ЎЗБЕКИСТОН РЕСПУБЛИКАСИ ОЛИЙ ВА ЎРТА МАХСУС ТАЪЛИМ ВАЗИРЛИГИ

БУХОРО ДАВЛАТ УНИВЕРСИТЕТИ

Физика-математика факультети
“Амалий математика ва ахборот технологиялари” кафедраси

5А480103 -“Амалий математика ва ахборот технологиялари” таълим йўналиши бўйича магистрлик даражасини олиш учун

Қаюмова Гулшан Асроровна

“PHP ва MYSQL ёрдамида университет талабалари билимини текширувчи интерактив дастурни яратиш”.

МАГИСТРЛИК ДИССЕРТАЦИЯСИ

Илмий раҳбар _____ _____ *Сайидова Н.С.*
“_____” _____ __2011 й.

БМИ “Амалий математика ва ахборот технологиялари” кафедрасининг 20___ йил _____ № ____ сонли йиғилишида кўриб чиқилди ва ҳимояга рухсат берилди.

Кафедра мудири _____ *Расулов И.Г.* **Тақризчи _____*Обидов К.***

«_____» _____2011 й. **“_____” _____ 2011 й.**

“Ҳимоя қилишга рухсат берилди”

Укув булими бошлиги _____ _____.

“_____” _____ _____ 20_____ й.

Бухоро-2011

МУНДАРИЖА

Ватанимизнинг келажаги, халқимизнинг эртанги куни мамлакатимизнинг жаҳон ҳамжамиятидаги обру-эътибори аввалambор фарзандларимизнинг униб-ўсиб, улғайиб, қандай инсон бўлиб, ҳаётга кириб боришига боғлиқдир. Биз бундай ўткир ҳақиқатни ҳеч қачон унутмаслигимиз керак.

И. А. КАРИМОВ

КИРИШ

Президентимиз Ислом Каримов раҳнамолигида юртимизда узлуксиз таьлим тизими жорий этилиб, таьлим муассасалари моддий-техник базасининг мустаҳкамлангани ҳамда педагогларнинг ҳар томонлама рағбатлантирилаётгани юксак интеллектуал салоҳиятли ва маьнавиятли авлодни тарбиялашнинг муҳим омили бўлаётгани таькидланди. Бу борадаги ишлар “Кичик бизнёс ва тадбиркорлик” йилида янада такомиллаштирилмоқда.

Бир қатор ўқув дастурлари ва дарсликлар такомиллаштирилгани, айрим фанлар бўйича қайта ишлаб чиқилган методик тавсиялар амалиётга жорий этилгани шулар жумласидандир. Миллий ва халқаро тажриба асосида ўқувчилар билимини баҳолашнинг янги механизми ўқув жараёнига татбиқ этилди.

Президентимиз Ислом Каримовнинг “Юксак маьнавият – енгилмас куч” асарида бугунги мураккаб глобаллашув даврида ёшлар маьнавиятини асраш ва юксалтириш, уларни зарарли ғоялар таьсиридан ҳимоялаш, маьнавий-мафкуравий иммунитетини мустаҳкамлаш масаласи долзарб аҳамият касб этаётгани алоҳида таькидланган.

Ёшларни мустақил фикр, юксак маьнавият, теран тафаккур соҳиблари этиб камолга етказиш масаласи давлатимиз сиёсатининг устувор йўналишларидандир. Таьлим-тарбия, соғлиқни сақлаш, ижтимоий ҳимоя,

маънавий-маърифий соҳаларда изчил, босқичма-босқич олиб борилаётган ислоҳотлар ушбу мақсадга йўналтирилган.

Юртимизда таълим ва ёшларни маънавий-ахлоқий тарбиялаш соҳасида ахборот-коммуникация технологиялари ҳамда хизматларни ривожлантиришда аниқ мақсадга йўналтирилган изчил чора-тадбирлар олиб бориляпти. Айни пайтда таълимнинг барча бўғинларида замонавий компьютер хоналари ҳамда ахборот-ресурс марказларини ташкил этишга, бунда ўқитувчиларнинг малака ва маҳорати, талаба-ёшларнинг билим ва кўникмаларини янада юксалтиришга алоҳида эътибор қаратилмоқда.

Бугунги кунда юртимиздаги барча олий ўқув юртлари ягона ZiyoNET таълим ахборот тармоғига бирлаштирилиб, ундан тизимда самарали фойдаланиш йўлга қўйилган. Унга ҳар бир олий таълим даргоҳининг ўз йўналишига оид маълумотлар, фанлар бўйича маъруза матнлари, илмий-ўқув адабиётлар, бакалавр ва магистрларнинг энг яхши илмий изланишлари киритилган. Шу билан бирга порталнинг асосий бўлинмалари орқали фан ва таълимга оид янгиликлар, шарҳлар, эълонлар ва видео материаллардан баҳраманд бўлиш, электрон, мультимедиа дарсликлари, мақолалар билан танишиш, олий ўқув юртлариаро ўзаро илмий-маърифий ҳамкорликни йўлга қўйиш каби кенг имкониятлар мавжуд.

Магистрлик диссертацияси мавзунинг долзарблиги: PHP ва MySQL ёрдамида университет талабалари билимини текширувчи интерактив дастур яратиш.

Магистрлик диссертацияси ишининг мақсади ва вазифалари: талабалар билимини текширувчи ва назорат қилувчи PHP WEB дастурлаш тили ва MySQL маълумотлар омборини бошқариш тизимидан фойдаланган ҳолда WEB дастур яратиш.

Магистрлик диссертацияси амалий аҳамияти: талаба махсус WEB дастурдан фойдаланган ҳолда турли хил фанларга оид тестларга жавоб

бериш, назорат ишларини бажариш имкониятига эга бўлади. Фан ўқитувчилари талабалар билимини назорат қилиб боради.

Магистрлик диссертацияси илмий янгилиги: талабалар ўз билимини юқори даражада тизимлаштирилган дастурдан фойдаланган ҳолда текшириш имкониятига эга бўлишади.

Магистрлик диссертацияси тадқиқот объекти ва предмети: PHP, MySQL, HTML WEB - ҳужжатларни яратиш тили, тестлар, WEB – технологиялар.

Магистрлик диссертацияси ишнинг ҳажми ва тузилиши: кириш, 2 боб, 7 бўлим, 2 бобнинг хулосаси, хотима, фойдаланилган адабиётлар рўйхати, 6 та жадвал, 6 та расм ва 92 бетдан иборат.

I БОБ. ИНТЕРНЕТ ТЕХНОЛОГИЯЛАРИ ОЛАМИГА САЁҲАТ.

1.1. ИНТЕРНЕТ – ҲАЛҚАРО ГЛОБАЛ ТАРМОҒИ.

Глобал тармоклар.

Глобал Компьютер тармоқлари турли мамлакатлар ёки қитъаларда жойлашган абонентларни бирлаштиради. Мазкур тармоқ абонентлар ўртасидаги алоқа телефон, радио алоқа ва космос алоқа тизими негизида амалга оширилади.

Глобал, минтақавий ва локал компьютер тармоқларининг бирлашуви кўп тармоқли иерархияни ташкил этиб, умумжаҳон ахборот ресурсларини бирлаштириш ва улардан коллектив равишда фойдаланиш имкониятларини яратади .

Компьютер - Интернет билан ишлашни таьминловчи воситадир. Компьютер бу ҳисоблаш машинасидир.

Интернетнинг таркибий қисмлари.

Интернет алоҳида компьютерлар ўртасида алоқа ўрнатибгина қолмай, балки компьютерлар гуруҳини ўзаро бирлаштириш имконини ҳам беради. Агар биронбир маҳаллий тармоқ бевосита интернетга уланган бўлса, у ҳолда мазкур тармоғнинг ҳар бир ишчи станцияси Интернетга уланиши мумкин. Шунингдек, интернетга мустақил равишда уланган компьютерлар ҳам мавжуд. Уларни хост компьютерлар (host — раҳбар) деб аташади. Тармоққа уланган ҳар бир компьютер ўз адресига эга ва унинг ёрдамида жаҳоннинг исталган нуқтасидаги исталган мижоз уни топа олиши мумкин.

Интернет - бу интернет технологияси, дастур таъминоти ва протоколлари асосида ташкил этилган, ҳамда маълумотлар базаси ва электрон ҳужжатлар билан коллектив равишда ишлаш имконини берувчи корхона ёки концерн миқёсидаги ягона информацион муҳитни ташкил этувчи компьютер тармоғидир.

Интернет бошқа компьютер тармоқларидан қуйидаги билан фарқланади. Бир ёки бир неча серверлардан ташкил этилган тармоқ мижози ундаги электрон ҳужжат, маълумотлар базаси ва файллардан фойдаланиш учун, уларнинг қайси серверда, қайси каталогда қандай ном билан сақланганлигини, уларга кириш усул ва шартларини билиши зарур бўлади.

