دستور کار کارگاه مبانی برنامه‌نویسی

احسان شجاع

تابستان ۱۴۰۳

فهرست آزمایش‌ها

آزمایش ۱
نصب و راه‌اندازی محیط برنامه‌نویسی و اجرای اولین برنامه به زبان C

۱-۱ اهداف

۱. آشنایی و آماده‌سازی محیط‌های توسعه و برنامه‌نویسی به زبان C

۲. نوشتن، کامپایل و اجرای اولین برنامه به زبان C

۱-۲ توضیحات

برنامه‌نویسی به زبان C جالب و سرگرم کننده است. در این آزمایش، شما می‌بایست بتوانید یک نمونه برنامه به زبان C را ایجاد، ویرایش، کامپایل و اجرا نمایید. قبل از اینکه یادگیری زبان C را شروع کنیم، لازم است تا حداقل یک کامپایلر زبان C را بر روی سیستم خود داشته باشیم.

برای ایجاد و اجرای یک برنامه C حداقل نیاز به دو نرم‌افزار داریم:

۱. یک ویرایشگر متن (مانند Notepad، Notepad++، Vim، Gedit، TextEdit، Sublime و ...)

۲. یک کامپایلر زبان C (مانند GCC، Clang، Borland Turbo C، Intel C++ و ...)

کامپایلر GCC

کامپایلر GCC (G++) یک کامپایلر مطمئن، کارا و محبوب در بین توسعه‌دهندگان برنامه به زبان‌های C (C++) است. این کامپایلر به صورت متن‌باز بوده و تقریبا امکان استفاده از آن در تمامی سیستم‌های عامل محبوب به ویژه سیستم‌های سازگار با استاندارد POSIX[1] فراهم می‌باشد. POSIX در واقع مجموعه‌ای از استانداردهای مشخص شده برای ایجاد سازگاری مابین سیستم عامل‌های مختلف است. برخی از سیستم عامل‌ها نظیر Unix و Linux تا حد خوبی از این استانداردها تبعیت نموده و با آن سازگار می‌باشند (هرچند که ممکن است دارای گواهی رسمی سازگاری نباشند). از سیستم عامل‌های دارای گواهی سازگاری در زمان آماده‌سازی این نوشته می‌توان به سیستم عامل Mac OS (ورژن ۱۰ به بعد) اشاره نمود. سیستم عامل ویندوز به طور کامل با استاندارد POSIX سازگار نبوده و واسط متفاوتی (WinAPI) را ارائه می‌دهد.

کامپایلر GCC در کنار GNU C Library برنامه‌ها را بر اساس استاندارد POSIX کامپایل می‌کند. بنابراین امکان نصب مستقیم و استفاده از این کامپایلر بر روی سیستم عامل ویندوز وجود ندارد. برای نصب GCC بر روی سیستم عامل ویندوز می‌بایست یکی از دو روش زیر اعمال شود:

Cygwin: کتابخانه‌ای به ویندوز اضافه گردد که با استفاده از تبدیل فراخوانی‌های POSIX به فراخوانی‌های Windows به نوعی سازگاری POSIX را به ویندوز اضافه کند. اکنون می‌توان برنامه‌های لینوکس نظیر GCC را با استفاده از این کتابخانه بر روی سیستم عامل ویندوز استفاده نمود.

[1] Portable Operating System Interface

Mingw, MSYS: کدهای سیستمی POSIX به نوعی بازنگاری شوند که خود شامل نگاشت POSIX به Window باشند. در این صورت کامپایلر یک فایل اجرایی بومی ایجاد کرده و برنامه برای اجرا دیگر نیازی به کتابخانه جانبی جهت تبدیل فراخوانی‌ها نخواهد داشت.

محیط توسعه یکپارچه Codelite

محیط‌های توسعه یکپارچه یا IDE[۲] ها ابزارهایی هستند که استفاده از ویرایشگر متن، کامپایلر و سایر قابلیت‌ها و امکانات را تحت یک نرم‌افزار جامع در اختیار برنامه‌نویس قرار می‌دهند. در این درس در کنار یادگیری روش‌های اولیه برای نوشتن، کامپایل و اشکال‌زدایی؛ برای راحتی کار از محیط توسعه یکپارچه Codelite نیز استفاده خواهیم نمود. مراحل نصب و استفاده از این IDE بر روی سیستم عامل ویندوز مختصرا به صورت زیر است:

۱. نصب پلتفرم ساخت نرم‌افزار MSYS2 (پلتفرم پیشنهادی توسط نرم‌افزار Codelite)

```
https://www.msys2.org/#installation
```

