ШАГ компьютерная академия

ПРОГРАММИРОВАНИЕ НА ЯЗЫКЕ С

Урок №1

C

Dilində proqramlaşdırma

Mündəricat

1. İLKİN MÜLAHİZƏLƏR

Giriş sözü.

Proqramlaşdırma aləminə xoş gəlmisiniz! Proqramlaşdırma nədir? Yəqin ki, hər biriniz nə vaxtsa bu sözü eşitmisiniz. Məşhur sitatı dəyişərək belə demək olar: “Proqramçı - qürurlu səslənir!”. Və bu həqiqətən belədir. Əgər siz İnternet axtarış apardıqda iş təklif edən saytlarla rastlaşmısınızsa, o zaman orada proqramçılara təklif olunan əmək haqqının miqdarına fikir vermiş olarsınız. Təbii ki, belə bir sual yaranır: Proqramçının əmək haqqı niyə belə yüksəkdir?

Bu tam olaraq bazar qanunlarına əsaslanır: Proqramçılara tələbat vardır lakin hal-hazırda yaxşı mütəxəssislər çox azdır. Əlbəttə ki, qısa zaman ərzində bir sıra sözlər öyrənərək proqrammist olmaq mümkün olsaydı, o zaman bu peşə çətin ki, bu qədər populyarlıq qazanmış olardı. Ancaq, təəssüflənməyin! Biz, Sizin müəllimləriniz çalışacayıq ki, Sizə nəinki necə proqramlaşdırmaq haqqda danışaq, Sizi proqramlaşdıraraq yaşamağı öyrətmək üçün bizdən asılı olan hər şeyi edək. Belə ki, bizim birgə işimiz qarşılıqlı olmalıdır. İş, yalnız öz üzərinizdə olan daimi iş, Sizi ustadlığın zirvələrinə gətirə bilər.

Ürəkdən ümid edirik ki, Siz proqramlaşdırmanın sehirli dünyasıyla maraqlanacaqsınız.

İstənilən elmin öyrənilməsinə başlamazdan öncə, ilk növbədə bu elmin hardan yarandığını, necə inkşaf etdiyini, köklərini və tarixi vacibliyini öyrənmək lazımdır..

Tarixi faktlar.

XIX əsr İngiltərə riyaziyyatçısı Şenks həyatının 20 ilini Pİ ədədinin vergüldən sonra 707 rəqəm dəqiqliyi ilə tapılması üçün sərf etmişdir. Bu nəticə XIX əsrin hesablanma şöhrəti rekordunu qazanmışdır. Lakin sonralar məlum oldu ki, Şenks 520-ci rəqəmdə səhv buraxmışdır və buna görə də ardınca gələn rəqəmlər də düzgün hesablanmamışdı.

1804 - cü ildə fransalı ixtiraçı Jozef Mari Jakkar “proqramla-idarəedilən” toxucu dəzgahını yaratmışdır. Dəzgahın idarə edilməsi üçün bir-biriylə lent şəkildə birləşdirilmiş perfokartlardan istifadə olunurdu. Dəzgahın “oxuyan-qurğusunun” taxta tutacaqları perfokart üzərində dəliklərini yerləşmə qaydasından asılı olaraq düzgün naxışların alınması üçün hansı ipləri qaldırmağı, hansını aşağısalmağı müəyyən edirdi.

1890-cı ildə ABŞ-da ixtiraçı German Xollerit perfokartlarla idarə olunan elektromexaniki hesablayıcı maşın - tabulyatoru tərtib etmişdir. Bu ABŞ əhalisi barədə yazıları yerləşdirmək üçün cədvəllərin tərtib olunması üçün istifadə edilirdi. Sonda əsası Xollerit tərəfindən qoyulmuş tabulyator istehsal edən şirkət, İnternational Business Machines (İBM) korporasiyasına çevrilmişdir.

1936-cı ildə Kembric universitetinin ingiltərəli, 25 yaşlı tələbəsi Alan Tyüринq “hesablanan ədədlər barədə” məqalə çap etmişdir, bu məqalədə istənilən riyazi və məntiqi məsələlərin həlli üçün uyğun gələn hipotetik qurğu (Tyüринq maşını) - proqramlaşdırıla bilən kompyuter nümunəsinə - baxış keçirilmişdir.

1941-ci ildə alman mühəndisi Konrad Tsuze işlək - Z3 kompyuterini qurmuşdur, burada ikili hesablama sistemindən istifadə olunurdu. Proqramlar perfolentə yazılırdı.

1945 - ci ildə, ABŞ - ın Pensilvaniya ali texniki məktəbində fizik Con Moçli və mühəndis Prosper Ekert tamamilə elktron maşın - "Eniak"-ı qurmuşlar. Proqramın verilməsi üçün, əl ilə minlərlə keçiriciləri qoşmaq və kontakt panelinin yuvalarına yüzlərlə ştepselləri taxmaq lazım olurdu. 1945-ci ilin 1 İyununda macar mənşəli amerikalı riyaziyyatçı Con fon Neymanın "Edvak maşını barədə ilkin hesabat" adlı hesabatı yayılmışdır, bu hesabat komputer komandalarının onun öz daxili yaddaşında saxlanma konsepsiyasından ibarət idi. 1948-ci ilin 21 İyununda Mançester universitetində (İngiltərə) "Mark - 1" maşınında maşının yaddaşında saxlanılan dünyada ilk olan proqram edilmişdir - verilmiş ədədin ən böyük vuruqunun axtarışı.

1949 - cu ildə Moris Ulksin rəhbərliyi altında "Edsak" kompyuteri yaradılmışdır. "Edsakın" tərtibatçıları mnemonik işarələrin sistemini tədbiq etmişdirlər, burada hər bir komanda bir baş hərif ilə təsvir olunurdu və altproqramların yaddaşının müəyyən yerdə sazlanmasını avtomatlaşdırdı. Moris Uilks mnemonik sistemi və sistemlə yığılan altproqram kitabxanaları - yığıcı sistem (assembly system) adlandırdı - assembler sözü buradan götürülmüşdür. 1949 - cu ildə ABŞ, Filadelfiyada, Con Moçlinin rəhbərliyi altında "Qısa kod" - proqramlaşdırma dilininin ilk primitiv interpretatoru yaradılmışdır.

1951 - ci ildə Remington Rand şirkətində amerikalı proqramçı Qreys Xopper ilk translyasiya edən proqramı tərtib etmişdir. Xopper onu kompilyator adlandırmışdır (complier - yığıb düzəldən, komponovşik).

1957 - ci ildə Nyu - York şəhərində, Medison-avenünün 20-ci mərtəbəsində yerləşən,İBM şirkətinin qərargahında Fortran dili (FORmula TRANslation - formulların tərcüməsi) yaranmışdır. Tərtibatçı qrupuna 30 yaşlı riyaziyyatçı Con Bekus rəhbərlik edirdi. Fortran - yüksək dərəcəli “həqiqi” dillərin ilkidir.

1963 - cü ildə Basic proqramlaşdırma dili yaradılmışdır. Dilin banisi Dartmouth Kollecinin işçiləri, Con Kemeni və Tomas Kurt olmuşdur. Yaradıcıların rəhbərliyi altında dil, kollecin bir qrup tələbələri tərəfindən realizə edilmişdir. Dilin ilk dialekti Dartmouth BASIC adlanırdı.

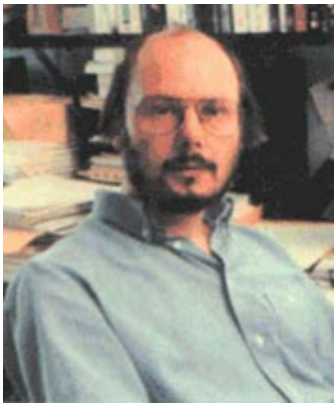
1958 - 1968 - ci illərdən bəri Algol adlı proqramlaşdırma dilinin təkmilləşdirilmə və tərtibat işləri aparılırdı, Algol adı (algorithmic language) ”alqoritmik dil”- söz birləşməsindən götürülmüşdür. Əsasən ABŞ və Kanadada istifadə olununa Fortrandan fərqli olaraq, Algol Avropa və SSR - də daha geniş yayılmışdı. Dil avropalı və amerikalı alimlər ibarət olan - Con Bekus, Piter Naur, Uolli Foyrtsoyq, Niklaus Virt olan beynəlxaq komitet tərəfindən yaradılmışdı.

1970 - ci ildə Niklaus Virt proqramlaşdırma dili yaratmışdır və onu fransız fiziki və riyaziyatçısı Blez Paskalın şərəfinə adlandırmışdır. Paskalı Virt prosedur proqramlaşdırmanı öyrədən dil kimi planlaşdırılmışdı.

1972 -ci ildə Bell Labs şirkətindən olan 31 yaşlı sistem proqramlaşdırma mütəxəssisi Denis Ritçı C proqaramlaşdırma dilini tərtib etmişdir. Dilin ilk təsviri rus dilinə tərcümə edilmiş B.Kernigan və D.Ritçinin kitabında verilmişdir. Uzun müddət bu təsvir standart idi, bir sıra momentlər birmənalı şərhlər vermirdi, bu da C dilinin bir çox izahlarının yaranmasına gətirib çıxardı. Bu vəziyyətin düzəlməsi üçün Amerika Dövlət Standart Universiteti (ANSİ) nəzdində C dilinin standartlaşdırma komiteti yaradılmışdı.

1995-ci ilin 23 Mayında Sun Microsystems şirkəti Oak adlı yeni proqramlaşdırma dilini buraxmışdır. Dil məişət elektronikasının proqramlaşdırılması üçün tərtib olunmuşdu. Nəticədə dil Java adını almış, əlavələrin və server proqram təminatının tərtibatında geniş istifadə olunmağa başlamışdır.

2000-2001 - ci illərdə xüsusi olaraq .NET platformaları üçün tərtib olunmuş, yeni proqramlaşdırma dili - C# (si -şarp) dili qəbul olunmuş və standartlaşdırılmışdır.



Dilin yaradılmasında Microsoft Research (Microsoft nəzdinə Elmi Tədqiqat İnstitutu) işçiləri - Anders Xeylsberq, Skott Viltamut, Piter Qolde və başqaları iştirak etmişdirlər. C# dilinin yeni versiyası 2005-ci ilin yazında təqdim olunmuşdur. C# dilinin əsasına qoyulmuş baza dillərdən biri də yenə də həmin köhnə C++ olmuşdur.

Beləcə, qısa tarixi faktlardan bizə aydın oldu ki, C və C++ proqramlaşdırma dilinin iki ayrıca dilidir, baxmayaraq ki, C++ C bazasında əsaslanmışdır. Bizim dərslər bu hər iki proqramlaşdırma dilin öyrənilməsinə həsr olunacaqdır. Əlbəttə ki, biliklərin alınması ardıcıl olacaqdır, əvvəlcə biz C diliylə tanış olacayıq, sonra isə ehmalca C++ keçəcəyik.

Keçmişə kiçik səyahətimizdən sonra gələcəyə keçə bilərik. Bizim sizinlə növbəti işimiz üçün sözsüz ki, bizə proqram təminatı lazım olacaqdır. İT - texnologiyalar bazarının sürətli inkşafıyla əlaqədar olaraq, müxtəlif proqramların daha yeni versiyaları yaradılır. Dərsin sonrakı bölümlərində biz, dərs prosesi çərçivəsində istifadə edəcəyimiz proqram produktunun installasiyasına baxış keçirəcəyik.

2. Microsoft Visual Studio 2013**-nun installasiyası**

Microsoft Visual Studio 2013 - Microsoft şirkətinin tələbələrin labarator işlərindən başlayaraq korporativ dərəcədə olan lahiyələrə qədər müxtəlif miqyaslı həllərin tərtibatı üçün tədbiq olunan proqram məhsulları dəstidir. Visual Studio-nun tərkibinə tərtbatın inteqrasiya edilmiş sahəsi (İDE - İntegrated development environment) çoxsaylı instrumental vasitələr və utillər daxildir. Visual Studionun köməyi ilə həm konsol əlavələri, həm də mürəkkəb qrafik interfeysə malik əlavələri yaratmaq olar. Biz dərsimiz çərçivəsində Visual Studio xətkeşindən olan produktları istifadə edəcəyik.