Сервер - бу бошқа компьютер ва дастурчиларга хизмат кўрсатадиган компьютер ёки дастурдир. Яъни бошқа компьютерларга ўзининг файлларидан фойдаланишга рухсат берувчи компьютер Сервер ҳисобланади. Битта компьютерда бирнеча Сервер ишлаши мумкин.

Интернетда эса бундай ноқулайликларни олди олинган бўлиб, унинг фойдаланувчиси бундай маълумотларни билиши шарт эмас. Бундан ташқари интернет тармоғида мавжуд бўлган барча электрон ҳужжат ва маълумотлар базасини гипер боғланишлар ёрдамида ўзаро боғлаб ягона информацион муҳит қуриш, унда қулай информацион қидирув тизимларини ташкил этиш мумкин бўлади.

Интернет ўз - ўзини шакллантирувчи ва бошқарувчи мураккаб тизим бўлиб, асосан учта таркибий қисмдан ташкил топгандир:

- техник,
- дастурий,
- информацион.

Интернетнинг техник таркибий қисми ҳар хил турдаги ва типдаги компьютерлар, алоқа каналлари (телефон, спутник, шиша толали ва бошқа турдаги тармоқ каналлари), ҳамда тармоқ техник воситалари мажмуидан ташкил топгандир. Интернетнинг ушбу техник воситаларининг барчаси доимий ва вақтинчалик асосда фаолият кўрсатиши мумкин. Улардан ихтиёрий бирининг вақтинчалик ишдан чиқиши Интернет тармоғининг умумий фаолиятига асло таьсир этмайди.

Протоколлар, мижозлар ва серверлар.

Протокол - бу компьютерлар орасидаги алоқа ўрнатилишида, маълумотларни қабул қилиш ва узатишда фойдаланиладиган сигналлар стандартидир. Яъни компьютерлар протокол ёрдамида бири - бири билан боғланади. Протокол тўғри бўлсагина, компьютерлар ўртасида алоқа ўрнатилади. Бу компьютерларнинг боғланиш тартиби ёки стандартидир.

Сервер - бу бошқа компьютер ёки программаларга хизмат кўрсатадиган компьютер ёки программадир. Яъни бошқа компьютерларга ўзининг файлларидан фойдаланишга рухсат берувчи компьютер Сервер ҳисобланади. Битта компьютерда бир нечта сервер ишлаши мумкин. Масалан, FTP, WWW, электрон почта серверлари.

Мижоз - Сервер ресурсларидан ва хизматидан фойдаланувчи компьютер ёки программадир. Худди Сервер каби, битта компьютерда бирданига бир нечта мижоз ишлаши мумкин. Масалан, компьютер файл серверининг мижози бўлиши мумкин (серверда жойлашган файллардан фойдаланиши), шу билан бир вақтда электрон почта программасида ишлаши мумкин. Яъни бир неча серверининг мижози бўлиши мумкин.

Шлюз - протоколни бир турдаги муҳитдан иккинчи турдаги муҳитга ўтказувчи тармоқ қурилмаси. Масалан, компьютер Интернетга боғланганда шлюздан фойдаланилади.

Proxy - бир неча компьютернинг Интернетга уланишини таъминловчи тизим. Proxy сервер одатда кўп ишлатиладиган ресурсларни сақлаш имкониятига эга.

URL(Uniform Resource Locator) - Internetra мурожаат қилишнинг энг оддий ва қулай усули бўлиб, у манзилни ифодалайди. URL адресидан ихтиёрий фойдаланувчи фойдаланиши мумкин. Яъни бу адресдаги маълумотдан барча фойдаланувчилар бир пайтнинг ўзида фойдаланиши мумкин.

IP ва URL Адреслар тушунчаси

Амалиётда интернетнинг реал, физик боғланишлар орқали ташкил топган тармоғидаги компьютерлар билан виртуал информацион фазони ташкил этувчи электрон ҳужжатлари ҳар хил адреслар ёрдамида ифодаланилади. Интернет таркибига кирган ҳар бир компьютер тўрт қисмдан ташкил топган ўз адресига эга, масалан: 142.26.137.07. Ушбу манзил IP (Internet Protocol) манзил деб аталади. Интернетга доимий уланган компьютерлар ўзгармас IPадресга эга бўлади. Агар компьютер фойдаланувчиси интернетга фақат вақтинчалик ишлаш учун уланадиган бўлса, у ҳолда ушбу компьютер вақтинчалик IPадресга эга бўлади. Бундай IP манзил динамик IP манзил деб аталади.

Тармоқда мавжуд бўлган ихтиёрий компьютер IP адресини билган ҳолда, унга ҳар хил кўринишдаги сўровлар билан мурожаат қилиши мумкин бўлади. Бу сўровлар ўша компьютерда сақланаётган электрон ҳужжатлар, маълумотлар базаси, ёки бўлмаса ундаги бирор бир программани ишлатишга, ўша компьютер таркибига кирган техник ресурслар имкониятидан фойдаланишга оид бўлиши мумкин ва ҳоказо.

Интернет информацион муҳитини ташкил этувчи электрон ҳужжатларнинг ҳар бири компьютерларнинг IPадресларидан бошқа ўзларининг такрорланмас, уникал адресларига эга. Бу адрес URL (Uniform Resource Locator) адрес деб аталади. Масалан, Ўзбекистон Республикаси ҳукуматининг расмий ахборотлари, Олий мажлис қарорлари ҳақида маълумот берувчи электрон саҳифа адреси **www.gov.uz**

Агар Интернет тармоғида бирор бир ҳужжат эълон қилинган бўлса, у ягона такрорланмас URLадресга эга. Компьютерда бир ном билан иккита файл мавжуд бўлмаганидек, интернетда ҳам икки электрон ҳужжат бир хил URL адресга эга бўлмайди.

WORLD WIDE WEB (Жаҳон ахборот тармоғи)

WWW Интернетнинг энг оммалашган ахборот хизматларидан бири саналади. Ҳозирги вақтда интернет хизматининг 90 % га яқинини WWW хизмати ташкил этади. Интернетга асос солингандан бошлаб (1969 йил) WWW хизмати ташкил этилгунга қадар интернет секин ривожланди ва 25 йил давомида бор йўғи 2 миллионга яқин фойдаланувчига эга эди холос. WWW хизмати ташкил этилгандан сўнг эса (1996 йил), ҳар ярим йилда интернет фойдаланувчиларининг сони 1,5 баробарга ортиб борди. Бугунги кунда интернет тармоғининг фойдаланувчилар сони 300 миллионга етди.

WWW хизматининг асосий тушунчалари:

HTML формати;

"Гиперматн" боғланиши;

HTTP "гиперматн" узатиш протоколи;

Web ҳужжатлар;

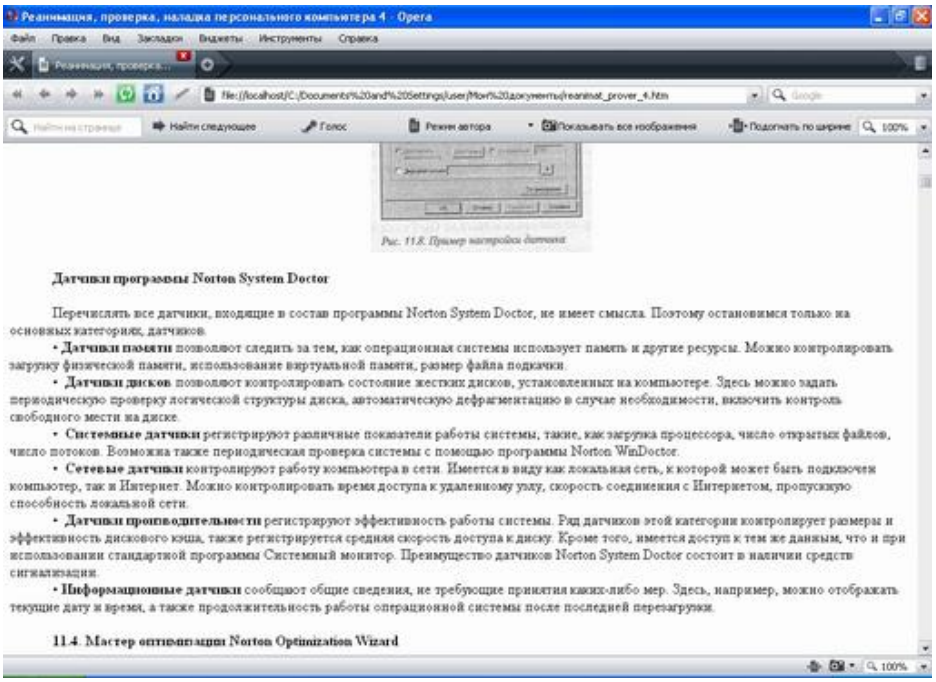
Web узел ва сайтлар;

Web саҳифаларнинг актив компонентлари.