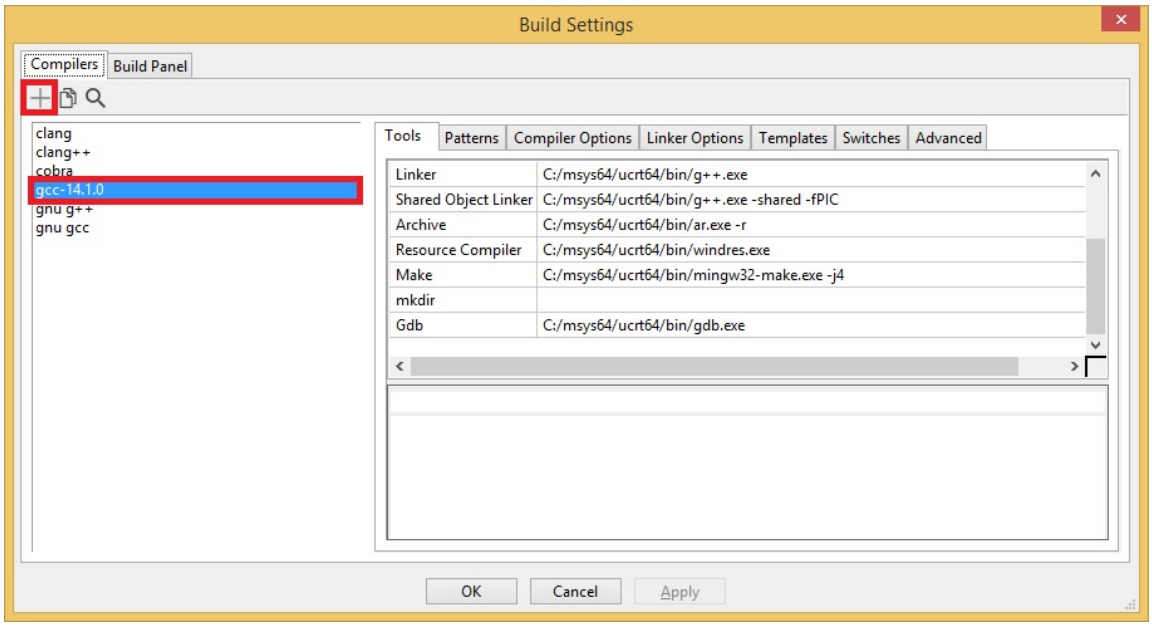
توجه داشته باشید که نسخه نصب شده می‌بایست با سیستم‌عامل شما سازگار باشد:

```
https://www.msys2.org/docs/windows_support
https://www.msys2.org/docs/updating
```

۲. نصب آخرین نسخه کامپایلر GCC با استفاده از MSYS2

```
pacman -Sy mingw-w64-ucrt-x86_64-gcc
pacman -Sy mingw-w64-ucrt-x86_64-gdb
pacman -Sy mingw-w64-ucrt-x86_64-make
echo 'export PATH=/ucrt64/bin:$PATH' >> ~/.$(basename $SHELL)rc
. ~/.$(basename $SHELL)rc
```

۳. نصب نرم‌افزار Codelite و افزودن کامپایلر GCC نصب شده (Settings > Build Settings...) در صورتیکه به صورت اتوماتیک تشخیص داده نشود (شکل ۱ - ۱).

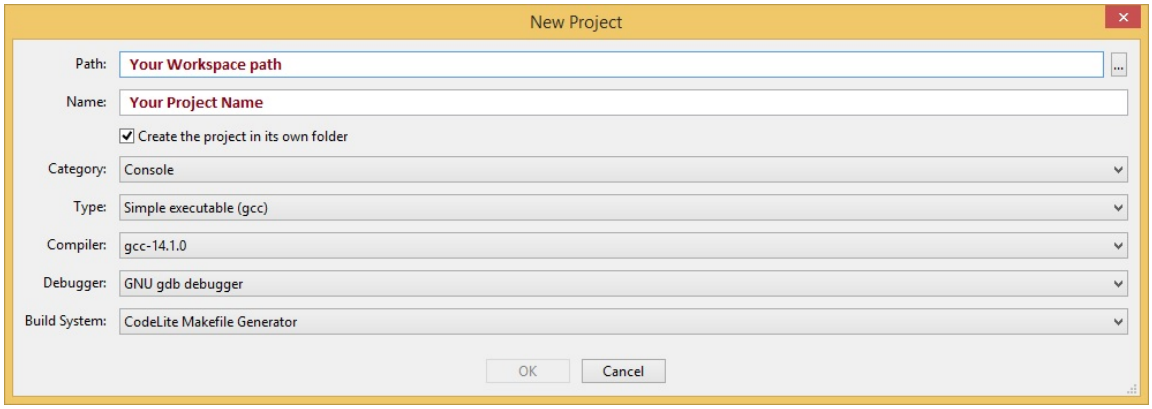


شکل ۱ - ۱

[2] Integrated Development Environment

۴. افزودن Workspace با نام و مسیر دلخواه

۵. افزودن پروژه جدید (New Project) به Workspace تعریف شده و اعمال تنظیمات مشابه شکل ۱-۲.



شکل ۱-۲

۱-۳ دستور کار

موارد زیر را انجام داده و در گزارش کار خود بیاورید:

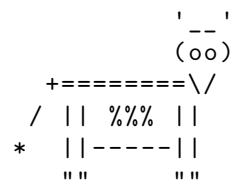
۱. برنامه‌ی Hello World! را نوشته، اجرا و اجزای آن را بررسی کنید.

```c
// First program in C.
#include <stdio.h>

// Program execution starts from the 'main' function
int main(void)
{
   printf("Let's C!\n");
   return 0; // EXIT_SUCCESS
}
```

۲. خروجی برنامه زیر را حدس بزنید!

```c
#include <stdio.h>

int main()
{
   long long value = 4 * 3 * 5 * 157 * 993893936993;
   printf("%s", &value);
   return 0;
}
```

۳. برنامه‌ای بنویسید که تصویر ۱-۳ را چاپ کند. توجه داشته باشید که برای چاپ کاراکترهای خاص لازم است تا از کاراکتر گریز استفاده شود.