Visual Studio-nun tarixi 1997 - ci ildə başlamışdır. Məhz o zaman Visual Studio 97 buraxılmışdır. Siz proqramlaşdırmanı Microsoft Visual Studio 2013-dən istifadə edərək öyrənəcəksiniz. Hazırki anda, bu ən aktual versiyadır. Visual Studionun installasiyasına keçməzdən öncə, Visual Studio redaksiyasının (edition) məsələ həllinə baxaq. Məsələn: mobil həllərin və ya veb-həllərin, və ya iş stolunun həllinin və s. tərtibatı üçün.

Express Editions tamamilə pulsuz istifadə etmək olar, həmçinin kommersiya məqsədli tərtibatlar üçün də. Qeyd etmək lazımdır ki, Visual Studio müxtəlif redaksiyaları müxtəlif dillərdə mövcuddur. Misal üçün, istiyərsiz ki, rusdilli lokalizasiyanı qurasız. Bunu etməyin!

Gələcək professional işlərinizdə, digər ölkələrdən olan proekt və kolleqalarla rastlaşacaqsız, onlar yəqin ki, ingilis dilli interfeysdən istifadə edəcəklər. Məhz buna görə həmişə orijinal versiya dilini - ingilis dilini üstün tutun.

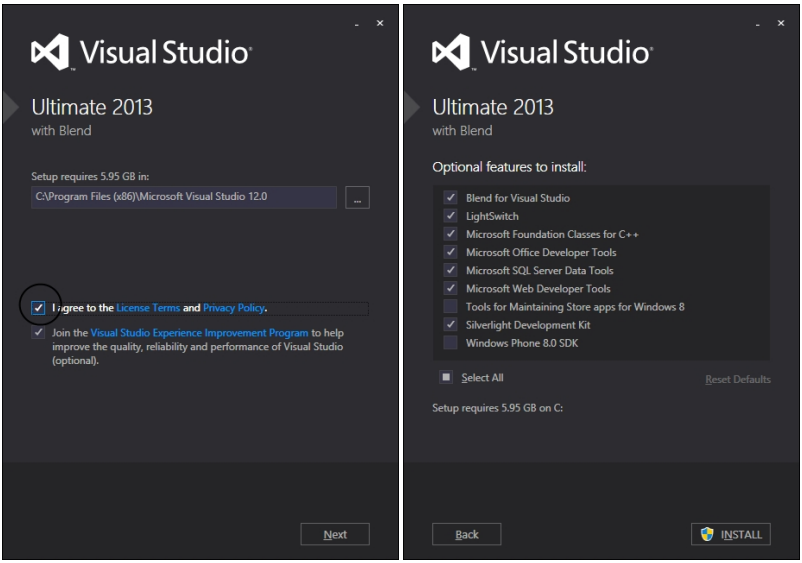
Biz Visual Studio quraşdırılmasına Microsoft Visual Studio 2013 Ultimate və Windows Desktop üçün Microsoft Studio Express 2013 nümunəsində baxaq.

Ultimate-dən başlayaq. Ultimate installasiyasına başlamaq üçün DVD diski DVD oxucusuna salın (və yaxud DVD diskin İSO - obrazını, misal üçün Daemon Tools Lite proqramının köməkliyi ilə yaradın) və vs_ultimate.exe faylını işə salın, bundan sonra sizdə belə bir pəncərə açılacaqdır:

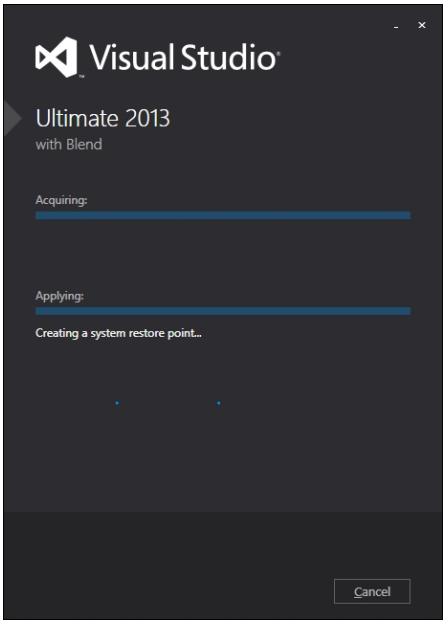
Bu pəncərədə Visual Studio-nun installasiya yolunu seçib, müəyyən edə bilərsiz ki, Visual Studio-nun təkmilləşdirilmə proqramında iştirak etmək istərdinizmi. Əgər siz keyfiyyətin təkmilləşdirilməsində iştirak etmək istəyirsinizsə, onda «Join the Visual Studio Experience Improvement Program …» opsiyasına quş qoyun.

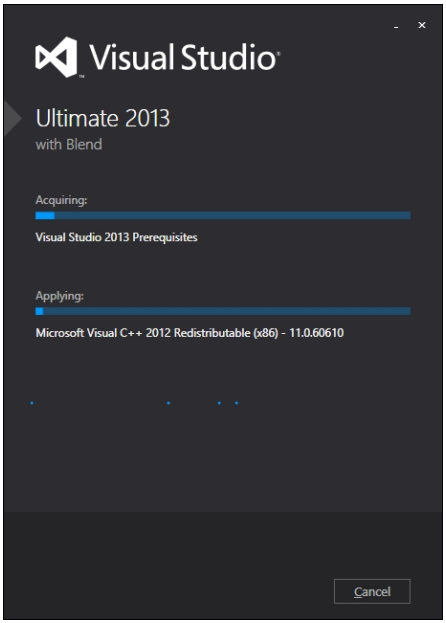
Quraşdırmaya başlamaq istəyirsinizsə «I agree to the License Terms and Privacy Policy» bölümünü seçin. Bu opsiyanı seçərək, siz təsdiqləyirsiz ki, bu lisenziyanın verilmiş bütün bölümləri ilə tanış və razı oldunuz.

Əgər siz düzgün seçim etmisizsə, pəncərənin aşağı hissəsində “Next” yazılmış düymə çıxır. Ona basın. Bundan sonra siz, quraşdırılma üçün bundan əlavə lazım olan proqram produktlarını seçməyə imkan verən pəncərə görəcəksiz. Bu mərhələdə cari seçilmiş proqramları saxlaya bilərsiniz. Gələcəkdə sonraya qalan komponentləri qurmaq imkanınız olacaq. Prosesin davam edilməsi üçün “İNSTALL” düyməsinə basın.



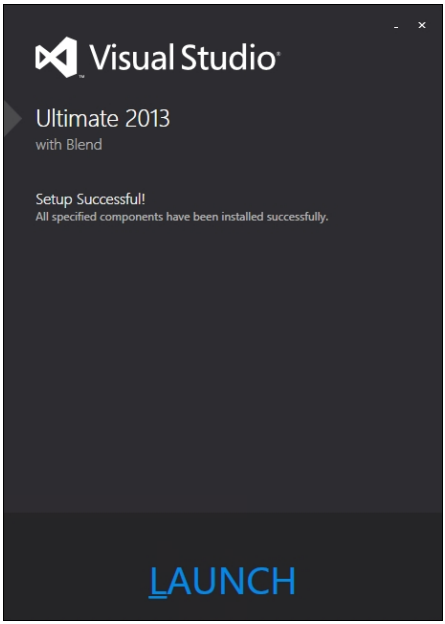
İnstallasiya prosesinə start verilməlidir. Bu bir qədər vaxt alacaq, bu vaxt ərzində işin proqressini müşahidə edə bilərsiniz. Aşağıda biz bu vəziyyətə xarakter olan pəncərələrin nümunəsini gətiririk.

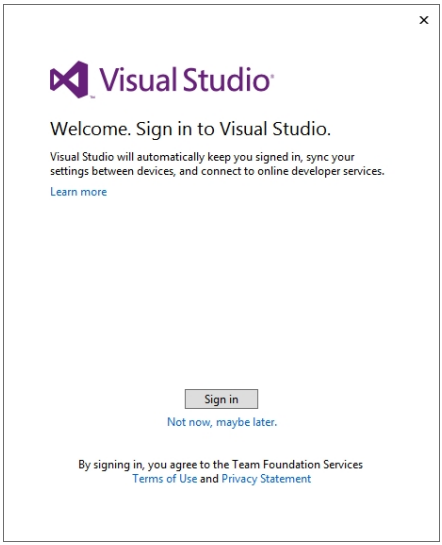


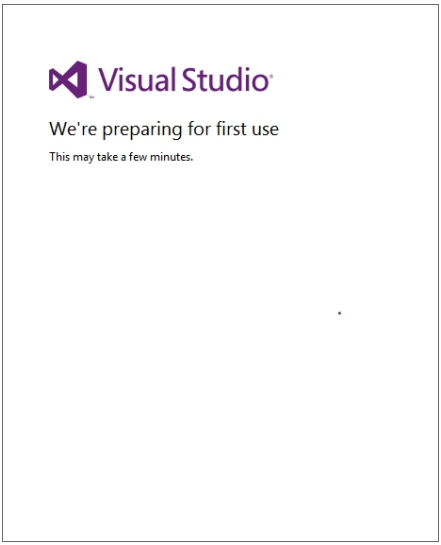


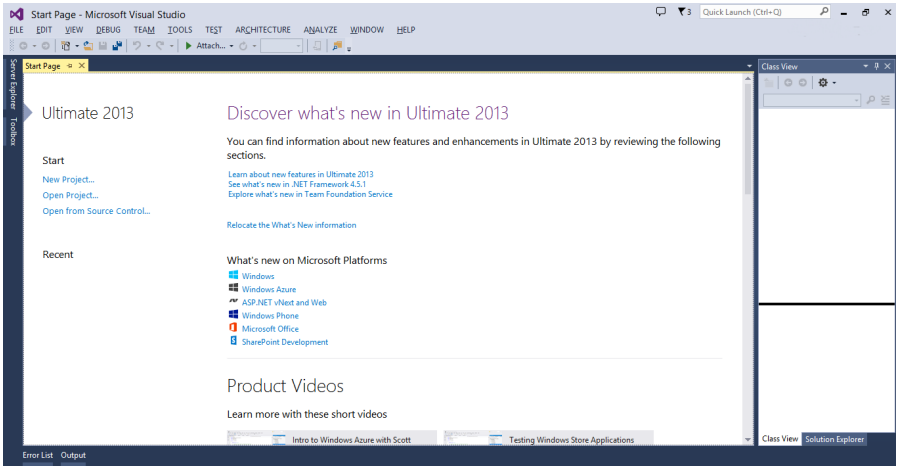
Əgər hər şey uğurla başa çatdısa onda installasiyanın sonunda “LAUNCH” düyməsi olan pəncərə görəcəksiniz. Onun üzərinə basaraq siz, Visual Studio - nu ilk dəfə olaraq işə sala bilərsiniz, həmçinin Windows proqramlarının (Start menyusu,

Əvvəl gördüyümüz işlərin uğurlu olmasına baxmaq üçün Visual Studio-nu işə salaq. İlk işə saldıqda sizdən sizin Microsoft Live İD - i daxil etməyi xahiş edə bilərlər. Eyni zamanda siz onu daxil etməyə də bilərsiniz. Bunun üçün «Not now, may be later» bölməsini seçin.