1.2. Web браузерлар.

Браузер, шарҳловчи(кузатиб борувчи), навигатор (инглиз тилидан browser – китоб варақловчи инсон маъносини англатади) – объектларнинг, масалан веб саҳифаларнинг визуаллаштирилишини кўриш учун мўлжалланган дастурдир. Бугунги кунда жуда кўп миқдорда браузерлар мавжуд бўлиб, улар ихтиёрий платформалар(амалиёт тизимлар) учун ёзилгандир. Бошқача қилиб айтганда, **Веб-шарҳловчи** ёки **бра́узер** — бу веб-сайтларни кўриш, яъни веб саҳифаларни қайта ишлаш, чиқариш, бир саҳифадан бошқасига ўтиш мақсадида уларни «Ўргимчак тўридан» сўров қилиш учун мўлжалланган дастурий таъминотдир. Кўпгина браузерлар FTP-серверларнинг мундарижаларини кўриш хусусиятига эгадир. Браузерлар «Бутун дунё ўргимчак тўри» пайдо бўлгандан буён ривожлана бориб, унинг

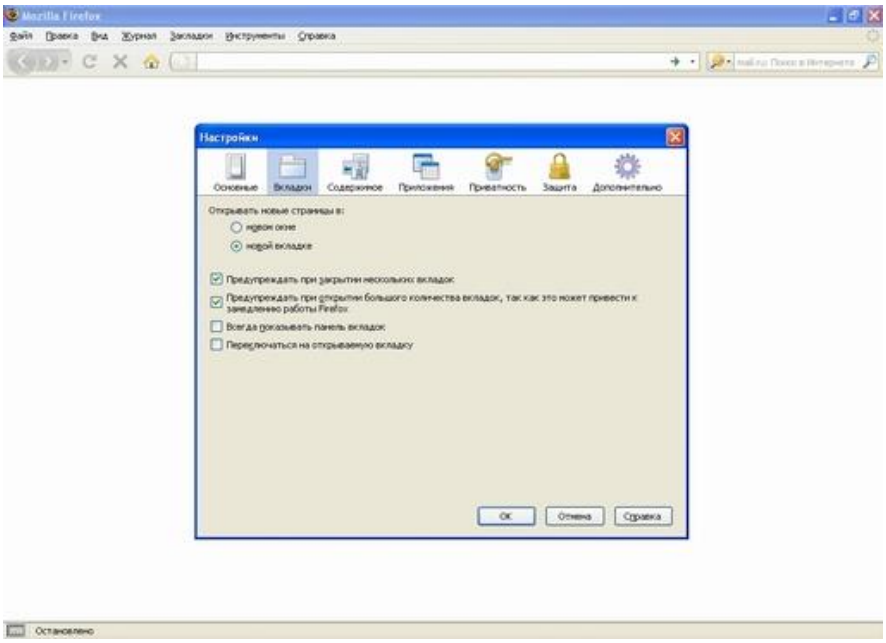
ўсиши натижасида шахсий компьютер учун муҳим дастурга айланган. Ҳозирда браузер — веб-саҳифаларининг турли ташкил қилувчиларини қайта ишлаш ва чиқариш ҳамда веб-сайтлар ва уларга ташриф буюрувчилар ўртасида интерфейсни яратиб берувчи комплекс илова ҳисобланади. Амалий жихатдан барча оммабоп браузерлар текинга ёки бошқа иловаларга қўшиб тарқатилади, масалан: Internet Explorer (Windows нинг қисми сифатида), Mozilla Firefox (эркин дастурий таъминот), Opera (8.50 версиясидан бошлаб текин), Safari (Mac OS нинг қисми сифатида).



1 – расм. Opera 9 web браузери.

Биринчи кенг тарқалган график интерфейсли браузер бўлиб NCSA Mosaic ҳисобланган, сўнгра узоқ вақтгача Netscape Navigator бозорни тарк этмаган. 1995 йилда Microsoft компанияси Internet Explorer 3.0 ни ўз ичига олган Windows 95 AT ни ишлаб чиқарди. Бу эса «браузерлар уруши» бошланишига асос бўлди. Натижада Netscape талофатга учради, Internet Explorer эса бозорнинг 95 фоиздан ортиғини эгаллади. Талофатга учраётган

Netscape эса MPL (Mozilla Public License) эркин лицензиясида ўз браузерининг дастлабки кодини чиқарди ва унинг асосида янги Mozilla ва Mozilla Firefox браузерлари яратилди ҳамда секин-аста оммабоп бўла борди. 2005 йилда Opera браузери ҳам текинга тарқатила бошланди.



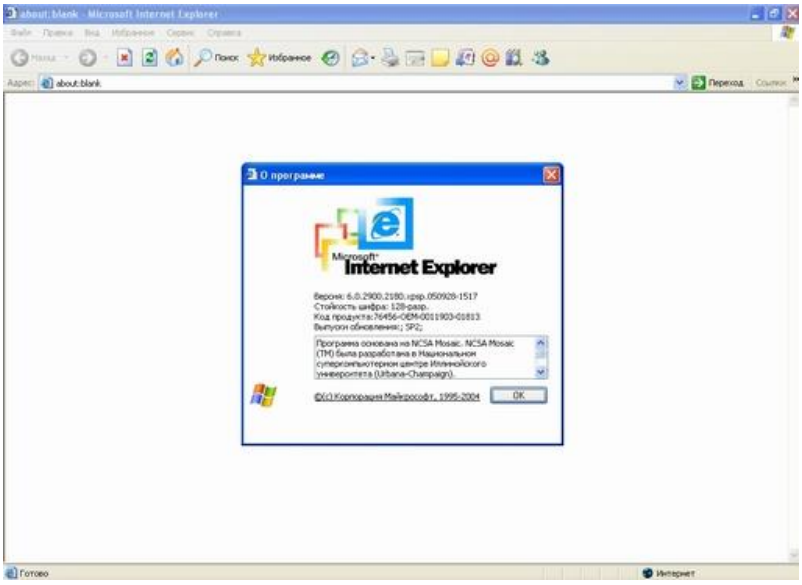
2 – расм. Mozilla Firefox 3 web браузери

Агар курашда асосий усул бўлиб браузерларга специфик ва ностандарт имкониятларни қўшиш бўлмаганда, браузерлар орасидаги уруш корпорацияларнинг тижорат ишлари бўлибгина ҳисобланар эди. Кўп тафовутлар хужжатларга интерактивликни берувчи сценарий тили ҳисобланган Javascript ни қўллашда юзага келди. Натижада кўп хужжатлар аниқ бир браузер учун оптималлашиб, бошқаларда умуман ўқилмади.

WWW-Консорциум кўпгина пухта ишланган стандартларни қабул қилади (HTML, Javascript, CSS ларнинг турли лахжалари ва бошқалар), лекин бу стандартларга амал қилиш бутунлай браузер ишлаб чиқарувчиларига юклатилди. Охирги йилларда стандартларни қўллаш даражаси анча ўсди ва замонавий браузерлардан фақат Internet Explorer (2001

йилда чиққан олтинчи лахжаси) стандартларга риоя қилишда муҳим камчиликларга эгадир (2006 йил 18 октябрда чиққан еттинчи лахжасини стандартларга мос келиши тўлиқ текширилмаган).

Windows (Microsoft) оиласидаги амалиёт тизимларининг локаллаштирилган лахжаларида браузерлар шунчаки *шарҳловчилар*, *тармоқ шарҳловчилари* ёки *веб-шарҳловчилар* деб номланади.



3 – расм. Microsoft Internet Explorer 6 web браузери

Браузерларнинг қўлланилиши ва уларнинг характеристикаси хақида.

Браузерларнинг оммабоплиги ва уларнинг қўлланилиш соҳалари устида сўз юритилар экан, бунда қуйидаги тақсимотни келтириш мумкин:

Оммабоп браузерлар:

- Opera;
- Mozilla Firefox;
- Flock;
- Internet Explorer;

- Maxthon — Internet Explorer учун дастурий қобиқдир;
- Safari — Konqueror кодига асосланган.

Камроқ тарқалган браузерлар:

- Netscape Navigator;
- Konqueror;
- Galeon;
- Epiphany;
- Kazehakase;
- Charon;
- Arachne;
- K-Meleon.

Матнли браузерлар:

- Lynx;
- Links;
- W3M;
- Netrik;
- Elinks;
- Internet Browser.

Мобиль телефонлар учун браузерлар:

- Opera Mini – ўз бозорида абсолют лидер.