شکل ۱-۳

آزمایش ۲
آشنایی با انواع داده‌ای اصلی و دستورات ورودی/خروجی

۲-۱ اهداف

۱. آشنایی با انواع داده‌ای اصلی زبان C

۲. آشنایی با اندازه و نحوه ذخیره انواع داده‌ای

۳. آشنایی با نحوه تبدیل نوع

۴. استفاده از دستورات ورودی و خروجی ساده برای خواندن و نوشتن اطلاعات

۵. انجام ورودی و خروجی به صورت قالب‌بندی شده

۲-۲ توضیحات

زبان C از تعداد محدودی نوع داده‌ای اصلی پشتیبانی می‌کند. جدول ۱.۲ لیستی از این انواع داده‌ای به همراه اندازه، کاربرد و اولین نسخه ارائه کننده آن نوع داده‌ای را نشان می‌دهد. علاوه بر انواع داده‌ای اولیه (جدول ۱.۲) با اعمال توصیف‌گرهای short یا long و signed یا unsigned بر روی انواع عددی می‌توان انواع مختلف با ویژگی‌های متفاوت داشت. جدول ۲.۲ لیست کامل‌تری از انواع داده‌ای زبان C را به همراه توضیحات، اندازه، کمترین مقدار ممکن و بیشترین مقدار ممکن (با توجه به ترتیب عددی) نمایش می‌دهد. توجه داشته باشید که در انواع مربوط به اعداد اعشاری کمترین مقدار بیانگر کمترین مقدار مثبت پشتیبانی شده توسط آن نوع می‌باشد. ضمن اینکه با توجه به محدودیت فضا برای کمترین و بیشترین مقدار، مقادیر تقریبی به جای مقادیر دقیق استفاده شده‌اند.

نوع داده‌ای	توضیح	اندازه (بایت)	نسخه معرفی شده
char	نوع کاراکتر	1	K&R
int	نوع اعداد صحیح	4	K&R
float	نوع اعداد اعشاری	4	K&R
double	نوع اعداد اعشاری با دقت مضاعف	8	K&R
_Bool	نوع بولی	1	C99
_Complex	نوع اعداد مختلط	8	C99
_Imaginary	نوع اعداد موهومی	8	C99

جدول ۱.۲: انواع داده‌ای اصلی در زبان C

نوع داده‌ای همراه با توصیف‌گر[1]	شرح	اندازه (بایت)	کمترین مقدار	بیشترین مقدار
char	کاراکتر (وابسته به پیاده‌سازی می‌تواند signed و یا unsigned باشد.)	1	-128 (0)	127 (255)
signed char	کاراکتر یا عدد صحیح یک بایتی علامت‌دار	1	-128	127
unsigned char	کاراکتر یا عدد صحیح یک بایتی بدون علامت	1	0	255
short short int signed short signed short int	عدد صحیح کوتاه علامت‌دار (حداقل ۱۶ بیتی)	2	$-32,768$	32,767
unsigned short unsigned short int	عدد صحیح کوتاه بدون علامت (حداقل ۱۶ بیتی)	2	0	65,536
int signed int	عدد صحیح علامت‌دار (حداقل ۱۶ بیتی)	4 (2)	$-2,147,483,648$ ($-32,768$)	2,147,483,647 (32,767)
unsigned int	عدد صحیح بدون علامت (حداقل ۱۶ بیتی)	4 (2)	0 (0)	4,294,967,295 (65,536)
long long int signed long signed long int	عدد صحیح طویل علامت‌دار (حداقل ۳۲ بیتی)	4 (8)	$-2,147,483,648$ ($-9,223,372,036,854,775,808$)	2,147,483,647 (9,223,372,036,854,775,807)
unsigned long unsigned long int	عدد صحیح طویل بدون علامت (حداقل ۳۲ بیتی)	4 (8)	0 (0)	4,294,967,295 (65,536)
long long long long int signed long long signed long long int	عدد صحیح بسیار طویل علامت‌دار (حداقل ۶۴ بیتی، استاندارد C99 به بعد)	8	$-9,223,372,036,854,775,808$	9,223,372,036,854,775,807
unsigned long long unsigned long long int	عدد صحیح بسیار طویل بدون علامت (حداقل ۶۴ بیتی، استاندارد C99 به بعد)	8	0	18,446,744,073,709,551,615
float	اعداد اعشاری با دقت یک برابر (استاندارد IEEE 754)	4	$1.175e$-38 (کوچکترین عدد مثبت)	$3.403e38$
double	اعداد اعشاری با دقت دو برابر (استاندارد IEEE 754)	8	$2.225e$-308 (کوچکترین عدد مثبت)	$1.797e308$
long double	اعداد اعشاری با دقت چهار برابر (حداقل ۶۴ بیتی، استاندارد IEEE 754)	16 12 10 8	$3.362e-4932$ $3.362e-4932$ $3.362e-4932$ $2.225e-308$	$1.1897e4932$ $1.1897e4932$ $1.1897e4932$ $1.797e308$
`_Bool`	نوع بولی (استاندارد C99 به بعد)	1	0	1
`_Complex`	اعداد مختلط (استاندارد C99 به بعد)	8	-	-
`_Imaginary`	اعداد موهومی (استاندارد C99 به بعد)	8	-	-