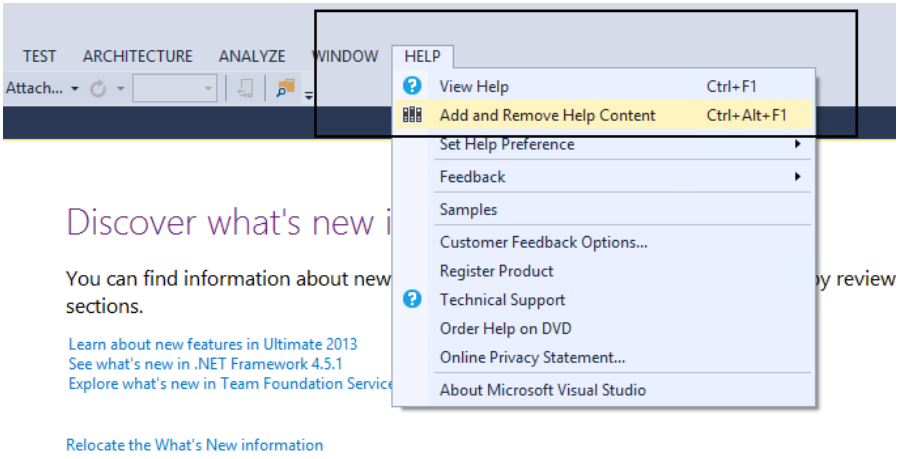




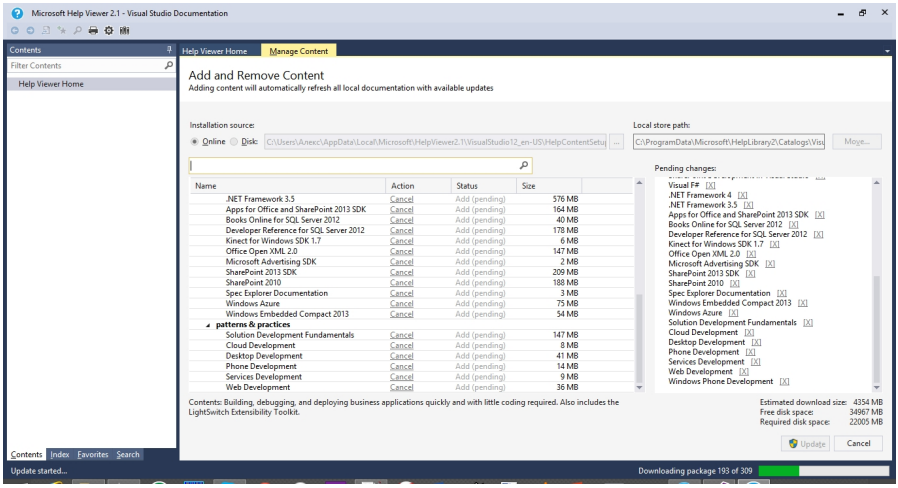




Əlavələrin öyrənilməsi və tərtibatı zamanı sizə Microsoftun informasiya sorağı (MSDN) çox kömək olacaq. Sorağın lokal kopiyasını quraşdırmaq üçün **Help->Add and Remove Help Content** menyu punktunu seçmək və yaxud **Ctrl+Alt+F1** basmaq lazımdır.



Siz soraq materialı seçimi ilə olan pəncərə görəcəksizniz. Hər yanda “Add” müraciətinə basmağı tövsiyyə edirik. Soraqlar çox sayda olmur :) Materialların seçimindən sonra onların lokal quraşdırılması üçün “Update”düyməsinə basın.

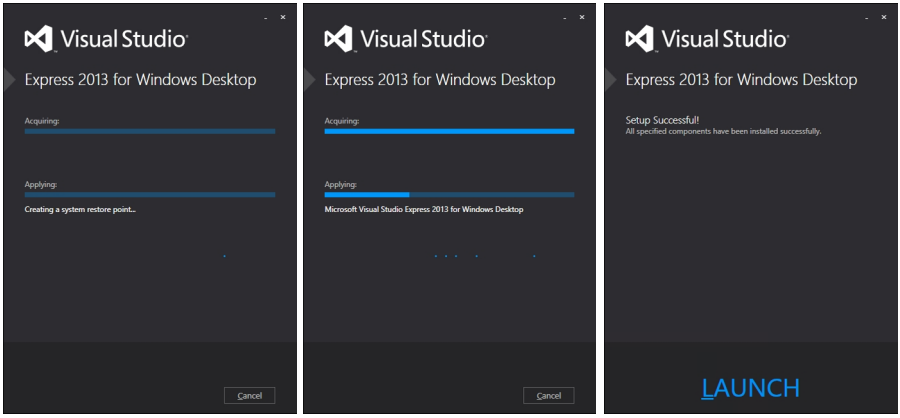


Windows Desktop üçün Microsoft Visual Studio Express 2013 installasiyasının prosesinə baxaq. Bu Utimate - in installasiyasıyla çox oxşardır. Expressin installasiyasını başlamaq üçün DVD diski DVD ötürücüsünə salın (və yaxud DVD diskin İSO - obrazını, misal üçün Daemon Tools Lite proqramının köməkliyi ilə yaradın) və wdexpress_full.exe faylını işə salın, bundan sonra sizdə belə bir pəncərə açılacaq:

Və yenə də pəncərədə Visual Studio-nun installasiya yolunu seçib müəyyən edə bilərsiz ki, Visual Studio-nun təkmilləşdirilmə proqramında iştirak etmək istərdinizmi. Əgər siz keyfiyyətin yaxşılaşdırılmasındq iştirak etmək istəyirsinizsə, onda «Join the Visual Studio Experience Improvement Program …» opsiyasına quş qoyun. Quraşdırmaya başlamaq istəyirsinizsə

«I agree to the License Terms and Privacy Policy» bölümünü seçin. Bu opsiyanı seçərək siz təsdiqləyirsiniz ki, bu lisenziyanın verilmiş bütün bölümləri ilə tanış və razı oldunuz.

Əgər siz düzgün seçim etmisinizsə, pəncərənin aşağı hissəsində “İnstall ” yazılmış düymə çıxır. Ona basın. Bundan sonra siz, quraşdırılma üçün bundan əlavə lazım olan proqram produktlarını seçməyə imkan verən pəncərə görəcəksiz. Bu mərhələdə cari seçilmiş proqramları saxlaya bilərsiniz. Gələcəkdə sonraya qalan komponentləri qurmaq imkanınız olacaq. Prosesin davamı üçün “İNSTALL” düyməsinə basın. İnstallasiya prosesinə start verilməlidir. Bu bir qədər vaxt alacaq, bu vaxt ərzində işin proqressini müşahidə edə bilərsiniz. Aşağıda biz bu vəziyyətə xarakter olan pəncərələrin nümunəsini gətiririk.



Əgər hər şey uğurla başa çatdısa onda installasiyanın sonunda “LAUNCH” düyməsi olan pəncərə görəcəksiniz. Onun üzərinə basaraq siz Visual Studio - nu ilk dəfə olaraq işə sala bilərsiniz, həmçinin Windows proqramlarının (Start menyusu, yarlıklar və s.) işə salınmasının adəti interfeysindən istifadə edə bilərsiniz. Ultimate-in installasiyasında olduğu kimi, burda da sizin LİVE İD (onu daxil etməsəz də olar) soruşacaqlar, həmçinin soraq sistemini də quraşdırmağı yaddan çıxarmayın. Əgər installasiyanın bütün bölümlərini uğurla sona çat-dırdızsa, siz öz yeni proqramınız yazmağa hazırsınız. Budur. İndi siz dərsimizin digər bölümlərinə keçə bilərsiniz, burada biz proqramlaşdırmanı öyrənəcəyik.

3. Microsoft Visual Studio 2015**-in instalasiyası**

İllər ötür və Visual Studionun yeni versiyaları yaranır. Siz artıq Microsoft Visual Studio 2013 installasiya edib və quraşdırağı bacarırsınız. İndi isə biz Visual Studio 2015-in nə olduğu barədə danışacayıq. Başa düşdüyünüz kimi, bu 2015 ci ildə yaranmış Visual studionun daha yeni versiyasıdır. Əlbəttə ki, istənilən proqram produktunun yeni versiyası köhnəsindən daha yaxşı və güclüdür və sözsüz ki, Visual Studio - da bu qaydalara riayət edir. Visual Studionun üç versiyası vardır: Visual Studio Community 2015, Visual Studio Professional 2015, Visual Studio Enterprise 2015. Onların hər biri haqqında danışaq.

Visual Studio Community 2015 - bütün tərtibatçılara proqramı pulsuz yükləməyə imkan verən versiyadır. O, zəngin imkanlara malikdir. Onu bütün: kommersiya və qeyri-kommersiya məqsədlər üçün istifadə etmək olar. Bu proqramla üçün tanışlıq üçün gözəl seçimdir.

Visual Studio Professional 2015 - məhsulun pullu verisyasıdır. Pulsuz versiyadan fərqli olaraq çoxsaylı təkmilləri vardır.

Visual Studio Enterprise 2015 - tam məhsulun pullu verisyasıdır. Ən mürəkkəb korporativ əlavələrin tərtibatı zamanı lazım olan güclü funksionallığa malikdir.

Versiyaların geniş izahlı cədvəli müraciətdə ilə mümkündür.

Proektin Visual Studio 2015 - in bütün versiyalarda installasiya və yaradılması Visual Studio 2013 lə oxşardır. Bizim kursda işləmək üçün siz həmçinin Visual Studio Community 2015 - i də qoya bilərsiniz.

4. İlk Lahiyə

Sergey adlı bir şəxs, Praqa barədə öz təəssüratların təsvir edərək mənə yazmışdı: “Əgər siz, rus dilini bilməyən ofisiantla qarşılaşmısınızsa onda menyunu əlinizə götürüb nə istədiyinizi barmağla göstərin. Məsələn, çex dilində meyvə sözü,demək olar ki, tərəvəz kimi səslənir. Təsəvvür edin, siz tərəvəz salatı sifariş etdikdə, sizə nə gətirəcəklər?:)” Sizə naməlum olan Sergey, öz dostlarını mümkün ola biləcək səhvlərdən qorumağa çalışmışdır ancaq çətin ki, o bizə ofisiantın əvəzinə kompyuterin olduğu zaman kömək edə bilər. Əfsuslar olsun ki, sonuncu bilmir siz menyunun hansı sətrinə barmağınızı göstərirsiniz. Komputer pedantlıqla aydın göstərişlərə əməl edir. Komandalar isə ayrıca sərbəst dillərdən ibarətdir. Komyuterə aydın olan dillər - proqramlaşdırma dilləri adlanır. Nəticə: çex ofisiantla ortaq dil tapmaq üçün fərasətli olmaq lazımdır. Kompyuterlə ortaq dil tapmaq üçün isə bu dili bilmək lazımdır. Birinci bölümdən bilirsiz ki, C - proqramlaşdırma dilidir. Bu dil bizə kompyuterdən nə istədiyimizi tam olaraq başa salmağa imkan verir. Tam səmimi desək, kompyuter yalnız bir dil - maşın kodları dilini bilir. Nümunə kimi “Hello, world!” ifadəsini ekrana gətirən proqram kompyuterə “doğma” olan dildə təxminən belə görünür:

```
_YH¦ў+._5lЧN№+f-H¦ў+ +f+ ¤д-ы+fO_fO¦¤+эmN№¤+¤+¦ФN№+¤-;-wGГ=xФN№+MZP __ + @ A -¦ ¦ -!+ L-!T
u  ._5lЧ№+f-&ЛZ¶QW&бфЬN№+&Л &ЛJf&OB_ФN№
+f;sЛlФN№+Л&lt; _t=ИФN№+_G¶u_O¶WfM+-чfЛN№3+M-Ac°
_>] Л¦Л+ЛMЛm _сбшЬN№+Л 9ШФN№+t§Л§pФN№+шN№) fd¦ __+RPбшЬ
N№+Л 9ШФN№+tЛ§pФN№+шC) +  +Э
tAc°-щЗ-Л_ФN№+f;s §lФN№+Л +u"бАФN№+¤+MAc°ЎD At  Э+- 3+
ыўы•ы°ыЎыЇЛD$П$._4ЧN№++     ._ dЛ4 Ў- ubSЛ dr4 +__e --fУЙf-+
-+ЙА+@УN№+Л-[Xf +tRPRVh•ў+ш_S Z+efO-fЛцf-d¤+§ RfRfh ш4
Л+Zы¦SшC¤д-ы¦¤+ьM-¤++¤_e fM+fO_fO¦f¤бЛЁdЎ  t3+ОшAc°Л§АФN№+-
dб4 ЙdЙ§4 f_f _t dЛ4 Ў- uSSЛ dr4 +__e --fУЙf-+-+ЙА+@УN№+Л-[
+eЙ]№fM+fO-fЛц+ыf-d¤+§ RfRfh шз Л+Zы+Sш¦
$ PE L  ¦7 p ¤!
` P  oK   P ў+       0
```

Siz bu cür kodla yazmaq mümkün deyildir deyəcəksiz və bu tamamilə düzdür!! Lakin, biz belə yazmayağıq. Elə məs proqramın tərtib olunmasını yüngülləşdirmək üçün proqramlaşdırma dili lazımdır. Proqramlaşdırma dilləri iki əsas qrupa bölünür: İNTERPRETASİYA oluna bilənlər və KOMPİLYASİYA edilə bilənlər. Bü cür bölgü komandaların proqramlaşdırma dilindən maşın dilinə hansı ixtisaslaşmış proqram tərcümə etməsindən asılıdır - KOMPİLYATORun və ya İNTERPRETATOR. Gəlin, bunlar arasında olan fərqi ayırd edək. Təsəvvür edək ki, bizdə komanda dəsti olan fayl vardır

Birinci situasiya. Komandalar interpretasiya olunan dillərdə yazılmışdır.