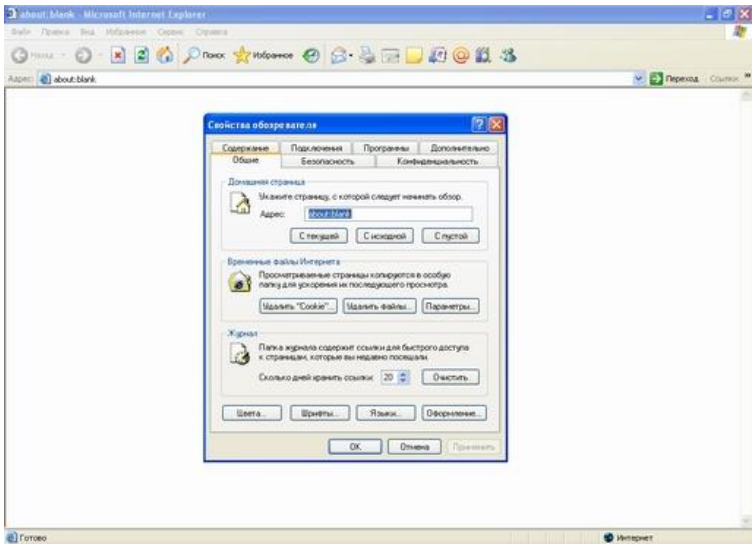
Internet Explorer дастури билан ишлаш.

Ҳозирги кунда интернетнинг WWW хизмати кундан - кунга ривожланиб мукаммал маълумотлар манбаасига айланиб бормоқда. Унинг ёрдамида исталган соҳада, исталган мавзуда ва исталган вақтда маълумотларни қидириб топиш, улардан фойдаланиш, зарур бўлса улардан нусхалар олиш мумкин. Интернетнинг ушбу хизмат туридан фойдаланиш учун аввало мижоз компьютерида худди шундай имкониятларни яратиб

берувчи махсус программа таъминоти бўлиши зарур. Бундай программа таъминоти броузерлар (browsers) деб аталади.

Энг биринчи броузер CERN (Европа Физика Тадқиқотлари Маркази) ходими Тим Бернер томонидан кашф қилинган. Энг биринчи график маълумотларни экранда акс эттирувчи броузер Mosaic Американинг NSCA (Миллий Супер Ҳисоблаш Маркази)да Марк Андриссон ва бир неча талабалар томонидан ишлаб чиқилган. Броузер бу инглизча сўз бўлиб кўришни таъминлаш, яъни кўрсатиш маъносини англатади. Дунёдаги энг кўп фойдаланиладиган броузерлар Netscape Communication ва Internet Explorer ҳисобланади.

Бугунги кунда Netscape ҳамда Internet Explorer дан ташқари яна кўплаб Opera, FMSD Friadna, MS ICE, Webcelerator, AtGuard, AdWiper каби броузерлар мавжуддир. Броузерларга қўйиладиган асосий талаблардан бири, бу интернетнинг WWW хизматидаги маълумотлар жойлашган веб саҳифаларини, қайси технология ёрдамида ишлашидан, ҳамда қайси программалаш тилида ёзилганидан қатъий назар, ундан тўлиқ фойдаланиш имкониятларини яратиб беришдир. Бу талабга ҳамма броузерлар ҳам жавоб бера олмайди.



4 – расм. Microsoft Internet Explorer 6 web браузерини созлаш ойнаси.

Internet Explorer броузери эса ихтиёрий веб саҳифани ҳеч қандай муаммоларсиз кўриш ва ундан тўлиқ фойдаланиш имкониятини яратиб беради. Opera броузерининг мухлислари эса уни жуда ҳам ихчам ҳажмда эканлиги учун яхши кўришади. Чунки бу броузер компьютер ташқи хотирасида атига 2 Mb гина жойни эгаллайди холос. Унинг жуда ҳам тезкор ишлаши ва кўплаб Netscape ишлайдиган plugin лар Macromedia Flash, Acrobat Reader, Cosmo playerларни ўзида акс эттира олиши унга бўлган қизиқишга сабаб бўлмоқда. Netscape нинг имкониятлари ҳам ўзига хосдир. У олинаётган информация ҳақидаги доимий маълумот, алоқа каналлари секин ишлайдиган жойларда жуда ҳам қўл келади. Бундан ташқари броузер ўзида почта хизматидан фойдаланиш йўлга қўйилганлиги жуда хам фойдаланувчиларга қулайдир. Яна программада интернетдаги бошқа фойдаланувчилар билан суҳбатлашишни таъминловчи программа ҳам ўрнатилган.

1.3. HTML – Web ҳужжатларни яратиш тили.

HTML нинг конструкцияси ТЭГ лар дейилади. Браузер ТЭГ ларни оддий матнлардан фарқлаши учун улар бурчак қавсларга олинадилар. ТЭГ тасвирлаш жараёни ҳатти ҳаракатларининг бошланишини билдиради. Агар бу ҳаракат бутун ҳужжатга таллуқли бўлса, бундай тэг ўзининг ёпилувчи жуфтига эга бўлмайди. Жуфт тэгларнинг иккинчиси биринчисининг ҳаракатини якунлайди. Масалан, ҳар бир Web саҳифа <*html*> тэги билан бошланиб </*html*> тэги билан ёпилиши керак. Эътибор берган бўлсангиз ёпилувчи тэг очилувчидан « / » белгиси билан фарқ қилади. Тэг номлари катта ёки кичик ҳарфлар билан ёзилиши мумкин, буларни браузер бир хил қабул қилади. HTML тилида бошқа компютер тилларидаги каби изоҳ бериш имконияти мавжуд. Изоҳ қуйидаги «<- - » ва «- ->» белгилар орсига ёзилади.

Масалан:

<-- *Бу изоҳ* -->

Ҳар қандай Web саҳифа иккита қисмдан ташкил топади. Булар сарлавҳа қисми ва асосий қисм. Сарлавҳа қисмида Web саҳифа ҳақидаги маьлумот жойлашади, асосий қисмда эса Web саҳифанинг мазмуни билан тасвирланиш қоидалари жойлашади. Сарлавҳа қисми қуйидаги очилувчи *<head>* ва ёпилувчи *</head>* тэглари орасида жойлашади. Асосий қисм эса *<body>* ва *</body>* тэглари орасида жойлашади. Одатда сарлавҳа қисми олдидан қўлланилаётган HTML стандартлари ҳақида маьлумот ёзилади. Ҳар қандай Web саҳифанинг умумий кўриниши қуйидагича бўлади:

Мисол:

```
<!DOCTYPE HTML PUBLIC  "-//W3C//DTD HTML 4.01//EN ">
<html>
  <head>
       <title>
    Ҳужжат сарлавҳаси
       </title>
  </head>
   <body>
    Асосий қисм
   </body>
</html>
```

Биринчи *<!Doctype>* тэги ўзининг параметрлари билан браузерга ушбу Web саҳифани қайси HTML версияда ёзилганлиги ҳақида маьлумот беради.

Web саҳифа ишга тушурилганда браузернинг энг юқори сатрида юкланаётган ҳужжат мазмунини англатувчи қисқача ёзув туради. Бу ёзувни ҳосил қилиш учун қуйидаги очилувчи *<title>* ва ёпилувчи *<title>* тэгларидан фодаланамиз.

Мисол:

```
<html>
 <head>
```

```
    <title>Web саҳифа  сарлавҳаси</title>
  </head>
<body>
</body>
</html>
```

Web саҳифанинг асосий қисми *<body>* ва *</body>* тэглари орасида жойлашади. Бу оддий матн бўлиши мумкин. Браузер бу матнни туғридан тўғри интерпретация қилиб экранда тасвирлайди. Бизга дастлабки Web саҳифамизни яратиш учун оддий «Блокнот» матн муҳаррири кифоя. Қуйида кўрсатилган мисолни матн муҳарририда ёзиб, уни хотирага ёзишда кенгайтмасини html ёки htm деб киритишимиз керак.

Мисол:

```
<html>
  <head>
    <title>Менинг биринчи Web саҳифам  </title>
  </head>
   <body>
    Менинг бу саҳифамга кирувчиларга алангали салом
   </body>
</html>
```

Бу файлни ишга тушириш учун сичқонча кўрсаткичини шу файл устига келтириб чап тугмасини икки марта босиш керак. Натижада экранда қуйидаги кўринишдаги натижа ҳосил бўлади:

<body> тэги бир қанча қўшимча параметрларга эга. Бу параметрлар тэгнинг очилувчи қисмида жойлашади. Параметрлар икки қисмдан иборат бўлади: параметр номи ва параметр қиймати. Масалан *bgcolor* параметри тасвирланаётган Web саҳифа фонининг рангини белгилайди.

Масалан:

<body bgcolor = “green”>

Параметрларнинг сатрли қийматлари қўштирноқ ичида ёзилади. Биз қуйида <*body*> тэгининг параметрлари билан танишамиз.

– *Background* - фон сифатида бирор бир график тасвирдан фойдаланиш. Параметр қиймати сифатида график тасвир жойлашган манзил (URL) берилади.

– *Text* - тасвирланаётган матн ранги.

– *Link* - Web саҳифадаги матнли гипермурожат ранги.