جدول ۲.۲: انواع مختلف داده‌ای در کنار توصیف‌گرها در زبان C

ورودی و خروجی قالب‌بندی شده

دو مورد از اصلی‌ترین اعمال در زبان C خواندن مقادیر ورودی از *دستگاه ورودی استاندارد* (کیبورد) و چاپ داده‌های تولید شده توسط برنامه در *دستگاه خروجی استاندارد* (مانیتور) می‌باشد. در آزمایش اول دیدیم که چگونه با استفاده از توابع printf و scanf به ترتیب می‌توان خروجی را نوشت و ورودی را دریافت نمود. در این آزمایش با جزئیات بیشتری از این دو تابع آشنا می‌شویم.

کاراکتر f در انتهای نام این دو تابع سرواژه کلمه formatted به معنی قالب‌بندی شده است. به عبارتی این دو تابع خروجی و ورودی را به صورت قالب‌بندی شده چاپ و یا دریافت می‌نمایند.

تابع printf

```
printf("formatted string", variable list);
```

در تابع printf، رشته‌ی قالب‌بندی شده می‌تواند شامل صفر یا تعداد بیشتری مشخص کننده قالب باشد. هر مشخص کننده با کاراکتر % شروع شده و ساختار کلی زیر را دارد:

مشخص کننده نوع [توصیف‌گر طول] [دقت.] [عرض] [پرچم‌ها]%

تابع scanf

```
scanf("formatted string", arg1, arg2, ..., argn);
```

در تابع scanf نیز مشابه تابع printf، رشته‌ی قالب‌بندی شده می‌تواند شامل صفر (بدون خواندن از ورودی) یا تعداد بیشتری مشخص کننده قالب با ساختار کلی زیر باشد:

مشخص کننده نوع [توصیف‌گر طول] [عرض] [*]%

موارد مشخص شده داخل کروشه اختیاری بوده و در کلاس توضیح داده شده‌اند. در اینجا برای یادآوری و حل تمرینات به بررسی مورد الزامی «مشخص کننده نوع» در کنار توصیف‌گر طول می‌پردازیم. توجه داشته باشید که توصیف‌گرهای طول h، l و L به ترتیب برای short int، long int و long double استفاده می‌شوند. در جدول ۳.۲ توصیف‌گر طول به عنوان بخشی از مشخص‌کننده نوع در نظر گرفته شده است.

۲-۳ دستور کار

موارد زیر را انجام داده و در گزارش کار خود بیاورید:

۱. برنامه زیر را پیاده‌سازی نموده و خروجی چاپ شده را تحلیل کنید.

```c
#include <stdio.h>

int main(void)
{
   long double start;
   long double ld = 1.23L;
   double d = 1.23;
   long long ll = 123LL;
   long l = 123L;
   float f = 1.23;
   int i = 123;
   short s = 123;
   char c = 'a';

   // Printing addresses of consecutive variables
   printf("first address is %p\n", &start);
```

```c
    printf("long double ld's address is %p\n", &ld);
    printf("double d's address is %p\n", &d);
    printf("long long int ll's address is %p\n", &ll);
    printf("long int l's address is %p\n", &l);
    printf("float f's address is %p\n", &f);
    printf("int i's address is %p\n", &i);
    printf("short int s's address is %p\n", &s);
    printf("character c's address is %p\n", &c);

    return 0;
}
```

۲. برنامه‌ای بنویسید که با استفاده از توابع scanf و printf، مقادیری با انواع مختلف داده‌ای را از ورودی خوانده و با توضیحات چاپ نماید.

۳. با توجه به نحوه نمایش و ذخیره‌سازی اعداد صحیح (2's Complement) و اعشاری (IEEE 754) سعی کنید با استفاده از پیشوندهای 0x (Hexadecimal) یا 0 (Octal) کمترین و بیشترین مقادیر ممکن برای انواع مختلف داده‌ای را در متغیرهای مختلف تعریف نموده و چاپ نمایید.

نوع کلی	مشخص کننده نوع	نوع و فرمت
عدد صحیح	%d or %i	(signed) int
	%u	unsigned int
	%o	unsigned int in octal
	%x, %X	unsigned int in hexadecimal
	%hd or %hi, %hu	(signed) short, unsigned short
	%ld or %li, %lu	(signed) long, unsigned long
	%lld or %lli, %llu	(signed) long long, unsigned long long
عدد اعشاری	%f	float in fixed notation (i.e. 123.449997)
	%e, %E	float in scientific notation (i.e. 1.234500e+02)
	%g, %G	float in shortest representation: %e/%E or %f
	%lf ،%f	double (only %lf for scanf)
	%LG ،%Lg ،%LE ،%Le ،%Lf	long double
کاراکتر	%c	char
رشته	%s	string of characters
-	%%	percent sign

جدول ۳.۲: مشخص کننده‌های نوع در تابع printf

آزمایش ۳
عملگرها

۳-۱ اهداف

۱. بررسی عملگرهای مختلف تعریف شده در زبان C

۲. انجام محاسبات با استفاده از اعمال عملگرهای مختلف بر روی متغیرها و ثابت‌ها

۳. آشنایی با تقدم و اولویت عمل‌گرها

۳-۲ دستور کار

موارد زیر را انجام داده و در گزارش کار خود بیاورید:

۱. برنامه‌ای بنویسید که دو عدد صحیح a و b را از ورودی دریافت کرده و حاصل اعمال $+$، $-$، $*$، $/$، $\%$، $<$، $>$، $=$! و $==$ روی آن‌ها ($a+b$، $a-b$، $a*b$، a/b، $a\%b$، $a<b$، $a>b$، $a!=b$ و $a==b$) را چاپ کند.

