بنام خدا – تدرسیار درس برنامهنویسی C++ – استاد درس: دکتر بهرام طاهری
دانشگاه صنعتی امیرکبیر – دانشکده ی مهندسی برق

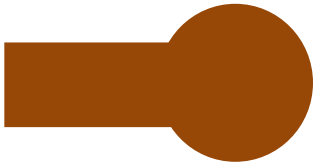
یادآوری: تمامی تمرینات و اطلاعات مربوط به تحویل آنها بصورت هفتگی در سایت درس قرار داده میشوند:

http://ele.aut.ac.ir/~btaheri/cpp/

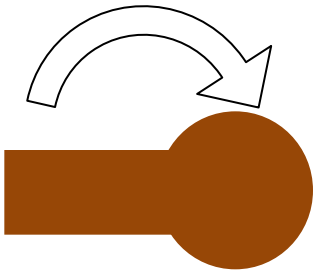
توجه: هنگام ارسال برنامه،کافیست فقط فایل .cpp رو ارسال کنید. (به هیچ وجه فایل .exe که فایل اجرایی است که توسط کامپایلر ساخته شده است را نفرستید!)

سوالات برنامه نویسی

۱ – **هدف:** برنامه نویسی گرافیکی:
برنامه ای بنویسید که کلید اتاق آقای دکتر طاهری را رسم کند:



۲ – **هدف:** برنامه نویسی گرافیکی:
برنامه ای بنویسید که شکل مثال قبل را با یک سرعت خط مشخص که از ورودی گرفته میشود، شکل فوق را بچرخاند.



اطلاعات اضافی: هدف در این مساله چرخش با هم مستطیل و دایره، با هم است، بطوریکه چرخش همزمان و هماهنگ آنها این ذهنیت را ایجاد کند که کلید در حال چرخش است. لذا به دلخواه، همانطور که خودتان صلاح می دانید، در یک نقطه ی دلخواه، یک محور دوران برای شکل درنظر بگیرید.

۳ – **هدف:** کار با آرایه ها:
تابعی برای محاسبه ی حاصلضرب دو ماتریس ۵ در ۵ بنویسید.
تابع باید دارای سه ورودی باشد:

```
void matrixMult(int A[][5], int B[][5], int result[][5])
{
// your code here
}
```

در حقیقت تابع دو آرایه ی دو بعدی A و B را به عنوان ورودی دریافت می کند و مقدار حاصل را در result[][] قرار می دهد. لازم به یادآوری است که ارجاع آرایه به تابع به صورت Call by Reference است؛ یعنی تمامی مقادیری که در داخل تابع در result[][] ذخیره کنید، در بیرون برنامه قابل دسترسی است. همچنین به یاد داریم که نمی توان هیچ موقع یک آرایه را به عنوان خروجی یک تابع (قبل از نام آن) تعیین کرد. مثلا ساختار زیر اشتباه است:

```
int [][5] matrixMult(int A[][5], int B[][5])
{

}
```

برای همین است که result[][] را به عنوان آرگومان ورودی به تابع می دهیم و با ذخیره ی مقادیر محاسبه شده در آن، مقادیر آن را به عنوان نتیجه ی برنامه در بدنه ی اصلی برنامه استفاده می کنیم. مثلا می توانید برای آزمایش برنامه ی خود، چیزی شبیه به برنامه ی زیر بنویسید:

```
// include headers here

void matixMult(int A[][5], int B[][5], int result[][5])
{
// your code here
}
int main()
{
   int A[5][5] = {{1,2,3,4,5}, {6,7,8,9,10}, {5,4,3,2,1}, {1,1,1,1,1}, {10,9,8,7,6}};
   int B[5][5] = {{6,7,8,9,10}, {5,4,3,2,1}, {1,1,1,1,1}, {10,9,8,7,6}, {1,2,3,4,5}};
   int C[5][5];
   matrixMult(A,B,C);

  // write a simple code to print matrix C here !
}
```