Komandanın hər işə salındığı zaman interpretator kodun yoxlanmasını sətri olaraq edir. Əgər sintaksisdə səhv yoxdursa, onda komandalar maşın koduna (prosessor üçün instruksiya dəsti) çevriləcəkdir. Proqram icra olunmaq üçün işə düşəcəkdir. Əgər səhv varsa onda interpretator dayanacaq və sizə onu düzəldib və proqramı yenidən işə salmağınız təklif olunacaqdır. Amma səhv olmasa belə yenə də proqramın

hər işə salındığı zaman interpretator işə düşəcək və yenidən kodu yoxlayacaqdır.

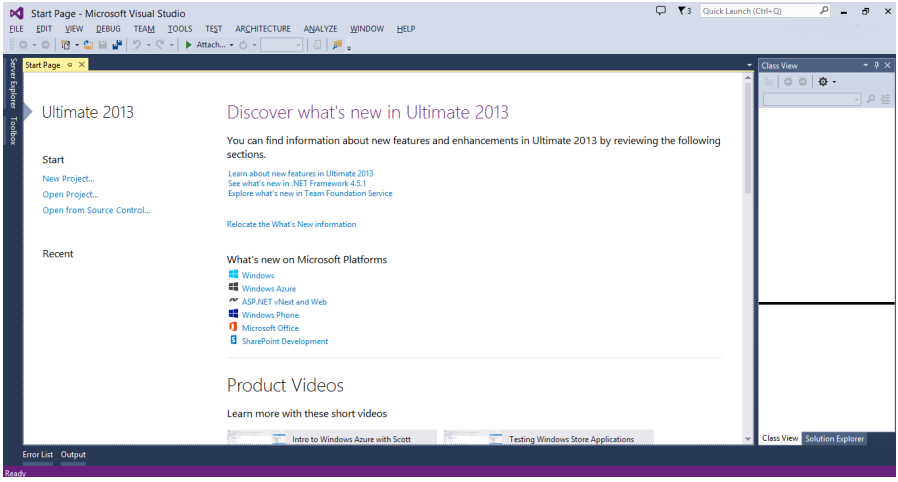
Beləliklə, kodun maşın verisyasının heç bir yerdə saxlanılmaması qənaətinə gəlmək olar. Bu cür yanaşmanın mənfi cəhətləri odur ki, proqramın işə düşmə sürəti yavaşıyır, yoxlamanı isə heç cürə söndürmək olmur.

İkinci situasiya. Komandalar kompilyasiya olunmuş dildə yazılmışlar.

Kompilyator da demək olar ki, həmçinin interpretator kimi fəaliyyət göstərir, yəni kodu sonacan yoxlayır, səhvləri tapır və bu barədə ismarıclar verir. Bundan əlavə kompilyator .OBJ genişlənməsi olan xüsusi obyekt faylı yaradır. Bu faylda maşın dilinə tərcümə edilmiş proqram mətni saxlanılır. Lakin, proqram bu faylla birbaşa işləmir. Komponasiya və linkləmə adlı anlayış vardır. Linkçi - daha bir xüsusi proqramdır ki, maşın kodunu (.OBJ genişlikli fayldan) və müxtəlif köməkçi məlumatları icra olunan vahid .EXE genişlikli fayla toplayır. Bu cür fayl icra olunmaq üçün ayrıca təkbaşına proqram kimi işə salına bilər və onun işə düşməyində, kompilyator iştirak etmir. Bizim halda, Microsoft Visual Studio ilə iş zamanı kompilyatorun çağırışı avtomatik olaraq edilir və çox asanlıqla C dilinin komandalarını maşın koduna çevirmək imkanı verir.

Qələmin sınağı

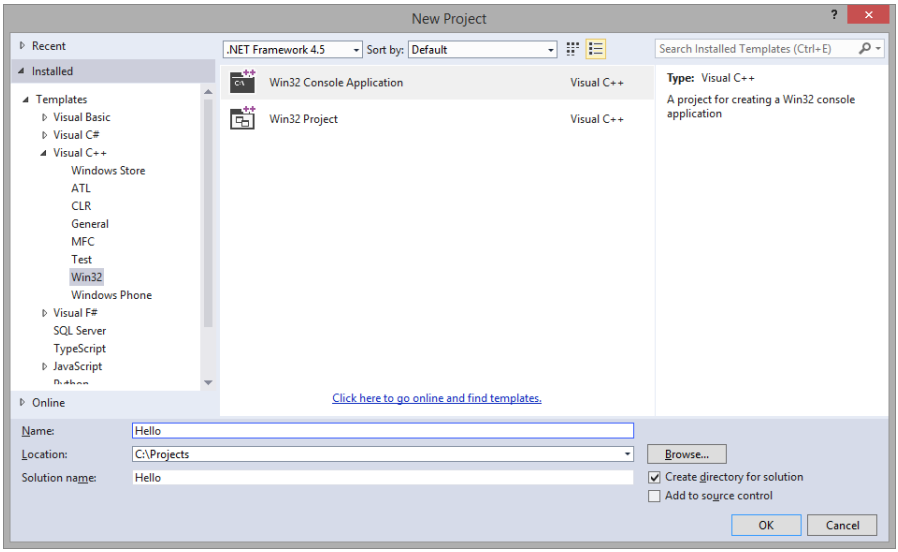
C dilinin təmələlçilərindən biri Brayan Kerniqan söyləmişdir: “ Proqramlaşdırma dilinin öyrənilməsinin yeganə yolu həmin dildə proqram yazmaqdır.” Biz də indi sizinlə elə bununla məşğul olacayıq. Yeni dildə olan ilk proqramın “Hello world!” proqramı olması proqramlaşdırma dünyasında artıq bir növ adət ənənədir. İlk proqramı yazmağa başlamaq üçün, ilk öncə Microsoft Visual Studio 2013 proqramının yarlıkını menyu bölməsindən işə salmaq lazımdır «Start» -> «Bütün proqramlar» -> «Microsoft Visual Studio 2013» və yaxud sizin üçün digər adəti üsulla. İşə salınmadan sonra, biz şəkildəki göstərilən kimi bir təsvir görəcəyik:



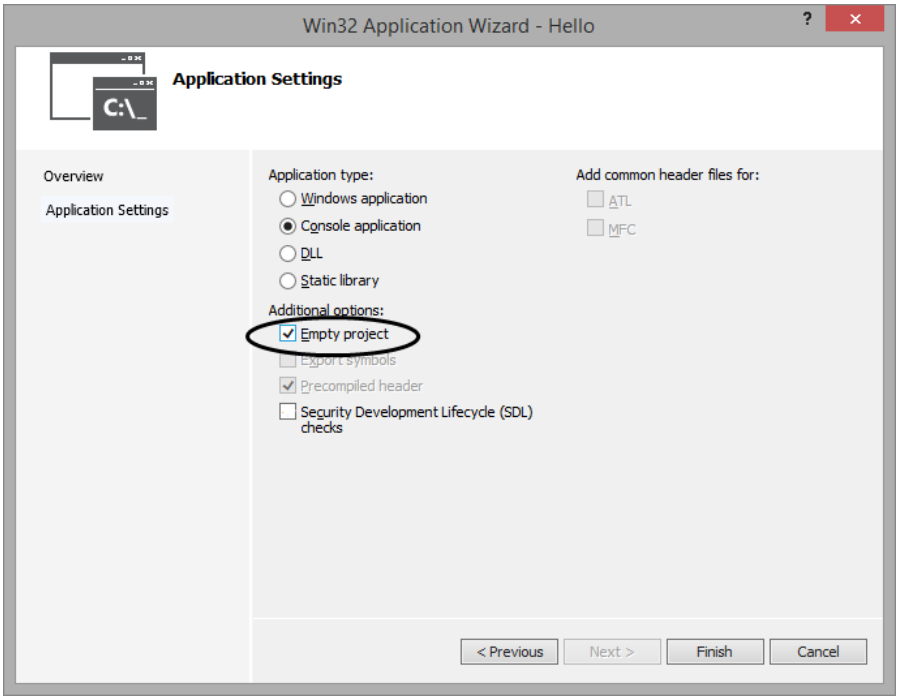
Son nəticədə bizim proqramı təsvir edən proekti yaratmağa çalışaq. Xırdalıqı ilə, böyük proqramlar yazarkən proektin nə olduğuyla sonra məşğul olacayıq. Hələ ki, biz proekti bir neçə faylın birləşməsi kimi təsvir edək.

İndi isə gəlin addım-addım nəzər salaq:

- Qabığı seçdikdə sonra, menyuda File -> New -> Project bölgülərini seçin.Qarşınızda dialoq pəncərəsi çıxacaqdır.
- Açılmış dialoq pəncərəsində - New Project (yeni proekt) Installed Templates (şablonlar) siyahısında Visual C++ -> Win32 -> Win32 Console Application proekt tipini seçin.
- Location sahəsində proektimizin hansı diskdə və hansı qovluqda olacağını qeyd edək. Bunun üçün, bu sahədə C:\Projects (və yaxud sizə uyğun olan, sizin ev proektlərinizin saxlanılacağı qovluğu) yazın.
- Proektimizə ad verək, bunun üçün Name sahəsində proektin adını yazın (məsələn “Hello”).
- İndi isə OK düyməsini basmaq olar.

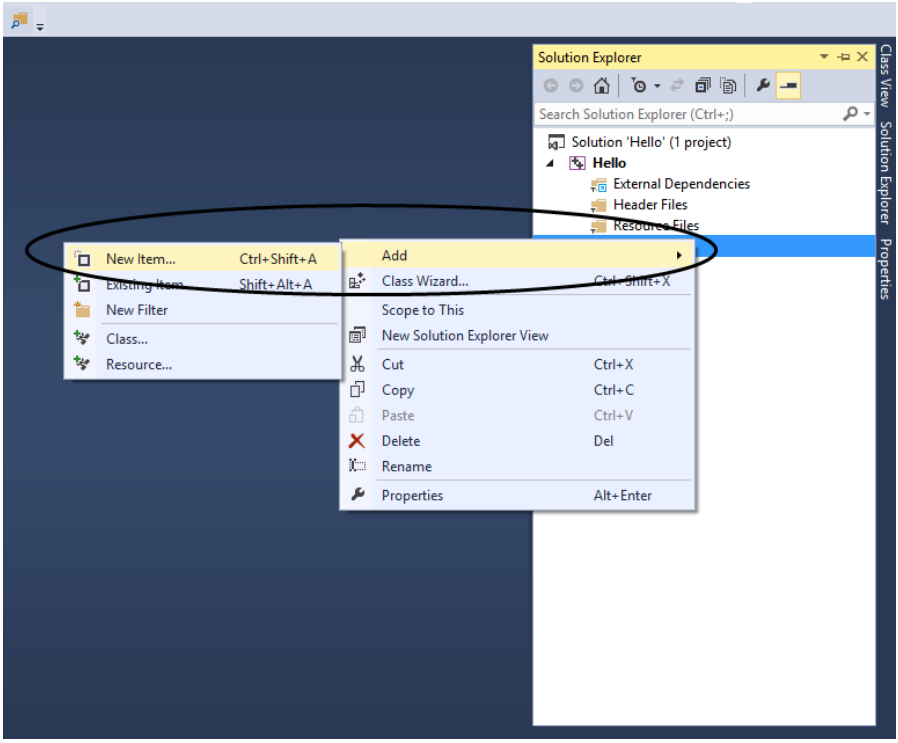


Qarşınızda proektin sazlanma xassələri pəncərəsi açılmışdır - Application Settings bölümünü seçin. Empty Project sahəsində quş qoyun - bu o deməkdir ki, biz boş proekt yaradırıq. Həmçinin, Security Development Lifecycle (SDL) checks - dən quşu götürün. İndi isə Finish düyməsinə basın.