– *Vlink*-фойдаланувчи томонидан олдин мурожат қилинган гипер-мурожаат ранги.

– *Alink* - фойдаланувчи томонидан танланган гипермурожаат ранги.

– *Lang* – Web саҳифа матни ёзилган тилни аниқлаш.

Метамаълумотлар

Энди биз метамаълумотлар билан танишиб чиқамиз. Web саҳифаларда мета маълумотларини ҳосил қилиш учун <*meta*> тэги ишлатилади, унинг умумий кўриниши қуйидагича:

<meta name=”ўзгаручи номи“ content=”ўзгарувчи қиймати”>

Агарда биз Web саҳифадаги автор ҳақида маълумот ёзмоқчи бўсак уни қуйидаги кўринишда ёзиш мумкин:

<meta name =”Auther”content=”Бу меники!!!”>

Мета маълумотлар асосан Internet да жойлашган қидириш машиналари учун зарур. Қидириш машиналари Web саҳифалар ҳақидаги маълумотларни қаердан олади деган савол пайдо бўлади. Худди шу маълумотларини қидириш машиналари метаўзгарувчилардан олади. (Web саҳифа қайси соҳага тегишли, унда қандай маълумотлар борлиги) <*meta*> тэгида *keymards* ва *description* ўзгарувчиси бор. *Keymards* ўзгарувчили Web саҳифадаги калитли сўзлар рўйхатини ўзида сақлайди. *description* ўзгарувчи эса Web саҳифанинг қисқача маълумотини ўзида сақлайди. Масалан, бизнинг Web саҳифамиз компьютер вируслари ҳақида тайёрланган бўлсин у ҳолда HTML ҳужжатда мета маълумотларни қуйидагича ёзиш мумкин:

Мисол:

```
<html>
    <head>
    <title>компюьтер вируслари <title>
    <meta name=”keywords” lang=”ru” content=” вирус,
      компьтер, антивирус,…”>
    <meta name=”description” content=”Web саҳифадаги компьтер
    вирусларига бағишланган.
    </head>
    <body>
        компьютер вируслари бу ….
    <body>
</html>
```

Браузерлар фойдаланувчи томонидан очиб кўрилган Web саҳифаларини кэш хотирада сақлаб қолади. Агар фойдаланувчи яна шу саҳифаларга мурожаат қилса, Web браузер олдиндан кэш хотирада мавжуд бўлган (яна янги саҳифани интернетдан олмасдан) нусхасини олиб кўрсатади. Бу жараён фойдаланувчининг вақтини ва иқтисодини тежайди. Энди Web саҳифа қачон янгиланади деган савол пайдо бўлади. Бу саволга метамаълумотлардаги *expres* ўзгаручилардан жавоб олиш мумкин. Бу ўзгарувчида Web саҳифанинг яроқлилик муддати кўрсатилади. Агар кэш хотирадаги web саҳифа яроқлилик муддати ўтган бўлса, браузер тармоқдан Web саҳифани қайтадан ўқиб олади.

Мисол:

<meta http-equiv=”Expres” content=”Tue,uf Aug 2002 14:56:27 Gmt”>

Web саҳифаларда маълумотлар тез ўзгариши мумкин, масалан чатларда ва биржа саҳифаларида маълумотлар тез ўзгаради. Бундай ҳолларда *refresh* ўзгарувчисидан фойдаланилади ва қийматлари секундларда берилади.

Масалан:

<meta http-equiv=”Refresh” content=10>

Бу ёзувдан кейин Web саҳифа ҳар 10 секунддан кейин автоматик тарзда ўзи қайта юкланади.

Идентификаторлар.

HTML тилида ҳар бир қўлланилган тэгга уникал идинтификатор бериш имконияти мавжуд. Масалан матн бир неча абзацлардан ташкил топган бўлсин. Ҳар бир абзацга мос махсус ном бериш мумкин, кейинчалик HTML тилининг қўшимча имкониятлари ёрдамида бу абзацларни бошқариш мумкин, яъни уларнинг бирортасини кўринмас қилиш ёки шрифти рангини ўзгартириш.

Юқоридаги ишлар фақат абзацлар учун эмас, балки Web саҳифанинг ихтиёрий қисми учун таълуқлидир. Бирор бир тэгни номлаш учун *id* параметри ишлатилади. Абзацлар * * ва * * орқали кўрсатилади.

Мисол:

```
<html>
    <head><title>Абзацлар</title></head>
    <body>
    <p id =”p1”>  Биринчи абзац </p>
    <p id =”p2”>  Иккинчи абзац </p>
    </body>
</html>
```

HTML ҳужжатда *id* параметри қийматлари такрорланмаслиги лозим, акс ҳолда бу қийматлар эътиборга олинмайди.

class параметри фақат шакл безаш ишларида ишлатилади. Биз Web саҳифанинг айрим элементларини синфларга бўламиз, кейинчалик синфни тасвирлаш қоидалари ёзувини бир жойдан ўзгартиришимиз мумкин ва бу ўзгартириш автоматик равишда шу синфга кирган барча тэгларга тарқалади.

HTML ҳужжатини ташкил этувчи барча элементлар икки турга бўлинади: *inline* элементлар ва *блокли* элементлар. Inline элементлар оддий

матн элементлари бўлиб сатр қисми ҳам бўлиши мумкин, блокли элементлар эса ҳар сафар янги сатрдан бошланиши шарт. Блокли элементлар бошқа блокли элементлар ва Inline элементларидан ташкил топиши мумкин, лекин Inline элементлар блокли эелментларни ўз ичига олмайди. Web саҳифа элементларини блокларга бирлаштириш уларга бирваракайига шакл бериш имконини беради, яьни, бирлаштирувчи ягона тэгни аниқлаб блок жойлашувини ўзгартириш мумкин. Табиийки бу Web саҳифа элементларининг ҳар бирининг жойлашувини алоҳида ўзгартиришдан осон.

Блокли тип элементларини бирлаштириш учун <*div*> ва <*/div*> тэглари қўлланилади. Inline элементлари эса <*span*> ва <*/span*> тэглари орқали бирлаштирилади. Юқорида айтилганларга асосан <*div*> тэги <*span*> тэги ичида жойлаша олмайди.

Мисол:

```
<html>
  <head>
     <title>div блокни ҳосил қилиш</title>
  </head>
  <body>
       <div>
            Менинг бу <div> саҳифамга кирувчиларга </div> алангали салом
       </div>
  </body>
</html>
```

<*span*> ва <*div*> тэглари қўшимча параметрларни ҳам ўз ичига олиши мумкин**.** Бизга маьлум бўлган *id* ва *class* параметрларидан ташқари *style* ва *align* параметрлари ҳам ишлатилади. *style* параметри шу блокдаги маьлумот стилини ўрнатса, *align* параметри шу маьлумотни қандай текислашни аниқлайди. HTML ҳужжатида сарлавҳанинг ўз тэглари мавжуд бўлиб, улар олтитадир. Энг юқори даражаси бу биринчидир. Ҳар бир сарлавҳанинг ўз

теги ва ўз тасвирланиш қоидаси мавжуд. Энг катта яьни биринчи даражали сарлавҳа *<h1>* ва *</h1>* тэглари орқали, иккинчи даражали сарлавҳа *<h2>* ва *</h2>* тэглари орқали ва охирги олтинчи даражали сарлавҳа *<h6>* ва *</h6>* тэглари орқали ифодаланади. Қуйидаги мисолда биз сарлавҳалар тасвирини кўрамиз:

Мисол:

```
<html>
    <head>
    <title>Сарлавҳаларни тасвирлаш</title>
    </head>
    <body>
    <h1>Биринчи даражали сарлавҳа</h1>
    <h2>Иккинчи даражали сарлавҳа</h2>
    <h3>Учинчи даражали сарлавҳа</h3>
    <h4>Тўртинчи  даражали сарлавҳа</h4>
    <h5>Бешинчи даражали сарлавҳа</h5>
    <h6>Олтинчи даражали сарлавҳа</h6>
    <p>Оддий матн</p>
    </body>
</html>
```

Сарлавҳа тэглари ** ва * * тэглари каби *id, class, style, align* параметрларини ўз ичига олади.

Ишлатиладиган белгилар.

Компьютерда ҳар бир белги қандайдир сондан иборатдир. Операцион система ҳар бир сонга мос келувчи белгини экранда тасвирлайди. Сонларга мос келувчи белгилар жадвали кодировка (кодлаш) дейилади. Ҳозирнинг ўзида рус тили белгилари учун камида 5 та кодлаш усуллари мавжуд. Агар Web саҳифа яратилаётганда қайси кодлаш усулидан фойдаланилганни браузер аниқлай олмаса, экранда тушунарсиз белгилар пайдо бўлади. HTML

тили Web саҳифа қайси усулда кодланганини кўрсатиб туриш имкониятига эга. Бунинг учун <*meta*> тэги ишлатилади.