۲. برنامه‌ای بنویسید که سه عدد صحیح مثبت را به عنوان طول اضلاع مثلث از ورودی دریافت نموده و با استفاده از فرمول Heron مساحت مثلث را محاسبه و چاپ کند. سعی کنید برای حالت‌های نادرست ورودی پیغام خطای مناسب را نشان دهید.

$$s = \frac{(a+b+c)}{2}$$

$$Area = \sqrt{s*(s-a)*(s-b)*(s-c)}$$

۳. خروجی برنامه‌ی زیر را تحلیل کنید.

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

int main() {
   // Range of int is [INT_MIN, INT_MAX]
   int a = INT_MAX;      // max int defined in limits.h
   int b = INT_MIN;      // min int defined in limits.h

   printf("%5s: %d\n","a", a);
   printf("%5s: %d\n","a + 1", a + 1);
   printf("%5s: %d\n", "a + 2", a + 2);
   printf("%5s: %d\n", "a * a", a * a);

   printf("%5s: %d\n", "b", b);
   printf("%5s: %d\n", "b - 1", b - 1);
   printf("%5s: %d\n", "b - 2", b - 2);
   printf("%5s: %d\n", "b * b", b * b);
   return 0;
```

```
19 }
```

۴. برنامه‌ای بنویسید که به نوبت سه عدد از ورودی دریافت کرده و سپس اعداد مربوطه را از کوچک به بزرگ چاپ کند.

آزمایش ۴
دستورات شرطی و تصمیم‌گیری

۴-۱ اهداف

۱. آشنایی و استفاده از دستورات شرطی و کنترلی زبان C

۲. تصمیم‌گیری و انتخاب مسیر با توجه به مقدار و نوع ورودی

۳. استفاده از دستورات شرطی به صورت تو در تو

۴-۲ دستور کار

موارد زیر را انجام داده و در گزارش کار خود بیاورید:

۱. برنامه‌ای بنویسید که درجه حرارت را به همراه واحد (مثلا 55.75C یا 132.35F) از کاربر دریافت نموده و به واحد دیگر تبدیل و چاپ نماید. توجه داشته باشید که محاسبات و نمایش آن با دقت مناسب انجام شود.

$$^\circ F = \frac{9}{5} * {}^\circ C + 32$$

$$^\circ C = \frac{5}{9} * (^\circ F - 32)$$

۲. برنامه‌ای بنویسید که قد و وزن کاربر را بر حسب متر و کیلوگرم دریافت نموده و با استفاده از فرمول و جدول زیر وضعیت وزنی او را چاپ کند.

$$bmi = \frac{\text{وزن } (kg)}{\text{(قد } (m))^2}$$

bmi	وضعیت
کمتر از ۱۶	کمبود وزن شدید
از ۱۶ تا ۵.۱۸	کمبود وزن
از ۵.۱۸ تا ۲۵	عادی
از ۲۵ تا ۳۰	اضافه وزن
از ۳۰ تا ۳۵	چاقی کلاس ۱
از ۳۵ تا ۴۰	چاقی کلاس ۲
۴۰ و بیش از ۴۰	چاقی کلاس ۳

۳. برنامه‌ای بنویسید که یک عبارت به صورت $a \ [+ - * / \%] \ b$ از ورودی دریافت نموده و حاصل آن را مقابلش چاپ کند.

۴. *برنامه‌ای بنویسید که تاریخ میلادی را در فرمت $d/m/yyyy$ از ورودی دریافت نموده و آن را به صورت گسترده‌تر (با مشخص کردن ماه انگلیسی) با فرمت $mmmm\ d, yyyy$ چاپ کند (شکل ۵-۱ خط ۲). تکه کد زیر را بررسی و آن را به صورت خواناتری در برنامه خود بازنویسی کنید. آیا می‌توانید روز هفته را نیز نمایش دهید (شکل ۵-۱ خط ۳).

```
int weekday  = (d += m < 3 ? y-- : y - 2, 23*m/9 + d + 4 + y/4- y/100 +
↪  y/400)%7;
```