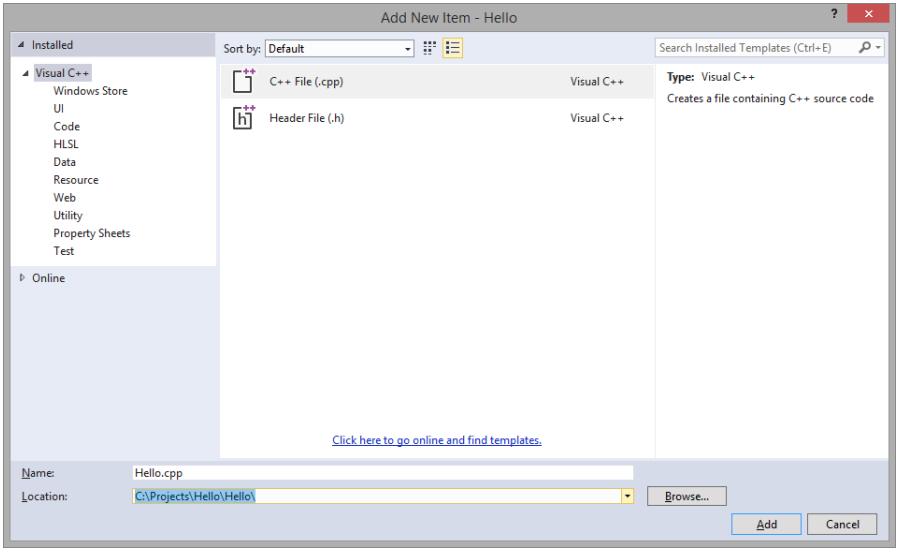
Beləliklə, biz, proqramımızın yerləşdirilməsi üçün yer hazırladıq. Nail olunanda dayanmadan, proektə təmiz fayl əlavə edək. Bunun içində proqramımızın mətnini yazacayıq. Bunun üçün növbəti hərəkətləri etmək lazımdır:

- Sag tərəfdə Solution Explorer adlı pəncərə peyda olmuşdur. Bu pəncərənin çağırılması üçün həmçinin Ctrl+Alt+L kombinasiyasını istifadə edə bilərsiniz. Bu pəncərədə Source Files qovluğunun üzərində siçanın sağ düyməsini sıxmaq lazımdır.
- Açılan menyuda Add->Add New Item… seçirik



Faylların seçim pəncərəsi açılır. Burada yenə də sizin geniş seçiminiz vardır. İndi C++File (.cpp) (C/C++ dilində proqramdan ibarət fayl) nişanını seçməyi məsləhət görürük.

- Name (faylın adı) mətn sahəsində “Hello” adını yazın. Add düyməsini basın.



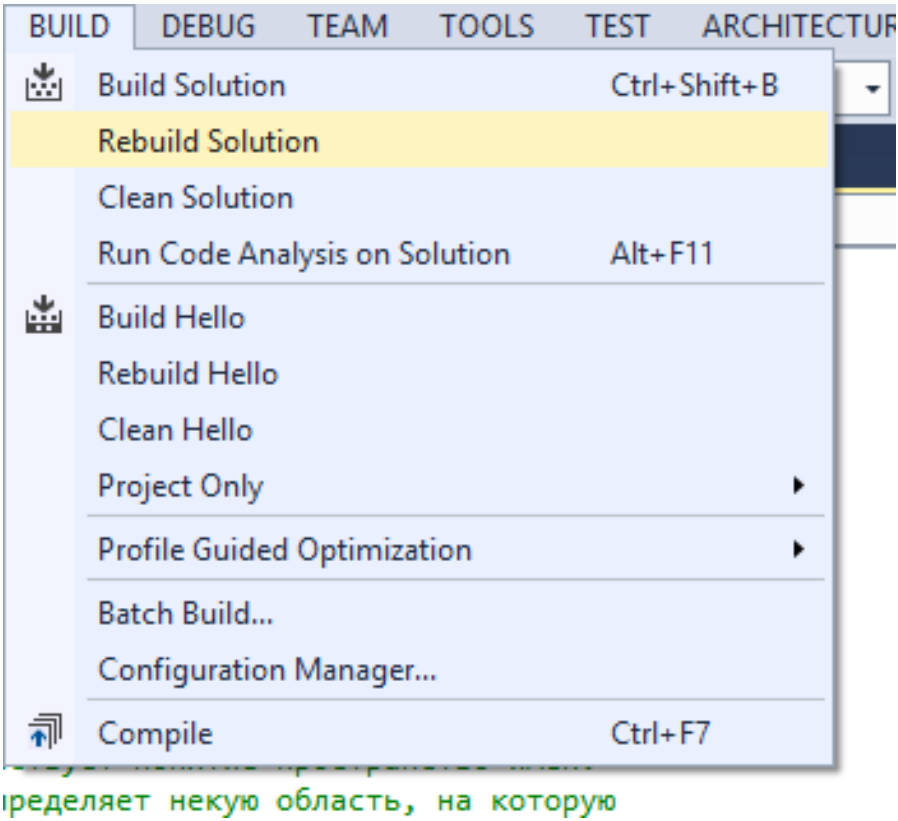
Add düyməsinə basdıqdan sonra mətn sahəsi yaranacaqdır, burada siz ilk proqramınızı yaza bilərsiniz.

C dilində ilk proqram nümunəsi

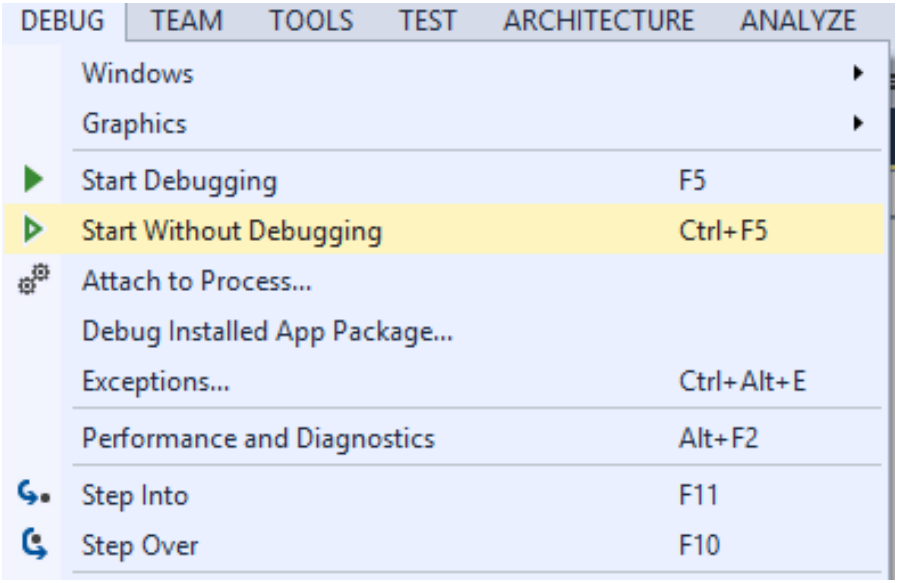
Proqramı yazmazdan öncə, gəlin rahatlıq üçün kommentariy anlayışını daxil edək. Kommentari - proqrama qeydlərdir, bunlar yalnız proqramçılar üçün nəzərdə tutulmuşdur. Kompilyator onları iqnor edir. Misal üçün, kommentarinin köməyi ilə proqramın, hansı bu və ya digər sətrinin nəyə görə istifadə olunduğunu qeyd etmək olar. Aşağıda göstərilmiş nümunədə kommentarilərni necə idarə edildiyi göstərilmişdir. İndi isə, yaranmış mətn pəncərəsində növbəti kodu yazaq:

```
// Это комментарии к программе
// Они выделяются зеленым цветом
// Начинаются комментарии двумя черточками //
/* если необходимо создать многострочный
комментарий, используется конструкция
/* комментарий */
/* Данная строчка подключает в программу библиотеку под названием
 iostream. Библиотека - файл, в котором содержатся описания
 различных функций, реализованных другими программистами.
 Данная программа получила возможность использовать функции
 находящиеся в библиотеке iostream */
#include <iostream>
/* В языке C++ существует понятие пространство имен.
 Это пространство определяет некую область, на которую
 приходятся действия оператора или функции. Для того, чтобы использовать
 оператор, находящийся в определенном пространстве,
 необходимо подключить это пространство в свою программу.
 Ниже подключается пространство под названием std */
using namespace std;
void main() // Начало программы, отсюда программа начнет своё выполнение
// Весь текст программы располагается между фигурными скобками
{          //Это фигурная скобка
           // Следующая строка выводит на экран приветствие Hello, World!
          // Данное действие осуществляется с помощью cout<< Именно для его
  // работы подключалась библиотека и пространство имен, в которых он располагается
         cout<<"Hello, World!\n";
         // В конце команды стоит точка с запятой. Этим знаком ДОЛЖНА заканчиваться
        // каждая команда в языке C.
}
```

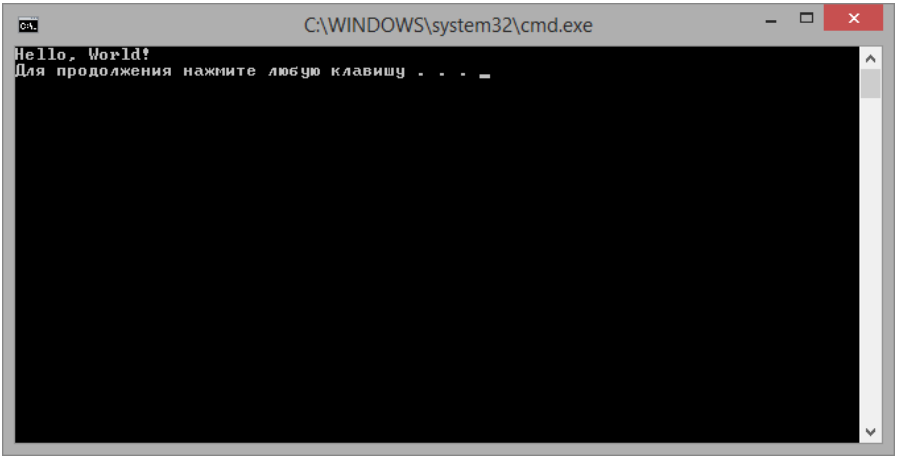
Bildiyiniz kimi kompyuter yalnız maşın kodlarının dilini başa düşür. Və proqramın kompyuter tərəfindən icra olunmasından öncə onu maşın kodları dilinə tərcümə etmək lazımdır. Yadda saxlayın ki, bunu biz yo, KOMPİLYATOR edəcəkdir. C - də yazılmış proektlər özündə bir çox fayl cəmləyir. Proektə daxil olan bütün faylları kompilyasiya edək. Bunun üçün Build menyu sətrini seçin, sonra isə Rebuild Solution (hər şeyi yenidən quraşdırma)



Ümid edirik ki, mətni səhvsiz yazdız. Bizim proqram uğurla maşın dili koduna tərcümə edilmişdir və onu artıq qoşmaq olar. Proqramın icrası yetəri qədər sadə prosesdir. Debug menüsunda Start Without Debugging-i seçin.



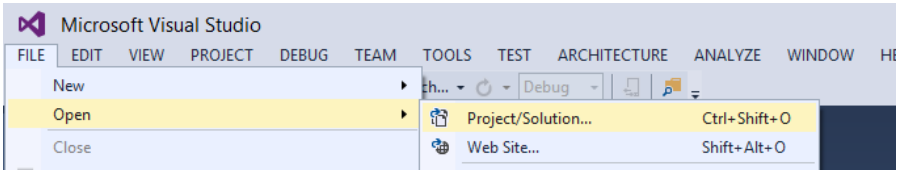
Proqram işi görəcək və siz növbəti pəncərəni görəcəksiniz:

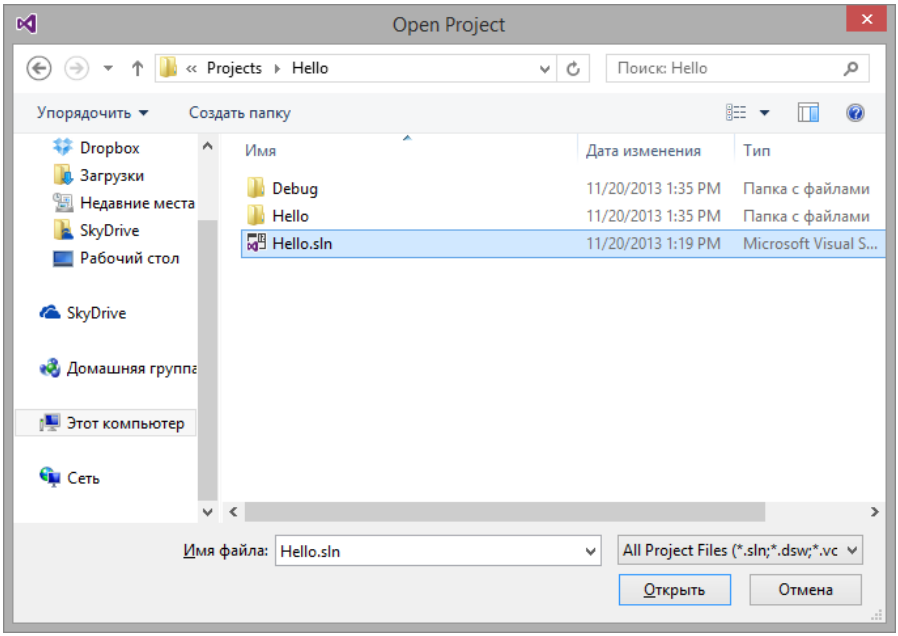


Təbrik edirəm! Olduqca asan idi, elə deyilmi? İşi başa çatdırmaq üçün klaviaturada istənilən klavişi sıxın (Press any key to continue).