۴ – **هدف:** کار با آرایه:

برنامه ای بنویسید که ابتدا تعدادی عدد از ورودی بگیرد و آنها را در یک آرایه قرار دهد. سپس یک عدد صحیح در محدوده تعداد اعداد وارد شده ازکاربر دریافت کند. برنامه باید شامل یک تابع به اسم `getArray()` باشد . تابع باید در ورودی یک آرایه، طول آن و یک عدد دلخواه n، را بگیرد. با گرفتن یک عدد صحیح(n) تابع باید مقدار آرایه در خانه ی n ام را حذف کرده و همه ی مقادیر موجود در خانه ی n+1 تا آخر را یکی به عقب شیفت دهد. در انتها باید بعد از اجرای تابع، در داخل main() مقدار مقادیر جدید خانه های آرایه را چاپ کنید.

وارد کردن عدد 0 به معنی اتمام وارد کردن اعداد از ورودی برنامه است. (خود آن محسوب نمی گردد)

نمونه:

INPUT:
Input values of the array:
1 19 20 30 20 3 0
Input n:
3

OUTPUT:
1 19 30 20 3

بدنه ی برنامه را به این صورت تعریف کنید.

```
// include headers here

void deleteOneMember(int A[], int length)
{
// your code here
}
int main()
{
   int arrayName[1000], arrayLength;
   // get array values here

  // write a simple code to print A here !

   deleteOneMember(A, arrayLength);

  // write a simple code to print A here !
}
```

۵ – **هدف:** کار با آرایه ها:

تابعی بنویسید که یک ماتریس را از ورودی دریافت کند و بعد از دوران آن به اندازه ی ۱۸۰ درجه آن را در خروجی چاپ کند.

```
// include headers here

void transpose(int A[], int size)
{
// your code here
}
int main()
{
   // sample input:
   int arraySample[3][3] = {{1,2,3},{4,5,6},{7,8,9}}, matrixSize=3;

   // write a simple code to print arraySample here !

   transpose(arraySample, matrixSize);

  // write a simple code to print arraySample here !
}
```

برای این سوال و سوالات بعدی حتما صفحات ۲۱۰ به بعد کتاب رو بخونید.

۶ – **هدف:** کار با رشته ها:

برنامه ای بنویسید که یک جمله را از ورودی گرفته و موارد زیر را چاپ کند:

- تعداد کلمات
- تعداد کاراکترهای کوچک (بدون در نظر گرفتن فاصله ها)
- تعداد کاراکتر های بزرگ (بدون در نظر گرفتن فاصله ها)
- تعداد اعداد

اطلاعات اضافی: تفاوت بین چند ساختار پدرسوخته برای گرفتن مقادیر ورودی!

۱. متداول ترین ساختار برای گرفتن مقادیر ورودی که تابحال با آن کار کرده ایم، عبارت است از:

```
string strVar;
cin >> strVar;
cout << strVar;
```

در صورتی که بعد از کامل کردن و اجرای برنامه، مقدار زیر را به به برنامه دهیم:

```
Bahram shahram 123!
```

در خروجی خواهیم داشت:

```
Bahram
```

در حقیقت ساختار فوق از `cin` تنها قسمت اول از رشته را تا قبل از رسیدن به فاصله از رشته ی ورودی می گیرد و بقیه را دور می ریزد!

یک اشتباه متداول: یک دانشجو فکر میکنه که برنامه ی مقابل حتما همیشه از ورودی دو مقدار رشته ای رو می گیره و در متغیر ها به ترتیب ذخیره می کنه.

```
string strVar1, strVar2;
cin >> strVar1;
cout << strVar1;
cin >> strVar2;
cout << strVar2;
```