```
Enter the date as d/m/yyyy: 12/10/2011
October 12, 2011
October 12 (Wednesday), 2011
```

شکل ۴-۱

آزمایش ۵
دستورات تکرار و حلقه

۵-۱ اهداف

۱. آشنایی با انواع مختلف دستورات تکرار

۲. استفاده از دستورات تکرار و حلقه به صورت تو در تو

۳. آشنایی و استفاده از دستورات `break` و `continue` برای تغییر و کنترل مسیر اجرا

۵-۲ دستور کار

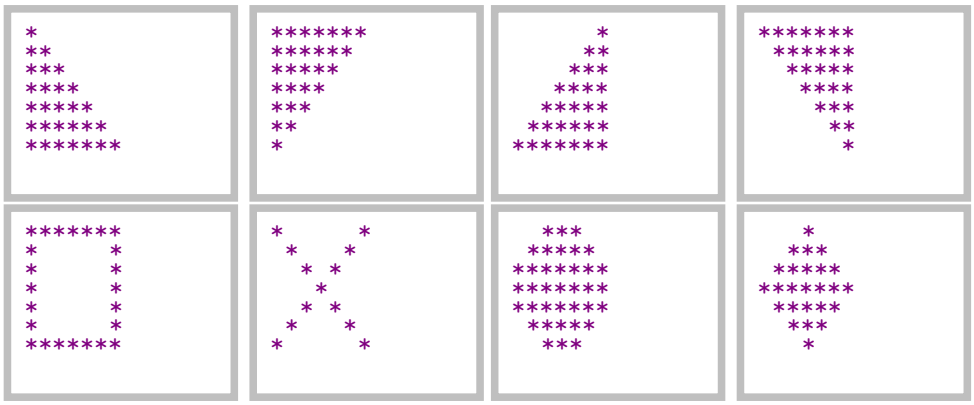
موارد زیر را انجام داده و در گزارش کار خود بیاورید:

۱. برنامه‌ای بنویسید که عددی صحیح را از ورودی خوانده و مجموع ارقام آن را چاپ کند.

۲. برنامه‌ای بنویسید که یک عدد صحیح x در مبنای 10 و مبنای مقصد b ($2 \leq b \leq 16$) را دریافت نموده و عدد مربوطه را در مبنای جدید چاپ کند.

۳. برنامه‌ای بنویسید که عدد n را از ورودی دریافت کرده و عدد nام فیبوناچی را چاپ کند.

۴. برنامه‌هایی بنویسید که عددی را از ورودی دریافت کرده (به عنوان مثال 7) و خروجی‌های زیر را تولید کنند (شکل ۵-۱).



شکل ۵-۱

۵. برنامه‌ای بنویسید که تا دریافت عدد -1 اعداد صحیح غیرمنفی را از کاربر دریافت کرده و در پایان کوچکترین و بزرگترین عدد وارد شده به همراه میانگین کل اعداد را چاپ کند. امکان محاسبه‌ی میانه اعداد را نیز بررسی کنید.

```
1
11
121
1331
14641
```

شکل ۵-۲

۶. برنامه‌ای بنویسید که ۵ سطر اول مثلث خیام پاسکال را چاپ کند (شکل ۵-۲).

۷. برنامه‌ای بنویسید که عدد n را از ورودی دریافت کرده و کوچکترین n عدد اول را چاپ کند.

آزمایش ۶
توابع

۶-۱ اهداف

۱. یادگیری مفهوم و مزایای برنامه‌نویسی ماجولار

۲. یادگیری نحوه تعریف، پیاده‌سازی و فراخوانی توابع

۳. آشنایی با روش‌های فراخوانی call by value و call by reference

۴. آشنایی با توابع بازگشتی و مفهوم tail recursion

۵. آشنایی با برخی از توابع ریاضی موجود در کتابخانه استاندارد زبان C

۶-۲ دستور کار

موارد زیر را انجام داده و در گزارش کار خود بیاورید:

۱. دو تابع برای محاسبه‌ی ${}^nP_r$ (جایگشت r از n) و ${}^nC_r$ (ترکیب r از n) نوشته و برای مقادیر مختلف $0 \leq n \leq 10$ و $0 \leq r \leq n$ حاصل فراخوانی این توابع را نمایش دهد. آیا می‌توان با افزودن تابعی دیگر محاسبات فوق را ساده‌تر نمود؟

۲. تابعی به نام `is_prime` را پیاده‌سازی کنید که یک عدد صحیح دریافت نموده و اول بودن یا نبودن آن را برگرداند. سپس با استفاده از این تابع اعداد اول بین 500 تا 1000 را چاپ کنید.

۳. تابع `gcd` برای محاسبه‌ی ب.م.م (بزرگترین مقسوم علیه مشترک) را با توجه به الگوریتم اقلیدس به صورت بازگشتی و غیربازگشتی پیاده‌سازی کنید.

۴. تابعی بنویسید (`void swap(int *a, int *b)`) که دو عدد؟ را به صورت آرگومان دریافت کرده و مقادیر این دو عدد را با یکدیگر جابجا کند. درستی پیاده‌سازی خود را با یک برنامه ساده بررسی کنید.

۵. فایلی با نام `my_math.h` ایجاد نموده و توابعی نظیر `fmod`، `abs`، `floor`، `ceil`، `round`، `sine`، `cosine`، `pow`، `log`، `exp` و غیره را درون این فایل پیاده‌سازی نمایید (در صورت نیاز با تقریب مناسب). سپس در یک فایل جداگانه برنامه‌ای نوشته و توابع پیاده‌سازی شده را با معادل خود در `math.h` مقایسه نمایید.

آزمایش ۷
آرایه‌ها

۱-۷ اهداف

۱. آشنایی با داده‌ساختار آرایه برای ذخیره و نمایش دسته‌ای از مقادیر با نوع یکسان

۲. نحوه‌ی تعریف و مقداری دهی اولیه آرایه‌ها

۳. نحوه تعریف و استفاده از آرایه‌ها با ابعاد بزرگتر

۴. نحوه‌ای ارسال و استفاده از آرایه‌ها در توابع

۵. آشنایی و استفاده از آرایه‌ها با طول متغیر

۲-۷ توضیحات

۳-۷ دستور کار

موارد زیر را انجام داده و در گزارش کار خود بیاورید:

۱. برنامه‌ای بنویسید که ۷ عدد از ورودی دریافت کرده و این اعداد را از انتها به ابتدا چاپ کند.

۲. عملکرد تابع زیر را بر روی آرایه `int arr[] = { 10, 3, 13, 5, 1 };` بررسی و تحلیل کنید. توجه داشته باشید که `n` اندازه آرایه بوده و می‌توان مقدار آن را با استفاده از `n = sizeof(arr) / sizeof(arr[0]);` قبل از فراخوانی تابع محاسبه کرد. بررسی کنید که چرا استفاده از `sizeof(arr) / sizeof(arr[0])` در داخل تابع مقدار متفاوتی را بدست می‌دهد!؟ آیا می‌توانید تابعی دیگر با عملکرد نهایی مشابه تابع `foo` ارائه دهید؟