Yaddaşa verilmiş proektin açılması

Bundan öncə yaddaşa verdiyiniz proektin yenidən-qurması üçün diskdə (əgər hələ yükləməmişsinizsə) Visual Studio-nu işə salın. File menyusunda Open ->Project/ Solution bölümünü seçin və proektinizin adını seçin.





Proekt açıldıqdan sonra onun üzərindəki işinizi davam etdirə bilərsiniz. Dərs çərçivələrində yaradılan proektləri dərsin alt qovluğunda Source adıyla tapa bilərsiniz. Bu proekt əsasən /Sources/Hello qovluğu istiqamətində yerləşir.

5. Məlumatların çıxarılması

Siz artıq bilirsiniz ki, cout<< komandasının köməkliyi ilə ekrana müxtəlif mətn sətrlərini çıxara bilirik. Lakin, kompilyatorun belə komandanı başa düşməsi üçün, üç əsas amili yadda saxlamalıyıq:

1. Proqramın başlığında #include <iostream> sətri olmalıdır.
2. Komandanın istifadə olunmasından qabaq, cout komandasının mənsub olduğu ad fəzasını qoşmaq lazımdır.

```
using namespace std;
```

3. cout<< dan istifadə edərək ekrana gətirmək istədiyimiz sətir, məsələn

```
cout<<"здесь пишем то, что хотим";
```

cout<< komandası ekrana nəinki sətirləri çıxarır, həmçinin onlara düzəlişlər etməyə də imkan yaradır. Sətrin çıxarılmasının düzəlişi üçün xüsusi idarəedici simvollardan istifadə olunur, hansı ki, / simvolunun kombinasiyasını və sətir üzərində lazım olan işi təyin edən simvolu özündə təsvir edir. Bu idarəedici simvollar Escape - ardıcıllıqlar adlanır. Aşağıda onların bəziləri göstərilmişdir:

```
\b                Удаление последнего выведенного символа
\n                        Перейти на начало новой строки
\t                  Перейти к следующей позиции табуляции
\\                               Вывести обратную черту \
\"                             Вывести двойную кавычку "
\'                           Вывести одинарную кавычку '
```

Son üç Escape ardıcıllıqın varlığı əvvəlcə bir qədər yüngül anlaşılmamazıq yaradır. İdarəedici simvollardan niyə istifadə olunsun ki, sadəcə " və ya \ yazmaq olmaz ki? Cavab səthdədir, bütün bu üç simvollar operatorlardır, və əgər onları "sadəcə yazsaq", onda kompilyator onları elə operator kimi qəbul edəcəkdir. Misal üçün, sözlər məcazi mənada istifadə olunursa, onlar dırnaq arasında yazılır.

Misal üçün, sizə növbəti testi ekrana çıxarmaq lazımdır:

```
The Man in red was "old friend" of John...
```

Əgər siz escape-ardıcıllıqlardan istifadə etmirsinizsə sizin kod belə olacaq:

```
cout<<"The Man in red was "old friend" of John...";
```

Bu isə qaçılmaz səhvə gətirib çıxaracaq. Kompilyator sətrin bir hissəsini anlayacaq, əsasən cout<<''The Man in red was". Was- dan sonrakı ikili dırnağı bağlayan kimi hesab edəcək, qalanları isə dilin səhv sintaksisi kimi qəbul edəcək. Əlbəttə ki, bu cür proqram icra olunmaq üçün işə düşməyəcəkdir. Düzgün variant belədir:

```
cout<<"The Man in red was \"old friend\" of John...";
```

İndi isə gəlin cout<< - da məhz harada Escape ardıcıllığını göstərməkdən danışaq. Ən əsas bilməli olduğunuz - Escape ardıcıllıq hər zaman dırnaq arasında yazılmalıdır, belə ki, o mətndir, sonra isə sizin imkanlarınız məhdudu deyildir. Misal üçün:

```
cout<<'' My name is''<<'' - Ira\n '';
cout<<''I'm from Odessa\n '';
cout<<''My eyes are blue"<<"\n ''<<"That`s all!!!";
```

Bu komandanın işi nəticəsində biz görəcəyik:

```
My name is - Ira
I'm from Odessa
My eyes are blue
That`s all!!!
```

cout<< istifadəsinə klasik nümunə

Proqram yazaq, o ekrana öyrəndiyimiz Escape ardıcıllıqları barədə qısa soraq çıxarır. Ekranda biz bunu görmək istəyirik:

```
\b                                  Backspace
\n                                   New line
\t                                Horizontal tab
\\                                   Backslash \
\"                          Double quotation mark "
\'                          Single quotation mark '
```

Visual Studio 2013 işə salırıq. EscapeSequences adlı yeni proekt yaradırıq. Aşağıda göstərilən kodu yığırıq.

```
// Заголовок
#include <iostream>
// определение пространства имен, в котором есть cout<<
using namespace std;
// Главная функция
void main()
{
/* Следующая команда через 4 табуляции выводит текст
   Escape Sequences
   и переводит вывод на следующую строку */
                                                cout<<"\t\t\t\tEscape Sequences\n";
             // Выводит пустую строчку
                                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \b,
   и еще через 1 табуляцию Backspace
   Затем \n переводит вывод на следующую строку */
                                                  cout<<"\t\t\\b"<<"\tBackspace\n";
            // Выводит пустую строку
                                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \n,
   и еще через 1 табуляцию New line
   Затем \n переводит вывод на следующую строку */
                                                   cout<<"\t\t\\n"<<"\tNew line\n";
           // Выводит пустую строку
                                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \t,
   и еще через 1 табуляцию Horizontal tab
   Затем \n переводит вывод на следующую строку */
                                               cout<<"\t\t\\t"<<"\tHorizontal tab\n";
             // Выводит пустую строку
                                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \\,
   и еще через 1 табуляцию Backslash \
   Затем \n переводит вывод на следующую строку */
                                                cout<<"\t\t\\\\"<<"\tBackslash \\\n";
             // Выводит пустую строку
                                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \t,
   и еще через 1 табуляцию Double quotation mark "
   Затем \n переводит вывод на следующую строку */
                                  cout<<"\t\t\""<<"\tDouble quotation mark \"\n";
            // Выводит пустую строку
                                                                        cout<<"\n";
/* Через 2 табуляции выводит текст \',
   и еще через 1 табуляцию Single quotation mark '
```

```
   Затем \n переводит вывод на следующую строку */
                              cout<<"\t\t\'"<<"\tSingle quotation mark \'\n";
            // Выводит пустую строку
                                                                  cout<<"\n";
}
```

Proqramı kompilyasiya edək (Build -> Rebuild Solution). Əgər çoxsaylı səhvlər çıxırsa, növbəti qaydaları yada salın:

Əgər proqramda ekrana ismarıclar çıxırsa onda proqramın əvvəlinə #include sətri yazılır və ad fəzası qoşulur, ona cout (using namespace std;) komandası məxsusdur. Hər bir proqram main() adlı funskiyaya malik olmalıdır. Proqramın işi bu funksiyanın icra olunmasıyla başlayır. main () funksiyasının komandaları { } fiqur mötərizələrinin içində yerləşir. Bütün komandalar mütləq vergüllü nöqtə ilə sona yetirilməlidirlər. Proqramı işə salaq (Debug -> Start Without Debugging).

P.S. Yəqin fikir verdiniz ki, məlumatları ekrana gətirmək üçün biz latın simvollarından istifadə edirik. Bu onunla bağlıdır ki, biz sizinlə OC Windows- da yazırdıq ona icrasını isə MS DOS həyata keçirir. Buna görə də sistem onu bu kodla identifikasiya edir. MS DOS və Windows-da kirilin simvol kodları uyğun gəlmir, buna görə kirildən istifadə olunan proqram düzgün işləməyəcək. Misal üçün biz Windows-da yazmışıq:

```
cout<<"Утро";
```

Ekrana çıxarılan isə

```
µЄЁю
```

Bunu belə başa salmaq olar ki, Windows-da misla üçün o hərfi - 238, DOS da isə bu koda ю hərfi uyğun gəlir. Latının kodları isə hər iki ОС-da eynidir. Nəticədə biz sizinlə bu situasiyaya düzəliş etməyi öyrənəcəyik. C++ 11 dil standartında proqramçılara xüsusi simvolların ekrana çıxarılması üçün yeni imkanları təqdim etmişlər. Buna görə «raw» sətrindən istifadə etmək lazımdır. «raw» kimi sətrin elan olunması kompilyatora onu simvol-simvol traktə etməyə göstəriş verir. Sətri bu cür sətir kimi elan etmək üçün növbəti formatdan istifadə etmək lazımdır.

```
R"(текст_строки)"
```

R, «raw» - nun sətir olmasına işarə edir. Sətrin tərkibini mütləq mötərizəyə almaq lazımdır. Bir neçə misal çəkək:

```
cout<<R"(hello\nworld)"; // на экране hello\nworld
   cout<<R"("Test 'string'\t")"; // на экране "Test 'string'\t"
   cout<<R"((Such brackets))"; // на экране (Such brackets)
```

6. Məlumat tipləri

Dərsin əvvəlki bölümlərini oxuduqdan sonra, sizin üçü, ekrana hər nəsə çıxaran proqramı yazmaq elə bir şey deyil.

```
// Заголовок
#include <iostream>
// определение пространства имен, в котором есть cout<<
using namespace std;
// Главная функция
void main()
{
                              // вывод фразы "Поздравляем с хорошим началом!!! :)"
                                          // фраза выводится через три табуляции,
                                           // затем добавляются две пустых строки
                      cout<<"\t\t\tCongratulation with good beginning!!! :)\n\n";
}
```

Hələlik bacardıqlarınızdır bu qədərdir. İnsan heç zaman nail olduqlarında dayanmamalıdır. Sizə tək məlumatların ekrana çıxarılması deyil, həm də bu məlumatlarla operasiya etmək də maraqlı gələcək. Misal üçün, hansısa hesablamaları aparmaq. Əlsiz jonqlyorlar olmur və şarların bir hissəsinin göydə olduğu zaman, qalan hissəni sirk ustası əllərində tutur. Nəyisə saxlamaq üçün (əsasən də məlumatları) konteyner olmalıdır. Bizim proqram üçün bu cür konteyner operativ yaddaş olacaqdır. Nəyisə hardasa yerləşdirmək üçün, ona uyğun bağlamanı seçmək lazımdır. Məsələn südü kibrit qutusuna çətin ki tökərsiniz. Proqramlaşdırmada informasiyanı operativ yaddaşa yerləşdirməkdən qabaq, siz mütləq bu informasiyanın xarakterini təyin etməlisiniz.

Beləliklə, **məlumat tipləri**. Məlumat tipi - proqramın istifadə edəcəyi maksimal ölçüləri (baytlarla) və informasiya tiplərini müəyyənləşdirən anlayışdır. Proqramlaşdırma xarici dünyanın obyektlərini onları sadələşdirərək əks etdirir. Öyrənilmənin əvvəlində tez-tez rastlaşdığımız anlayışlarla rastlaşacayıq. Gəlin məlumat tiplərini şərti olaraq aşağıdakı qruplara bölək:

1.Ədədi.

