مقادیر را با قرار دادن مقدار کوچکتر روی یکی از سیم‌ها و مقدار بزرگتر روی سیم دیگر، مرتب می‌کند. هر سیم دارای یک مقدار می باشد، و هر مقایسه کننده دو سیم را به عنوان ورودی و خروجی می‌گیرد. زمانی که دو مقدار وارد مقایسه کننده می‌شود، مقایسه کننده مقدار کوچکتر را در سیم بالاتر، و مقدار بزرگتر را در سیم پایین قرار می‌دهد. به شبکه‌ای از سیم‌ها و مقایسه‌کننده‌ها که به طور صحیح تمام مقادیر ورودی را به صورت صعودی مرتب کنند یک شبکه مرتب‌سازی گفته می‌شود.

شکل زیر (چپ)، نشان‌دهنده یک شبکه مرتب‌سازی است. سیم‌ها در این شکل به صورت افقی و مقایسه کننده‌ها به صورت عمودی نشان داده شده‌اند. مراحل انجام مرتب‌سازی توسط این شبکه در شکل راست نشان داده شده است. فهم چگونگی صحیح عمل کردن این شبکه مرتب‌سازی آسان است. این را مدنظر داشته باشید که مقایسه کننده‌ها مقدار بزرگ را به سیم پایین و مقدار کوچک را به سیم بالا منتقل می‌کنند.

هدف این تمرین این است که عملکرد یک شبکه را روی یک دنباله‌ی ورودی شبیه‌سازی کنیم. در ورودی، شبکه و دنباله‌ی اعداد داده می‌شوند. خروجی برنامه تعیین می‌کند که آیا شبکه‌ی داده شده اعداد را به درستی مرتب می‌کند یا نه. توضیح این که لزوماً هر شبکه، به طور صحیح اعداد را مرتب نمی‌کند. مثال‌هایی از حالت‌های درست و نادرست در نمونه‌های زیر ذکر می‌شوند. علاوه بر این، ممکن است در توصیف شبکه‌ی ورودی خطاهایی وجود داشته باشد که برنامه‌ی شما باید آن‌ها را تشخیص دهد.

برای توصیف یک شبکه‌ی ورودی، از ماتریسی از کاراکترها استفاده می‌کنیم. به عنوان مثال شبکه‌ی نشان داده شده در شکل فوق توسط ماتریس زیر توصیف می‌شود:

a-c-

-bce

a-de

-bd-

هر سطر از ماتریس یک سیم را مشخص می‌کند. هر کاراکتر از یک سطر یا ’–’ است که نشان دهنده‌ی عدم وجود مقایسه کننده در آن بخش است یا با یک حرف لاتین مشخص میشود که در این صورت، باید در یکی (و تنها یکی) دیگر از کاراکترهای آن ستون کاراکتر مشابهی پیدا شود. به این ترتیب، فرض می شود یک واحد مقایسه‌کننده بین سیم های متناظر آن دو سطر وجود دارد. به عنوان مثال، در توصیف فوق یک مقایسه کننده بین دو سیم اول و سوم وجود دارد که با حرف a مشخص شده. ستون دوم نیز وجود یک مقایسه کننده بین سیم‌های دوم و چهارم است. ترتیب انتقال اعداد از چپ به راست فرض می شود.

در توصیف داده شده از شبکه ممکن است خطاهایی به شرح زیر وجود داشته باشد که برنامه‌ی شما باید آنها را تشخیص دهد:

* در یک ستون تنها یک مورد از یک حرف پیدا می‌شود یا این که بیش از دو مورد از آن حرف وجود دارد.
* در توصیف شبکه کاراکتری غیر از حروف کوچک لاتین و ’–’ وجود دارد.

دقت کنید که دو مقایسه کننده در دو ستون مختلف می توانند با یک حرف یکسان نمایش داده شوند، چون این امر ابهامی در عملکرد شبکه ایجاد نمی‌کند.

**ورودی**

خط اول هر مورد آزمون حاوی دو عدد صحیح N و K است که به ترتیب تعداد سطرها و تعداد ستون‌های ماتریس شبکه را مشخص می‌کنند. کنند. بعد از آن N خط پشت سر هم می‌آیند که هریک از K کاراکتر تشکیل شده‌اند. سپس در یک خط N عدد صحیح که به ترتیب روی سیم‌های 1 تا N قرار خواهند گرفت ذکر می‌شوند. در ورودی مسئله ممکن است تعداد بیش از یک مورد آزمون ذکر شود که هریک از قالب فوق پیروی می‌کند. آخرین خط ورودی شامل دو عدد صفر است.

**خروجی**

برای هر مورد آزمون، یک خط در خروجی بنویسید که با قالبی مانند آنچه در بخش «نمونه خروجی» آمده، یکی از سه نتیجه‌ی ممکن را مشخص کند: Not Sorted به این معنی که اعداد ورودی پس از پردازش توسط شبکه مرتب نمی‌شوند، Sorted به این معنی که اعداد ورودی پساز پردازش توسط شبکه مرتب می‌شوند و Invalid Network به این معنی که ورودی داده شده قوانین مطرح شده در توصیف مساله را نقض می‌کند.

|  |  |
| --- | --- |
| **نمونه ورودی** | **نمونه خروجی** |
|  |  |
|  |  |
|  |  |

دقت کنید

برنامه‌ی خود را فقط در قالب یک فایل به زبان سی پلاس پلاس تحویل دهید.

به جز کتابخانه‌های استاندارد زبان سی پلاس پلاس و std\_lib\_facilities.h از هدرفایل دیگری استفاده نکنید.

قبل از تحویل، برنامه‌ی خود را با انواع ورودی‌ها بیازمایید. حالت‌های مرزی ورودی‌های مسئله را در نظر بگیرید.

به قالب خروجی دقت کنید. وجه چیزی به جز آنچه 􀀁 ها مهم هستند. به هیچ 􀀁 های خالی و شکست خط 􀀁 بزرگی و کوچکی حروف، فاصله