```c
void foo(int arr[], int n)
{
    for (int i = 1; i < n; i++) {
        int key = arr[i];
        int j = i - 1;

        while (j >= 0 && arr[j] > key) {
            arr[j + 1] = arr[j];
            j = j - 1;
        }
        arr[j + 1] = key;
    }
}
```

۳. برنامه‌ای بنویسید که ابتدا تعداد اعداد و سپس اعداد را از ورودی دریافت کرده و سپس بررسی کند که آیا عدد تکراری وجود دارد یا خیر؟ اگر تعداد اعداد داده نشود و اعداد را تا دریافت عدد -1 از ورودی بخوانیم راهکار شما چیست؟ در

صورتی که اعداد وارد شده بین ۰ تا ۱۰۰ باشد کد خود را به نحوی تغییر دهید که اعداد و تکرار (فراوانی) آن را نمایش دهد.

۴. برنامه‌ای بنویسید که عدد صحیح n را از ورودی خوانده و مثلث خیام پاسکال را تا سطر n محاسبه و چاپ کند.

۵. برنامه‌ای بنویسید که دو ماتریس 3×4 را از ورودی دریافت کرده و ترانهاده مجموع آن‌ها را چاپ کند.

۶. برنامه‌ای بنویسید که ۵ نقطه مختصات را از کاربر به صورت (x, y) دریافت نموده و در نهایت دورترین دو نقطه به یکدیگر و فاصله‌ی آن‌ها را چاپ کند.

آزمایش ۸
اشاره‌گرها

۱-۸ اهداف

۱. آشنایی با اشاره‌گرها و مفاهیم آن

۲. آشنایی و استفاده از عملیات جبری بر روی اشاره‌گرها

۳. آشنایی با روش‌های تخصیص حافظه به صورت پویا و در زمان اجرا

۴. آشنایی با انواع توابع تخصیص، ست، کپی و غیره

۵. آشنایی با نحوه تعریف اشاره‌گر به تابع، اشاره‌گر به آرایه و استفاده از آن به عنوان آرگومان در فراخوانی توابع

۶. استفاده از نوع void و *void و مفهوم آن در بحث اشاره‌گرها

۲-۸ دستور کار

موارد زیر را انجام داده و در گزارش کار خود بیاورید:

۱. خروجی کد زیر را حدس بزنید.

```c
#include <stdio.h>

int main(void)
{
   int num1=1, num2=2, *pnum1, *pnum2;
   pnum1 = &num1;
   pnum2 = pnum1;
   *pnum2++;
   *pnum1 = 10;
   pnum2 = &num1;
   *pnum2++;
   pnum1 = pnum2 - 1;
   printf("num1: %d\nnum2: %d\n*pnum1: %d\n*pnum2: %d", num1, num2, *pnum1,
   ↪  *pnum2);
   return 0;
}
```

۲. برنامه‌ای بنویسید که اعداد صحیح غیرمنفی را تا دریافت عدد -1، از کاربر دریافت کند. سپس میانه‌ی اعداد را یافته و چاپ کند. سعی کنید در حد امکان اتلاف حافظه کمی داشته باشید.

۳. برنامه‌ای بنویسید که ابتدا ابعاد m و n ماتریس و سپس درایه‌های آن را از ورودی خوانده و در نهایت ماتریس مربوطه را چاپ کند.

۴. تابعی را پیاده‌سازی کنید که دو آرایه را دریافت کرده و این دو آرایه را با یکدیگر جابجا کند.

۵. تابع qsort با امضای زیر از کتابخانه stdlib.h الگوریتم Quicksort؟ را پیاده‌سازی می‌کند.