2.Simvollu

3.Məntiqi

Növbəti olaraq, C dilində məlumat tiplərinin işarələnməsi üçün istifadə olunan açar sözləri sırasına nəzər salacayıq.

Ədədi tiplər.

Ədədlər məlum olduğu kimi, tam və cismi olur. Cismi ədədləri biz üzən nöqtəylə olanları adlandıracayıq. Xüsusilə qeyd edək ki, tam hissəni, kəsri hissədən ayıran vergül nöqtəyə dəyişir. Misal üçün 7,8 - 7.8 kimi yazılır. Cismi ədədlərin qiymətlərini saxladığımız dəyişkənlər, float və double kimi elan ediləcək. Bu tiplər arasındakı fərqlər nədədir?

Float tipini tək dəqiqliyi olan, üzən nöqtələri olan ədədləri təsvir edir, double isə ikili. Bildirək ki, riyaziyyatda dəqiqlik, ədədi təqdim edən rəqəmlərin sayı ilə təyin olunur. İkili dəqiqlik isə, rəqəmlərin adi sayı ilə müqayisədə iki dəfə artırılan rəqəm təsvirinin metodu adlandırırlar. Üzən nöqtəli lər üçün tiplərin xarakteristikası:

Aydınlatma	Tip	Baytla ölçüsü
Tək dəqiqliyi olan cismi ədədləri təsvir edir	float	4
İkili dəqiqliyi olan cismi ədədləri təsvir edir	double	8

Cismilərdən savayı, C -də tamədədli məlumatları elan edən 3 tip nəzərdə tutulmuşdur. Cədvəldə bu tiplərin əsas xarakteristikaları göstərilmişdir:

Aydınlatma	Tip	Baytla ölçüsü	Dəyər diapazonu
Tam ədədləri təsvir edir	int	4	от -2147483648 до 2147483647
Qısa tam ədədləri təsvir edir	short	2	от -32768 до 32767
Uzun tam ədədləri təsvir edir	long	4	от -2147483648 до 2147483647
Uzun tam ədədləri təsvir edir	long long	8	от –9,223,372,036, 854,775,808 до 9,223,372, 036,854,775,807

Simvol tipi

Yalnız bir simvolun saxlanması üçün nəzərdə tutulub. Dərhal qeyd edək ki - C -də sətrlərin saxlanması üçün tiplər mövcud deyildir.

Aydınlatma	Tip	Baytla ölçüsü
Simvolları təsvir edir	char	1

Məntiqi tip

Məntiqi məlumatların saxlanması üçün nəzərdə tutulmuşdur. Sonra onunla daha ətraflı tanış olacayıq. Məntiqi tiplər iki ədəddən birini qəbul edə bilər: ya həqiqət (true) ya da yalan (false).

Aydınlatma	Tip	Baytla ölçü	Dəyərlər
Dinamik qiymətləri təsvir edir	bool	1	true false

Qeyd: Əgər məlumat tipi diapazonundan mənfi qiymətləri çıxarmaq lazımdırsa, tipin adından qabaq unsigned açar sözünü yazmaq lazımdır. Misal üçün, unsigned int. Bu cür özündə yalnız müsbət qiymətləri saxlayacaq misal üçün, 0 dan 4292967294

Beləliklə, biz ayırd etdik ki, hansı məlumat tipləri olur və onların işarələnməsi üçün C dilində hansı açar sözlərindən istifadə olunur. Sonda, qeyd edək ki, C dili reqistrə hissiyətcil olur (yəni burada BAŞ və kiçik hərflər eyni şey demək deyildir.) fikir verin yuxarıda göstərilən məlumat tipləri kiçik hərflərlə yazılmışdır. Buna fikir verin, belə ki int- məlumat tipidir, İNT - isə səhvdir. Növbəti mövzumuzda məlumat tiplərini praktikada tədbiq edəcəyik

7. Dəyişənlər və Konstantalar

Biz sizinlə artıq məlumat tipləriylə tanışıq və informasiyanın saxlanması üçün necə klassifikasiya edildiyini bilirik. Qaldı, öyrənək ki, onları əməli yaddaşa necə yazaq və onları istifadə edib dəyişmək üçün onlarla necə rəftar edək. Beləliklə, dəyişən məlumatları DƏYİŞƏNLƏR, daimiləri isə KONSTANTLAR adlandıracayıq.

Dəyişən - əməli yaddaş sahəsidir, öz adı vardır və dəyişə bilən məlumatları saxlamaq üçün nəzərdə tutulmuşdur.

Konstanta - əməli yaddaş sahəsidir, öz adı vardır, və daimi məlumatların saxlanması üçün nəzərdə tutulmuşdur.

Konstantlara nümunə:

Hamıya məlum olan həftənin günləri və ildə olan ayların sayı... Bunlar heç zaman dəyişmir və ona görə də bu qiymətlər konstantdır. Lakin, bizim yaşımız isə - dəyişəndir. Bu gün kiminsə 26 yaşı var, bir ildən sonra isə 27 olacaqdır. Bu aydınlatmalardan aydın olur ki, yaddaşda məlumatların axtarışı üçün onlara ad verilir (məsələn baqaj vaqonundakı əşyalara birkalar vurulduğu kimi.). Proqramlaşdırma sahəsində onları identifikator adlandırırlar. Yeni doğulan körpəylə bağlı ilk problem, valideynlərinin onun üçün ad seçməsidir.

Adın insanın taleyinə və xarakterinə iz qoyması sualı çətin və mübahisəlidir. Ancaq, o faktı ki, yeni dəyişən konstanta verilən ad (identifikator) intuitiv olaraq başa düşülən olmalıdır və dəyişənin mənasını bildirməlidir, heç bir təcrübəli proqramçı sübhə altına almayacaq. Məlumatlara adlar ciddi qaydalara uyğun olaraq verilir. Bu qaydaları pozmaq olmaz!

Adların tərtib olunma qaydaları.

Adlarda aşağıdakı simvollar yol veriləndir:

1. Latın əlifbasının böyük və kiçik hərfləri. Bu zaman dilin reqistr-asılı olduğunu unutmayın. Misal üçün, Age və age iki ayrı-ayrı adlardır.

2. Rəqəmlər. Lakin, rəqəm ilk simvol kimi işarələnə bilməz. Yəni, Name1 yol veriləndir, 1Name isə yox.

3. Aşağıdan xətt simvolu «_». İş burasındadır ki, siz yadda saxlamalısız ki, probel də simvoldur və bu simvol dəyişənin adında yol verilməzdir. Onu aşağıdan xətt simvolu əvəz edəcək, bu da adların görünüşünü yaxşılaşdıracaq. Misal üçün, müqayisə edin: ageofman və Age_of_Man.

Dəyişənlərə adları seçəndə aşağıdakıları yadda saxlayın:

1.Dəyişənləri dilin açar sözləriylə adlandırmaq olmaz. Açar sözü - proqramlaşdırma dilinin sintaksisi üçün rezerv edilmişdir (int, float, double və s.). Visual Studioda açar sözləri göy rənglə görünür, bu çaşmamağa kömək edir

2.İki eyni adlı identifikatorların mövcud olmağı məsləhətli deyildir.

3. İcazə verilmiş simvollardan başqa simvollardan istifadə etmək olmaz. (yuxarıda baxın)

Dəyişən konstantların elan olunması və istifadəsi.

İndi biz dəyişənin yaradılması (elan olunması) üçün bütün məlumatı əldə etmişik. Qaldı öyrənək ki, ümumi sintaksis necədir:

1. məlumat_tipi_dəyişənin_adı ; Bu halda, əməli yaddaşda verilmiş tipə uyğun ölçüdə gözlər ayrılacaq. Bu gözə sizin seçdiyiniz ad veriləcəkdir. Orada nə saxlanılacaq? Yenicə yaradılmış dəyişənə əməli yaddaş tərəfindən təyin edilmiş, ixtiyari rəqəm veriləcəkdir. Bu rəqəm yaddaşda o vaxta qədər saxlanılacaq ki, siz dəyişəni xüsusi mənimsəmə operatorunun = köməyi ilə başqa qiymətlə dolduracaqsınız.

2. məlumat_tipi_dəyişkənin_adı=qiymət; belə bir imkan da var ki, dəyişkəni yaradılma zamanı da qiymətlə doldura bilərsiniz. Bu cür prosesi biz inisializasiya adlandıracayıq.

3. məlumat_tipi_const_dəyişkənin_adı=qiymət; - bu isə konstantın elan edilməsidir. Əsas anlar bundan ibarətlar ki, hansı məlumat tipi olursa olsun, onun önündə açar sözü const göstərilir. Bundan əlavə konstanta yaradılma zamanı mütləq inisializasiya edilməlidir. Sonra onun qiymətini dəyişmək mümkün olmayacaq.

Dəyişkənin qiymətinin ekranda görünməsi. Dəyişkənin qiymətinin ekranda görünməsi üçün cout<< istifadə olunur.

```
cout<<имя_переменной; // кавычки в данном случае не указывают
```

Bir neçə dəyişənin tərkibini << ilə göstərmək olar.

```
cout<<имя_переменной1<<имя_переменной2;
// кавычки в данном случае не указывают
```

Mətn tərkibli və Escape ardıcıllıqlı dəyişənlərin tərkini << ilə növbələmək olar

```
cout<<"Текст"<<имя_переменной1
<<"Текст"<<имя_переменной2<<"\n";
```

Konstanların tərkibinin göstərilməsi də eynilə dəyişənlərdə olduğu kimidir.

Praktiki nümunələr

Müxtəlif məlumat tipləri üçün dəyişən və konstantların yaranmasına və inisializasiya edilməsinə nümunələr göstərək.

Tamədədli dəyişənlər və konstantlar

Tam ədədlərlə biz hər yerdə rastlaşırıq: yaş, stulların sytı, otaqların sayı, həftənin günlərinin sayı və s.

Tam ədədlər saxlanılan dəyişənlər bu cür elan edilir:

Int Age;

Bu sətir nədən danışır? Age (yaş) adlı dəyişəndə tam qiymət saxlanılacaq. int sözü isə Age adlı dəyişənin qiymət TİPİNİ elan edir.

İndi isə biz Age dəyişəninə 34 qiymətini daxil etmək istəyirik. Bunu necə edək?

Age=34;

Bu sətir belə oxunur: “Age dəyişəninə 34 qiyməti verilsin.”

Bir daha mənimsəmə operatoruna baxaq: Age=34;

Bərabərlik işarəsindən soldaq qiymətin mənimsəyən dəyişənin adı yerləşir, sağda isə verilən mənimsədilən qiymət.

Tam ədəd saxlanılan konstanta isə belə elan edilir:

Const int Count_Days_in_Week=7;

Bu sətir nədən bəhs edir? Const sözü konstantanın elan edildiyini bildirir. int xəbər edir ki, konstanta tam ədəd olacaq, sonra konstantın adı gəlir Count_Days_in_Week və onun qiyməti 7.

İndi isə gəlin dəyişənin qiymətini necə aydın etməyi nəzərdən keçirək.

Bu nə üçün lazımdır? Sadə misal, 2000 ci ildə neçə saatın olduğunu necə hesablayaq? Məyər, siz bütün bu rəqəmləri özünüz saymaq istəyirsiniz?

Əslində kompyuteri bunu çox asanlıqla etməyə məcbur etmək olar. Biz sadəcə bu hesablamanın formulunu yaza bilərik.

2000 ci ildə 366 sutka var, sutkada 24 saat var. Onda 2000 ci ildəki saatların hesablanma formulu belədir:

366 vuraq 24 saata.

C dilində vurma işarəsi kimi * işarəsindən istifadə olunur (ulduz - Shift+8 kombinasiyasıdır)

2000 ci ildəki saatların sayını sayan proqram tərtib edək.

Proqramın yaradılmasından öncə qısaca olaraq onun alqoritmini cızmağı məsləhət görürük.

Alqoritm - qarşıya qoyulan məsələni həll etmək üçün yerinə yetirilməsi vacib olan əməliyyatlar ardıcıllığıdır.

Verilib: ilin günlərinin sayı - 366. Bu qiymət dəyişməyəcək, onu görə onu DayIn_2000Year adlı tam tipli konstant kimi elan edək. Sutkanın günlərinin sayı - 24. Bu da dəyişmir. HourInDay tam tipli konstant kimi elan edək.Proqramımızda yeganə dəyişən olacaq, ona biz hesablamanın nəticələrini yazacayıq. Bu dəyikəni HourIn_Year2000 adlandıraq. O tam tipli olacaq (int).