HTTP протоколида (баённомасида) олдиндан аниқланган *Content– Type* номли ўзгарувчи мавжуд. У ўзида Web саҳифа тилини ва кодлаш усулини сақлайди. Умумий кўриниш қуйидагича бўлади:

<META http–equiv = "Content–Type" content = "text/HTML; charset=ISO–8858–5">

Юқоридаги мисолда ўзгарувчининг қиймати ";" белги билан ажратилган икки қисмдан иборат. Биринчи қисм матннинг HTML тэглари ёрдамида яратилган оддий матн эканлигини билдирса иккинчи қисм фойдаланилган кодлаш усулини кўрсатиб туради. Юқоридаги мисолда халқаро стандартлаштириш ташкилоти (ISO) томонидан тасдиқланган стандарт кодлаш усули кўрсатилган. Афсуски браузерлар матнда учрайдиган баъзи бир белгиларни экранда акс эттира олмайди. Агар браузер матнда "кичик" тенгсизлик белгисини учратса уни тэг учун очилувчи қавс деб тушунади. Матнда бу белгидан кейин ҳеч қандай тэг учрамаса матннинг бирор бир қисми эътиборсиз қолдирилади ва экранда акс эттирилмайди. Бундай хатоликларнинг олдини олиш учун матнда бундай белгилар ўрнига қўштирноқ ичига олинган махсус белгилар кетма – кетлиги қўлланилади.

1.4. JavaScript тили асослари.

Юқорида қайд этиб ўтилганидек, JavaScript интерпретация қилинадиган тиллар тоифасига киради, яъни Java виртуал машинаси дастурнинг бошланғич кодини ўқийди ва шу заҳоти уни бажаради. Шу билан JavaScript дастурнинг бошланғич коди компилятор дастур ёрдамида процессорнинг командаларига айлантириладиган компиляция қилинувчи тиллардан фарқ қилади. Интерпретация қилинаётган тилдаги дастурларни бажарувчи виртуал машина *интерпретатор*, деб аталади.

JavaScript тилидаги дастур билан ишлашга оид кичик бир мисолни кўриб чиқамиз.

```
function somefunction (x, y, z, t)
{var a, b, c;
a = x * y;
b = z / t;
c = a–b;
return c;}
```

Бу ерда формулалар *return* параметрлари сифатида сонларни қабул қилувчи ва улар билан ҳисоблаш амалларини бажарувчи функцияни тавсифлайди.

Маълумки, исталган тилдаги ҳар қандай дастур муайян *ифодалардан* таркиб топади. Ифодалар бу дастур томонидан бажариладиган муайян амаллар тавсифидир. Ифодалар *операторлар* ва *операндлардан* ташкил топади; операторлар бажариш лозим бўлган амални, операндлар эса– тавсифланган амалларни бажариш талаб этилган маълумотларни юклайди. Ўзгарувчилар ва литераллар операндлар бўлиши мумкин.

Бу ерда биз, Web-скриптларни ёзишнинг бир нечта принципларини билиб оламиз. Энг аввало, <SCRIPT> тегига тўхталамиз. Бу тег скриптларни HTMLга киритишга хизмат қилади. У қуйидаги форматга эга:

```
<SCRIPT [LANGUAGE = “{Скрипт ёзилган дастурлаштириш тили}”]
[SRC = “{Скриптли файл адреси}”]>
... Скрипт матни
</SCRIPT>
```

Скрипт матни <SCRIPT> тегининг ичига жойлаштирилади. LANGUAGE атрибути скрипт қайси дастурлаштириш тилида ёзилганлигини кўрсатиш имконини беради. Ўз-ўзидан белгиланувчи ифода–“JavaScript”. Internet Explorer, бундан ташқари, VBScript тилида ёзилган скриптларни ҳам қўллаб-

қувватлайди. Бу тил атрибутининг ифодаси–“VBScript”. Navigator JavaScript интерпретатори версиясини яратиш имконини беради.

<SCRIPT LANGUAGE = “JavaScript 1.2”>

Бу JavaScript тилининг муайян версияга хос бўлган имкониятидан фойдаланганингизда қўл келиши мумкин.

1-жадвалда JavaScript интерпретаторининг версиялари ва Navigator иловасининг ҳар версияларига уларнинг мувофиқлиги келтирилган.

1-жадвал. **JavaScript интерпретаторининг версиялари.**

JavaScript	Navigator	JavaScript	Navigator
1.0	2.0	1.2	4.0
1.1	3.0	1.3	4.06

Иккинчи атрибут SRC скриптли файл адресини юклаш учун хизмат қилади. Бу ҳолда одатда <SCRIPT> жуфт теги якка тегга айланади.

<SCRIPT SRC = “printofdate.js”

js–JavaScript файллари учун стандарт кенгайиш.

Хўш, агар браузер скриптларни қўллаб-қувватламаса (масалан, бу дастурнинг ҳаддан ташқари эски версияси бўлса ёки фойдаланувчи хавфсизлик настройкаларидан скриптларни қўллаб-қувватлашни олиб ташлаган бўлса), нима бўлади? Браузер <SCRIPT> тегига эътибор бермайди ва экранга скрипт матнини чиқаради. Бунга йўл қўймаслик учун <SCRIPT> теги ичидаги скриптни шарҳ ичига олиш тавсия этилади.

```
<SCRIPT>
<!--
... Скрипт матни
-->
</SCRIPT>
```

Хўш, Web-скрипт нималарга эга бўлиши мумкин? JavaScript тилининг исталган кодига. Масалан, функция кодига.

Оддий Web-саҳифанинг бироз такомиллаштирилган кўринишига мисол келтирамиз.

```
<HTML>
<HEAD>
<TITLE> Бугун </TITLE>
<SCRIPT>
function writeDate ( ) {
var d;
d = new Date ( );
document.write (d.toString ( )); }
</SCRIPT>
</HEAD>
<BODY>
<P>
<SCRIPT LANGUAGE = “JavaScript”>
write Date ( );
</SCRIPT>
</P>
</BODY>
</HTML>
```

Бу саҳифа икки скриптга эга: бири–функция кодили скрипт, иккинчиси–ҳужжатга санани киритиш учун бу функцияни амалда чақирувчи скрипт. Биз, функция барча чақиришлардан олдин белгиланиши учун функция кодили скриптни HTML ҳужжатининг сарлавҳасига жойлаштирдик. Бу динамик Web-саҳифаларни ёзишда одатдаги амалдир: барча функцияларнинг белгилари саҳифа сарлавҳасига чиқарилади.

Энди, HTML ва JavaScript бир-бири билан қандай алоқа қилишини кўриб чиқамиз.

Ҳужжатнинг объект модели

Яна ўз скриптимизни кўриб чиқамиз.

```
var d;
d = new Date ( );
document.write (d.toString ( ));
```

document ҳужжати–умуман бизнинг ҳужжатимиз, write эса–унинг методи. Бу метод параметр сифатида берилган матнни HTML ҳужжатининг тегишли жойига киритади.

Web-саҳифани тавсифловчи объектлар, уларнинг хоссалари ва методлари мажмуи ҳужжатнинг объект модели (Document Object Model, DOM) деб аталади. Web-саҳифани яратиш, яъни HTML кодини JavaScript коди билан бирлаштириш технологияси (бунда JavaScript коди объект модели ёрдамида саҳифани бошқаради) динамик HTML (Dynamic HTML) деб аталади. Илгари Microsoft ва Netscape объект моделини амалга ошириш учун таклиф қилган одатдаги HTML кенгайишлари мажмуи ҳам динамик HTML деб аталар эди. W3C 1 даражали DOM спецификацияси (DOM level 1)ни қабул қилганидан кейин бу қўшимчаларнинг барчаси HTML 4.0 нинг охирги таҳририга киритилди.

HTML ва JavaScript. Ҳодисалар

Маълумки, HTML маълумотларни ифодалаш учун жавоб беради, Java Script эса, уларнинг хулқ-атворини тавсифлайди. Жорий санани кўрсатувчи динамик Web-саҳифага мисол келтирамиз. Бу саҳифа скриптга эга. Келинг, унинг кодига назар ташлаймиз.

```
<HTML>
<HEAD>
<TITLE> Бугун </TITLE>
</HEAD>
```

```
<BODY>
<P>
<SCRIPT LANGUAGE = “JavaScript”>
var d;
d = new Date ( );
document.write (d.toString ( ));
</SCRIPT>
</P>
</BODY>
</HTML>
```

Браузер бу саҳифани юклагач, у шу заҳоти скриптни бажаради. Натижада, <P> тегига минтақавий хусусиятдан келиб чиқиб тузилган бугунги сананинг матнли ифодаси киритилади. Бунга қуйидаги код ёрдамида эришилади:

```
var d;
d = new Date ( );
document.write (d.toString ( ));
```

Бу ерда биз, жорий сана ифодаси билан инициализация қилинган Date туркуми объектини яратамиз, toString методини чақирамиз ва у қайтарган натижани document объектининг write методига берамиз. Охирги метод аргумент сифатида берилган матнни Web-саҳифанинг тегишли жойига қўяди.