```
void qsort(void *base, size_t num, size_t size, int (*comparator)(const void *,
↪  const void*))
```

پارامترهای استفاده شده در تابع و نوع آن‌ها را تحلیل نموده و بخش خالی کد زیر برای مرتب‌سازی آرایه‌ای از اعداد صحیح را تکمیل کنید.

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

// Comparison function
// Should return
//     <0 if the element pointed by a goes before the element pointed by b
//      0 if element pointed by a is equivalent to the element pointed by b
//     >0 if the element pointed by a goes after the element pointed by b
int compare(const void* a, const void* b) {



}

int main() {
   int arr[] = {10, 3, 13, 5, 1};
   int n = sizeof(arr) / sizeof(arr[0]);

   qsort(arr, n, sizeof(int), compare);

   printf("Sorted array:\n");
   int i;
   for (i = 0; i < n; i++) {
      printf("%d ", arr[i]);
   }
   printf("\n");
   return 0;
}
```

۶. کد بالا را به گونه‌ای تغییر دهید که امکان مرتب‌سازی صعودی یا نزولی وجود داشته باشد. برای این کار از کاربر بخواهید تا از بین دو گزینه (۰ برای مرتب‌سازی صعودی و ۱ برای مرتب‌سازی نزولی) یکی را انتخاب کند و سپس با توجه به انتخاب کاربر نتیجه مرتب‌سازی را نمایش دهید. سعی کنید فقط یک فراخوانی تابع qsort داشته باشید. (از آرایه‌ای از اشاره‌گرها به توابع استفاده کنید.)

آزمایش ۹
رشته‌ها

۹-۱ اهداف

۱. آشنایی با نحوه‌ی تعریف و استفاده از رشته‌ها

۲. آشنایی با نحوه‌ی ذخیره شدن رشته‌ها و کاراکترها در حافظه

۳. آشنایی با توابع مختلف موجود برای دستکاری رشته‌ها در کتابخانه‌های `ctype.h`، `string.h` و `stdlib.h`

۹-۲ دستور کار

موارد زیر را انجام داده و در گزارش کار خود بیاورید:

۱. برنامه‌ای بنویسید که یک رشته را از کاربر دریافت نموده و حروف آن را به حروف بزرگ تغییر دهد.

۲. برنامه‌ای بنویسید که دو رشته را از کاربر دریافت و به همدیگر بچسباند.

۳. تابع `substring` را به گونه‌ای پیاده‌سازی کنید که یک رشته، اندیس شروع و طول رشته را دریافت نموده و زیررشته‌ی مربوطه را برگرداند.

۴. تابع `replace` را به گونه‌ای پیاده‌سازی کنید که یک رشته‌ی ورودی، یک رشته‌ی جستجو و یک رشته‌ی جایگذاری را دریافت نموده و تمامی موارد رشته‌ی جستجو را با رشته‌ی جدید جایگذاری کند.

۵. برنامه‌ای بنویسید که حداقل از ۱۰ تابع دستکاری رشته‌ها که در کتابخانه‌های `ctype.h`، `string.h` و `stdlib.h` به درستی استفاده کرده باشد.

۶. برنامه‌ای بنویسید که دو عدد صحیح بسیار بزرگ را به صورت رشته دریافت کرده و چهار عمل اصلی ریاضی را روی آن‌ها اعمال و چاپ کند. توضیح دهید که چگونه می‌توان عملیات دیگر (مانند انواع مقایسه) را پیاده‌سازی نمود.

آزمایش ۱۰
پردازش فایل

۱۰-۱ اهداف

۱. یادگیری مفاهیم مربوط به فایل‌ها و جریان‌ها

۲. خواندن و نوشتن خطی فایل‌های متنی

۳. خواندن و نوشتن فایل‌ها به صورت دسترسی تصادفی

۱۰-۲ دستور کار

موارد زیر را انجام داده و در گزارش کار خود بیاورید:

۱. برنامه‌ای بنویسید که یک رشته دریافتی را در یک فایل جستجو کند. برنامه می‌بایست تمامی موارد تکرار رشته درون فایل را با ذکر سطر و ستون چاپ کند.

۲. برنامه‌ای بنویسید که دو فایل را ادغام نموده و در یک فایل مقصد بنویسد.

۳. برنامه‌ای بنویسید که یک فایل متنی بزرگ را به صورت تکه تکه نمایش داده و با دریافت کلیدهای خاص بر روی متن حرکت کند (بصورت سطر به سطر یا صفحه به صفحه).

۴. برنامه‌ای بنویسید که یک فایل را خوانده و اطلاعات آماری نظیر تعداد کل کلمات، تعداد کلمات غیر تکراری، تعداد حروف، تعداد اعداد، تعداد جملات، کلمه با بیشترین تکرار و کلمه با کمترین تکرار را چاپ کند.

۵. برنامه‌ای بنویسید که یک دستور فایلی (مانند دستورات کپی، حذف، تغییر نام و ...) را به همراه پارامترهای لازم توسط تابع `main` دریافت نموده و در صورت درست بودن دستور آن را اعمال کند و در صورت نادرست بودن آن، کاربر را راهنمایی کند.

۶. برنامه‌ای بنویسید که اطلاعات (نام، نام خانوادگی و موجودی حساب) رکوردهای مربوط به چند مشتری را از کاربر دریافت نموده و به صورت متنی و دودویی در یک فایل ذخیره کرده و سپس بخواند.

۷. برنامه‌ای بنویسید که در ادامه‌ی برنامه‌ی بالا، یکی از اعمال insert، update، remove و find را در قالب یک منو و سپس شماره حساب را از کاربر دریافت نموده و با دسترسی تصادفی عملیات خواسته شده را روی رکورد مربوطه اعمال کند.

```
typedef struct client {
   int account;
   char lastName[25];
   char firstName[15];
   double balance;
};
```