Alqoritm növbəti olaraq olacaq:

1. Dəyişən və konstantların elanı və inisializasiyası.
2. Nəticənin hesablanması.
3. Ekrana nəticənin çıxarılması.

Dəyişənlərin adını özünüz fikirləşə bilərsiniz (adların qurulma qaydasını yaddan çıxarmayın).

İndi isə həmişə olduğu kimi proekt yaradaq və növbəti kodu daxil edək:

```
// Заголовок
#include  <iostream>
// определение пространства имен, в котором есть cout<<
using namespace std;
// Главная функция
void main()
{
          // вывод пустой строки
          cout<<"\n";
          //Объявляем целочисленные константы
          int DayIn_2000Year=366;
          int HourInDay=24;
         //объявляем целочисленную переменную
        int  HourIn_Year2000;
        // вычисляем искомое значение и
        // помещаем его в переменную HourIn_Year2000
        HourIn_Year2000=DayIn_2000Year*HourInDay;
        // выводим значение переменной HourIn_Year2000 на экран
        cout<<"\t\t In 2000 year "<< HourIn_Year2000;
        cout<<" hours\n ";
}
```

Tamam! Proqramı kompilyasiya edin!

Cismi dəyişənlər və konstantlar.
Elan və inisializasiya nümunə

```
float Weight;
Weight=12.3452;
double weight_atom;
weight_atom= 2.5194e+017;
```

2.5194e+017 rəqəmi nə deməkdir?

Bu cismi ədədlərin qısa yazılışıdır. Bu - eksponesial rəqəm yazılışı forması adlanır. Sizə yazılanların şifrəsinin açılmasının sirrlərini deyir. Bu simvol yığması ilə 251940000000000000 və ya 2,1594×1017 rəqəmi təsvir olunur.

-3.4E008 şifrin açması belədir: 3,4×10-8, bu da 3,4:108 - ya analojidir.

-1.5E+003 açması belədir - 1,5×103 .

float tipli üzən nöqtəli rəqəmlər -3,4×1038 və 3,4×1038 arasında dəyişə bilərlər.

-3,4×-38 və 3,4×10-38 arasındakı qiymətlər sıfıra bərabər hesab edilir.

İndi isə əlin cismi ədədlərlə praktikada işləyək:

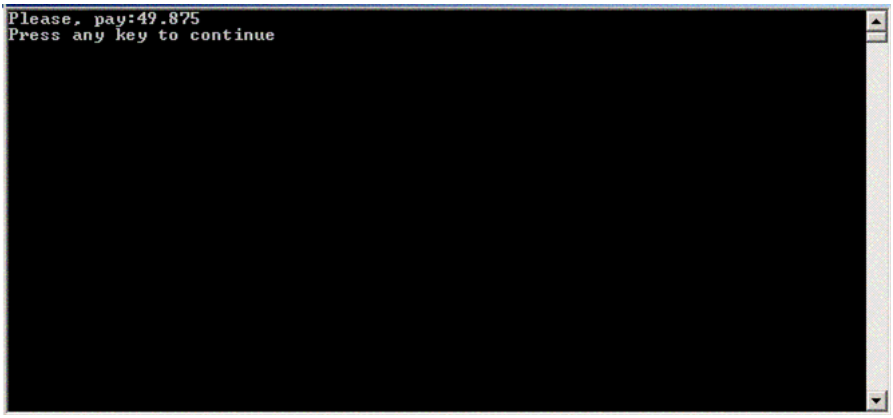
Alış-verişin dəyərini hesablaya bilən proqram yazaq. Qoy proqram malın qiymətini (Cost), alınan malın sayını (Count), endirimi də nəzərə alaraq(Discount) qiymətin dəyərini (Price) hesablasın.

Yeni **Bazarliq** adlı proekt yaradaq və növbəti proqramın mətnini yazaq.

```
// Заголовок
#include  <iostream>
// определение пространства имен, в котором есть cout<<
using namespace std;
// Главная функция
void main()
{
          //Объявляем переменную Discount
          float Discount=0.05;
          //Объявляем переменную Cost
          float Cost=10.50;
          //Объявляем переменную Count
          int Count=5;
          //Объявляем переменную Price
          float Price;
           //Вычисляем значение переменной Price
            Price=Count*Cost-Count*Cost*Discount;
           // Выводим итоговую стоимость товара со скидкой
           cout<<"Please, pay:"<<Price<<"\n";
}
```

Proqramı kompilyasiya edin və icra olunmağa yolayın.,

Ekranda görməli olduğunuz aşağıda göstərlib:



Simvollu və məntiqi dəyişənlər və konstantlar.

Bu dərsimizdə simvollu və məntiqi dəyişənlər və konstantlara nümunələr göstərməyəcəyik. Onları gələcək dərslərdə daha geniş öyrənəcəyik. Yalnız elan edilməsi və inisializasiyasından danışaq.

```
// Логическая переменная
bool Flag;
Flag=true;
// Один символ всегда указывается в одинарных кавычках
char Symbol='A';
/* Escape - последовательность рассматривается компилятором, как один символ
     и соответственно может быть записана в переменную или константу
     типа char*/
const char NewLine='\n';
cout<<NewLine// показывает пустую строку
```

8. Məlumatın daxil edilməsi

Biz artıq kompyuterin ekranına məlumatların çıxarılması əməliyyatı ilə - cout tanışıq. Lakin, proqramların çoxunda nəinki çıxarış, həm də klaviaturadan məlumatların daxil edilməsi imkanı da tələb olunur. Bundan öncəki bülümdə endirimin hesablanması proqramı göstərilmişdi. Əlbəttə ki, malın sayı və qiyməti kimi parametrlərin, proqamın icra olunduğu mərhələdə klaviaturadan daxil edilməsi pis olmazdı.Gəlin baxaq görək, bunu necə edə bilərsiz.

Əgər bizə məlumatları kompyuterə daxil etmək lazımdırsa onda cin komandasından istifadə edək. Ondan necə istifadə olunur? Daxiletmə operatorunun sintaksisi:

```
cin>>имя_переменной;
```

dəyişənin_adı laviatkuradan daxil edilən məlumatların hansı dəyişənə yerləşdirilməsinə işarə edir:
Məsələn:

```
cin>>Age;
```

Bu komanda Age adlıdəyişənə klaviaturadan daxil edilmiş rəqəmi yerləşdirir. Rəqəmin Number adlı dəyişənə daxil edilməsi üçün, sadəcə belə komandanı yığmaq lazımdır:

```
cin>>Number;
```

Hamısı bir yerdə yazılan dəyişənlərə nümunə:

```
cin>>имя_переменной1>>имя_переменной2>>...>>имя_переменнойN;
```

Dəyişənlərin adlarının siyahısında bütün dəyişənlər olmalıdır, hansılara ki, siz klaviaturadan məlumatları daxil etməki istəyirsiniz. Dəyişənlərin adlarının siyahısı simvollar kombinasiyasına bölünmüş dəyişən adları sayından ibarət ola bilər.

```
cin>>Quantity>>Price>>Discount;
```

Gəlin Bazarliq proqramına məlumatları klaviaturadan daxil edək:

```
//Заголовок
#include  <iostream>
// определение пространства имен, в котором есть cout<<
using namespace std;
// Главная функция
void main()
{
            //Объявляем переменную Discount
            float Discount=0.05;
            //Объявляем переменную Cost
            float Cost=10.50;
            //Приглашение ввести цену товара
             cout<<"What's the cost?\n";
            //Ввод значения в переменную Cost
            cin>>Cost;
            //Объявляем переменную Count
            int Count=5;
            //Приглашение ввести количество
            cout<<"How much?\n";
            // Ввод значения в переменную Count
            cin>>Count;
            //Объявляем переменную Price
            float Price;
            //Вычисляем значение переменной Price
```

```
            Price=Count*Cost-Count*Cost*Discount;
            // Выводим итоговую стоимость товара со скидкой
            cout<<"Please, pay:"<<Price<<"\n";
}
```

İndi siz cin>> operatorunun işinin əsasını gördüz. Proqram bu operatoru görən kimi dayanır, və istfi-adəçinin reaksiyasını - istifadəçi məlumatları daxil edib Enter düyməsini basanadək gözləyir. İcra yalnız bundan sonra davam edir.

Misal üçün, yenidən daxil etmə/çıxarışla işləyək. Yalançı - proqram yazaq: proqram rəqəmlərlə bağlı oyun təqdim edir, kim çox rəqəm desə o da udur.

Yeni **Game** proekt yaradaq və belə bir mətn yazaq:

```
// Заголовок
#include  <iostream>
// определение пространства имен, в котором есть cout<<
using namespace std;
// Главная функция
void main()
{
            // Приглашение "Давай играть!"
            cout<<"Let's play!\n";
            //Объявление переменной i
             int i;
            //Приглашение "Введите число"
             cout<<"Enter a number:";
            //Ввод числа
             cin>>i;
            //Вывод числа, которое "загадал" компьютер
            cout<<"I have "<<i+1<<"\n";
            // Вывод результата игры
            cout<<"I'm winner!\n";
}
```

Proqramı kompilyasiya edin. Siz sadəcə rəqəm yazırsız, və nəticədə aydın olur ki, proqramın tutduğu rəqəm sizinkindən çox olur, və o həmişə udur.

Proqramı qoşduğunuz zaman daxil etdiyiniz rəqəm 67 olduğu təqdirdə görəcəyiniz:

```
Let's play!
Enter a number: 67
I have 68
I'm winner!
Press any key to continue...
```

Niyə axı o həmişə udur?

Gəlin bu sətrə nəzər salaq:

```
cout<<"I have "<<i+1<<"\n";
```

Burada i dəyişkənin qiyməti çıxarılır, siz onu klaviaturadan 1 dənə çox kimi yazmısız, yəni kompyuter həmişə bir 1 rəqəm çox olan rəqəm çıxarır.

Əgər i+1 ifadəsini i-1 ilə əvəzləsək, onda həmişə siz udacaqsız, belə ki, kompyuterin çıxardığı rəqəm, sizin klaviaturadan yazdığınız rəqəmdən 1 rəqəm az olacaq.

Son olaraq, sizi +(plus) və -(minus) operatorlarına diqqət yetirməyə çağırırıq. Onlar toplama və çıxma üçün istifadə olunurlar. C dilində həmçinin -/ bölmə operatoru da vardır. Bu informasiya sizə gələcəkdə ev tapşırıqlarını edərkən kömək edəcəkdir, operatorlar barədə ətrafli isə növbəti dərslərimizdə danışacayıq.

9. Literallar

Literallar (literals) - kompyuterin dəyişmək halında ola bilmədiyi, fiksə edilmiş qiymətlərdir. C dilinin hər tipi üçün literallar mövcuddur, simvollu və bul tipləri, tam ədədlər, üzən nöqtəli ədədlər kimi tiplər də daxil olmaqla. Paradoksal olsa da, C - də sətrlərin saxlanması üçün məlumat tipləri yoxdur, sətri literallar isə mövcuddur.

Bəzi misallar:

5	tamrəqəmli literallar-int
5l	l və ya L long deməkdir+
true	məntiqi literal boolllüzən nöqtəli literal,
5.0	double kimi başa düşülürüüf və ya F -
5.0f	üzən nöqtəli, float kimi başa düşülürff
0.3e-2	double üzən nöqtəli literal, e və ya E ek sponensial hissəni ayırırddsimvol literalıss
'd'	simvollu literal
"Visual"	sətri literal - bu dirnaq arasına alınmış ixtiyari simvol yığımıdır. Kompilyator onu məhz simvol yığımı kimi qəbul edir və onu emal etməyə çalışmır, əgər, dırnaq arasında hansısa açar sözləri və əməliyyatlar olarsa belə.

Tərkibində literal olan kod nümunəsi

```
//"abrakadabra" - строковый литерал '\n' - символьный литерал
cout<<"abrakadabra"<<'\n';
int a = 2; // 2 - литерал типа int
```

Birinci dərsin materialının izah olunmasını burada bitirilmiş hesab etmək olar. Növbəti bölümdə qeyd edilmiş ev tapşırığını yerinə yetirməyi məsləhət görürük. Sizə uğurlar!