Бундай скриптлар энг содда скриптлардир. Амалда улар нодинамик саҳифалар яратади, чунки улар бир марта бажарилганидан кейин саҳифа фойдаланувчининг ҳаракатларига жавоб бермайди ва ўз мазмунини ўзгартирмайди.

Ҳодисаларга ишлов берувчилар

Сценарийларнинг тилларига мувофиқлаштириш учун айрим HTML тегларига юзага келувчи ҳодисаларга ишлов бериш махсус параметрлари

киритилган. JavaScript тилининг операторлари мазкур параметрларнинг ифодалари бўлиши мумкин. Одатда ифода тариқасида функция номи юкланади. Ҳодисага ишлов бериш параметри билан белгиланган тегишли ҳодиса рўй берганида функция номи чақирилади. Параметрнинг номи *on* олд қўшимчаси билан бошланади, унинг кетидан ҳодиса номи келади. Масалан, Click ҳодисасига ишлов бериш параметри («сичқон» тугмаси босилади) onClick деган номга эга бўлади.

Ҳодисалар асосан фойдаланувчи HTML шакли элементлари ёрдамида амалга оширувчи ҳаракатлар билан боғлиқ. Шу боис, ҳодисаларни ушлаш ва уларга ишлов бериш кўпинча шакллар элементларининг параметрларида юкланади. Бу киритилган ахборотни CGI-сценарий билан ишлов беришга юборишдан олдин текшириш имконини беради.

Функция ёки *процедура*–операторларнинг номланган кетма-кетлиги бўлиб, у муайян вазифани бажаради ва муайян ифодани қайтариши мумкин. Функция қуйидаги синтаксисга эга бўлган *function* оператори билан белгиланади:

function функция_номи ([параметрлар]) {

[JavaScript операторлари]

[return ифода]}

Функция ифодаловчи параметрлар вергуллар (,) билан ажратилади. Функция танаси (фигурали қавс ичига олинган операторлар блоки)даги *return* номажбурий оператори функция қайтарувчи ифодани белгилайди.

Функцияни эълон қилиш ва уни чақириш ўртасидаги фарқни аниқ тушуниш лозим. *Функцияни эълон қилиш* фақат унинг номини юклайди ва функция чақирилганида у нима қилишини белгилайди. Сценарийда функция *чақирилганида* ва унинг ҳақиқий параметрлари узатилганида функциянинг бевосита бажарилиши амалга ошади.

Зарур функцияларни белгилаш <HEAD> тегида амалга оширилиши лозим, чунки унда белгиланган сценарийнинг барча операторлари саҳифа акс

этгунга қадар интерпретация қилинади ва бутун саҳифа акс этиши жараёнида маълум бўлади.

Қуйидаги мисол ҳужжат саҳифасини шакллантириш жараёнида функциянинг юкланиши ва унинг чақирилишини намойиш этади.

```
<HTML>
<HEAD>
<TITLE> функцияни юклаш </TITLE>
<SCRIPT LANGUAGE = “JavaScript”>
<!--// JavaScriptни қўллаб-қувватламаётган браузерлардан сценарийни яшириш
function square (number) {
alert (“Функцияларни ҳисоблашим ва кейин ҳужжат тузишим керак!”);
return number * number; }
//-->
</SCRIPT>
</HEAD>
<BODY>
<P> функцияни ҳисоблаш сценарийси киритилган саҳифа акс эта бошлайди </P>
<SCRIPT>
<!--
document.write («Ҳисобланган ифода тенг», square (s), “.”);
//-->
</SCRIPT>
<P> саҳифанинг шаклланиши шу билан тугади.
</BODY>
</HTML>
```

Тег <HEAD>да ўз параметри ифодасининг квадратини қайтарувчи, шунингдек маълумот дарчасини акс эттирувчи square ( ) функциясининг тавсифи берилган. Функцияни чақириш HTML ҳужжати танасида жойлашган сценарийда амалга ошади. Бу сценарийда HTML саҳифасини шакллантиришга мўлжалланган *document* объектининг *write* методи қўлланилади.

Ушбу ҳужжат Internet Explorer браузерига юкланганида янги абзац акс этади, сўнгра маълумот дарчаси чиқади.

OK тугмасининг босилиши маълумот дарчасини беркитади ва саҳифани акс эттиришни давом эттиради.

Ўзгарувчилар

JavaScript тили маълумотларга муваффақиятли ишлов бериши учун у мазкур маълумотларни бирор ерда сақлаши керак. Бунинг учун хотирада *ўзгарувчилар*, деб аталадиган махсус жойлар ажратилади.

Ҳар бир ўзгарувчи уни аниқ идентификация қилувчи муайян номга эга бўлиши керак. Ном лотин ҳарфлари, рақамлар ва чизиш белгилари “_” дан ташкил топиши лозим. Бунда биринчи символ ё лотинча ҳарф, ё чизиш белгиси бўлиши керак. Мисол учун, FrameName, _link–ўзгарувчиларнинг тўғри номлари, 678vasya, Имя Фрейма–нотўғри номлар. JavaScript ҳарфлар регистрига таъсирчан, шу боис framename ва FrameName–ҳар хил ўзгарувчилардир.

Ўзгарувчиларнинг номлари *таянч сўзлар*, яъни тил операторларини ёзиш учун қўлланиладиган тиллар билан мос келмаслиги керак. Бундай таянч сўзлар унча кўп эмас, шу боис уларни эслаб қолиш осон.

Энг аввало, ўзгарувчини эълон қилиш керак. Бу JavaScript интерпретатори унинг ўзгарувчи эканлигини билиши ва уни нотўғри ёзилган оператор деб қабул қилмаслиги учун зарурдир. Ўзгарувчилар икки усулда:

– *var* оператори ёрдамида;

– *қайд этиш* оператори ёрдамида эълон қилинади.

var оператори ўзгарувчини юклаш ёки уни инициализация қилиш учун қўлланилади. У қуйидаги синтакисга эга:

var ўзгарувчи_номи [= бошланғич_ифода];

Мисол учун: **var** FrameName, _link;

Бу ўзгарувчиларни тўғри эълон қилишга мисолдир (var таянч сўзи қора шрифт билан ажратилган). Бу ўринда шуни қайд этиб ўтиш керакки, бирваракай бир нечта ўзгарувчиларни эълон қилиш мумкин, бунинг учун уларнинг номларини вергул билан ажратиш лозим.

Қайд этиш номажбурий оператори ўзгарувчининг маълумотларини юклашга хизмат қилади. Ўзгарувчининг типи маълумотлар типи билан белгиланади. Масалан, қуйидаги оператор

var weekDay = “Жума”;

weekDay ўзгарувчисини юклайди, уни «Жума» сатр ифодаси билан қайд этади ва шу тариқа унинг сатр типини белгилайди.

Агар ўзгарувчини белгилашда унинг бирон-бир ифодаси қайд этилган бўлмаса, унинг типи белгиланмаган бўлади. Ўзгарувчининг типи фақат қайд этиш оператори = унинг муайян ифодасини қайд этганидан кейин белгиланади.

Бу операторни дастурнинг исталган жойида қўллаш ва унинг ёрдамида ўзгарувчининг типини ўзгартириш мумкин. Шу билан JavaScript тили дастурлаштиришнинг ўзгарувчилар типи тилдан фойдаланишдан олдин белгиланиши лозим бўлган қатъий типлаштирилган тилларидан (масалан, Java ёки C^{++} дан) фарқ қилади. Коднинг қуйидаги фрагментида weekDay ўзгарувчиси ўз типини ўзгартиради:

```
var weekDay // Ўзгарувчи белгиланган, лекин унинг типи маълум эмас
. . . . . .
weekDay = “Жума”; // Сатрли ўзгарувчи
. . . . . .
```

weekDay = 5; // Бутун ўзгарувчи

Литераллар

Дастурлаштириш тили, айни ҳолда JavaScript тилининг қоидаларига мувофиқ ёзилган муайян типга мансуб маълумотлар литераллар, деб аталади. Шундай қилиб, литераллар тегишли типдаги маълумотларнинг айнан ифодаларидир.

Литералларни ёзиш қоидалари 1-жадвалда кўриб чиқилган. Бу ерда биз, фақат бир нечта мисоллар келтириб ўтамиз:

“Сатр”;

“Апостроф” ичидаги сатр;

0123456789–ўнли бутун сон;

01234567–саккизли бутун сон;

0X17AF–ўн олтили бутун сон;

7.8–сузувчи нуқтали сон;

4.5–сузувчи нуқтали, лекин каср қисмига эга бўлмаган сон;

1.2E-10–JavaScript қоидаларига кўра ёзилган $1{,}2 \cdot 10^{-10}$ сони.