دانشجو برنامش رو اجرا می کنه و ابتدا می نویسه !salam Hassan و دکمه ی تایید رو می زنه. همین که میاد یه رشته ی دیگه رو بزنه تا وارد `strVar2` بشه، در عین ناباوردی می بینه که برنامه تموم شده(مقدار دیگری به `strVar2`) تخصیص داده شده. (بعد از این مساله دانشجو اعصابش به شدت از این مساله خورد شده و داره به تدریسیار فحش میده!) حاج آقا (استاد) درس رو اینطوری توضیح میده:

اگر ما مقداری به cin اولی بدهیم که خارج از ظرفیت آن باشد، و لذا قسمتی از سرریز شود، مقدارسرریز شده در بافر(یک حافظه ی میانی) کامپیوتر جمع می شود. لذا زمانی که اجرای برنامه به cin دومی می رسد، مقادیر جا مانده از روی بافر خوانده می شود(به جای اینکه از ورودی درخواست مقدار جدیدی شود.) لذا استاد درس چاره ی رفع مشکل در چنین شرایطی را پاک کردن بافر کامپیوتر معرفی می کند.

۲. ساختار بعدی به صورت زیر است:

```
string str = cin.get();
```

این ساختار فقط یک کاراکتر را می گیرد و بقیه را در نظر نمی گیرد.

۳. ساختار پدر سوخته ی آخر به صورت زیر است:

```
string str = cin.get();
```

این ساختار همان چیزی است که معمولا به آن در گرفتن یک جمله (همراه با فاصله بین کلمات) احتیاج داریم.

طرز استفاده از آن در صفحه ی ۲۱۱ کتاب بیان شده است.

۷ – **هدف:** کار با string ها

تابعی بنویسید که یک رشته بگیرد و بعد از عکس کردن آن، مقدار عکس شده را به عنوان خروجی بدهد. تابع را در بدنه ی یک برنامه ی کامل به کار ببرید و نشان دهید که به صورتی صحیح عمل می کند.

```
// include headers here

string inverse(string str)
{
// your code here
}
int main()
{
   // sample input:
   string sampleStr = "Hi! Nice day! Ha?!";

   // write a simple code to print sampleStr here !

   SampleStr = inverse(sampleStr);

   // write a simple code to print sampleStr here !
}
```

۸ – **هدف:** کار با string ها

تابعی بنویسید که با دریافت یک رشته از ورودی (string) تعداد کلماتی که با 'A' شروع می شوند را بشمارد. مشابه سوال ۷ تابع را در بدنه ی یک برنامه ی کامل به کار ببرید و نشان دهید که به صورتی صحیح عمل می کند.

۹ – **هدف:** کار با string ها

تابعی بنویسید که یک رشته(string) و یک عدد صحیح(int) و یک کاراکتر(char) را به عنوان ورودی بگیرد، و بعد از جایگزینی کاراکتر ورودی، در مکان تعیین شده در ورودی تابع، رشته ی حاصل را به عنوان خروجی برگرداند. مشابه مثال ۷ تابع را در بدنه ی یک برنامه ی کامل به کار ببرید و نشان دهید که به صورتی صحیح عمل می کند.

۱۰ -**هدف:** کار با رشته ها: برنامه ی رمز سزار:

امپراطور سزار در زمان های قدیم برای اینکه نامه هایش را کسی نتواند بخواند آنها را به اینصورت کد می کرده است: بجای حرف اول، حرف بعد از خود و به جای حرف دوم، ۲ حرف بعد از خود، به جای حرف سوم، سه حرف بعد از خودش را قرار داده و سپس از اول شروع می کرد. برای مثال اگر کلمه Book باشد، کد شده ی این کلمه به صورت Cqrl می باشد که C حرف بعد از B، q دو حرف بعد از o، r سه حرف بعد از o ، l حرف بعد از k می باشد.

سوال: تابعی بنویسید که یک کلمه را بگیرد و آن را رمز گشایی کرده و بنویسد.

```
// include headers here

string decode(string str)
{
// your code here
}
int main()
{
   // sample input:
   string code = "Cqrl";

   // write a simple code to print code here !

   string decoded = decode(code);

   // write a simple code to print decoded here.  It should be Book!
}
```