Ифодалар

Ифодалар–ўзгарувчилар, литераллар ва операторлар комбинацияси бўлиб, уларни ҳисоблаш натижасида ягона бир ифода ҳосил бўлади. Бу ифода сонли (бутун ёки моддий), сатрли ёки булли бўлиши мумкин.

Ифодаларда ўзгарувчилар ё var оператори билан, ё қайд этиш оператори (=) билан инициализация қилиниши лозим. Агар ифодани ҳисоблашда инициализация қилинмаган ёки мутлақо белгиланмаган ўзгарувчи учраса, интерпретатор “undefined variable” («ўзгарувчи белгиланмаган») хатосини қайд этади ва унинг HTML саҳифасидаги ўрнини кўрсатади.

Энди JavaScript тилида ифодалар тузишнинг бир неча қоидаларини келтириб ўтамиз:

–ҳар бир JavaScript ифодаси нуқта ва вергул “;” билан тамомланиши керак;

–ифода бир сатрни ёки бир неча сатрни эгаллаши мумкин бўлиб, бу ҳолда ҳам нуқта ва вергул ифода тугаганлигининг белгиси ҳисобланади;

–ифода эга бўлиши мумкин бўлган операторлар ва операндлар сони чекланмаган.

Бироқ, шуни қайд этиб ўтиш керакки, ҳаддан ташқари мураккаб ифодаларни нафақат бошқа дастурчи, балки интерпретаторнинг ўзи ҳам тушунишга кўпинча қийналади. Шу боис, мураккаб ифодаларни нисбатан содда ифодаларга ажратган маъқул.

JavaScript тилидаги ҳар хил ифодаларни ёзишнинг *асосий принципларини* ва бунда қўлланиладиган операторларни кўриб чиқамиз.

Маълумотларга ишлов бериш ифодалари

JavaScript арифметик ифодалари “=” қайд этиш оператори билан ажратилган икки қисмдан ташкил топади. Унинг чап томонида янги ифода берилаётган ўзгарувчи, ўнг томонида эса–мана шу янги ифода кўрсатилади. У литерал, бошқа ўзгарувчи ёки муайян ифодани ҳисоблаш натижаси бўлиши мумкин.

Бунга бир неча мисолларни кўриб чиқамиз:

```
a = 127;
b = a;
c = a + b;
```

Бу уч ифода мутлақо тўғри ифодалардир. Улар математик формулаларга ўхшаб кетади, бироқ нисбатан қатъий нотацияда ёзилган.

Арифметик операторлар

Булар, аввалambор, ҳаммамиз яхши биладиган қўшиш (+), айириш (-), кўпайтириш (*) ва бўлиш (/) операторларидир. Улар иккита операндни қабул қилади ва битта натижа беради. Уларга қўшимча тарзда қолдиқни олиш (%)

оператори ҳам мавжуд бўлиб, у бир сонни бошқа сонга нобутун бўлади ва бўлишдан қолган қолдиқни қайтаради.

Математик инверсия оператори операнд белгисини ўзгартиради. Масалан, мусбат сонни манфийга, манфий сонни эса–мусбатга айлантиради.

Инкремент (++) ва декремент (--) операторлари операнднинг ифодасини биттага кўпайтиради ёки камайтиради. Бунда, агар оператор белгиси операнд олдида турган бўлса, аввал операция–инкремент ёки декремент амалга оширилади, сўнгра операнднинг ўзгартирилган ифодаси қайтарилади. Бордию, оператор белгиси операнддан кейин турган бўлса, аввал операнднинг ўзгартирилмаган ифодаси қайтарилади, сўнгра операция амалга оширилади.

$a = b^{++}$;

$c =_{-}^{-} c$;

Охирги оператор мутлақо тўғридир. Бу ҳолда с ўзгарувчисидан аввал эски ифода олинади, сўнгра декрементация қилинади ва яна с ўзгарувчиси берилади. JavaScript ва бошқа дастурлаштириш тиллари оддий арифметикада қўлланмайдиган бундай ёзувнинг қайд этилишига йўл қўяди.

Сатрларга ишлов бериш оператори

Сатрларга ишлов бериш оператори атиги битта. Бу сатрларни қўшиш оператори (+)дир. У ўзининг арифметик цехдаги «ҳамкасби»га ўхшайди, лекин у икки сатрни бирлаштиради, яъни иккинчи сатрни биринчи сатр охирига жойлаштиради.

title = “Java” + “Script”;

Иккиталик операторлар

Иккиталик операторлар сонларнинг иккиталик ифодаси билан боғлиқ бўлиб, асосан JavaScript тилида қуйи даражали дастурлаштириш учун қўлланилади. Бундай операторлар еттита бўлиб, уларнинг ҳаммаси 2-жадвалда келтирилган.

2 - жадвал. Иккиталик операторлар

Оператор	Ифодаси	Тавсифи
Иккиталик инверсия	a = b	
Иккиталик ВА	a = b & c	
Иккиталик ЁКИ	a = b / c	
Соқит этувчи иккиталик ЁКИ	a = b ^ c	
Чапга силжиш	a = b << c	Кичик даражаларни нол билан тўлдириш орқали чапга c белгига силжиш
Ўнгга силжиш	a = b >> c	Катта даражаларни белгилар разряди (катта даража) ифодалари билан тўлдириш орқали ўнгга c белгига силжиш
Белгини ҳисобга олмасдан ўнгга силжиш	a = b >>> c	Катта даражаларни нол билан тўлдириш орқали ўнгга c белгига силжиш

Қайд этиш операторлари

Қайд этиш операторларидан бири– = бизга маълум. Бу ҳар қандай ҳолатда иш бериши мумкин бўлган универсал оператордир. Бироқ, JavaScript тили яна бир нечта қайд этиш операторларига эга бўлиб, улар асосан маълумотларга ишлов беришнинг мураккаб ифодаларини тузишга мўлжалланган. Улар 3 - жадвалда келтирилган.

3 - жадвал. Қайд этиш операторлари

Оператор	Эквивалент ифода
a + = b;	a = a + b;
a-= b;	a = a-b;
a * = b;	a = a * b;
a / = b;	a = a / b;
a % = b;	a = a % b;
a << = b;	a = a << b;
a >> = b;	a = a >> b;
a >>> = b;	a = a >>> b;
a & = b;	a = a & b;
a ^ = b;	a = a ^ b;
a / = b;	a = a /b;

Маълумотлар типларининг ўзаро мувофиқлиги

Ниҳоят, маъруза охирида яна бир муҳим масала–маълумотлар типларининг ўзаро мувофиқлиги ва бир типни бошқа типга автоматик тарзда айлантириш масаласини кўриб чиқамиз.

Агар иккита сонли ифода қўшилса, нима бўлади? Яна битта сонли ифода ҳосил бўлади. Агар сон билан сатр қўшилса-чи? Бу ерда JavaScript интерпретатори маълумотлар типларининг номувофиқлиги муаммоси билан тўқнаш келади ва улардан бирини иккинчисига айлантириб, бу типларни ўзаро мувофиқлаштиришга ҳаракат қилади. Бизнинг ҳолатда сон сатрга айлантирилади ва кейин олинган икки сатр биттага бирлаштирилади.

Type of оператори

Type of оператори ўзгарувчи ёки ифодани операнд сифатида қабул қилади ва операнд маълумотлари типини тавсифловчи сатрни қайтаради.

Агар операнд сонли (бутун ёки сузувчи нуқтали) маълумотга эга бўлса, “number” сатри қайтади, бордию операнд сатрли маълумотга эга бўлса, “string” сатри, мантиқий маълумот учун эса–“bolean” сатри қайтади. Агар ўзгарувчи умуман маълумотга эга бўлмаса, (var оператори томонидан эълон қилинган, лекин ифодаси берилмаган бўлса) ёки ифода (типлар муваффақиятсиз ўзгартирилиши натижасида) ноаниқ маълумот қайтарса, “indefined” сатри қайтади.

typeof оператори қуйидаги форматга эга:

typeof ({Ифода})

Ушбу операторнинг қўлланилиши билан боғлиқ мисолларни кўриб чиқамиз:

if typeof (a + 1) == “number”

f = typeof (somevar);

Операторлар устуворлиги

Мураккаб ифодалар тузишда дастурчи назарда тутиши лозим бўлган яна бир тушунча бу операторлар устуворлиги, яъни уларни бажариш кетма-кетлиги тушунчасидир. Биз, ўрганган барча операторлар 4 - жадвалда устуворлик даражасига қараб келтирилган.

4 – жадвал. Операторлар устуворлиги

Операторлар	Тавсифи
++ −−−typeof	Инкремент, декремент, арифметик ва мантиқий инверсия, тип белгиси
* / %	Кўпайтириш, бўлиш, бўлишдан қолган қолдиқ
+ −	Сатрларни қўшиш ёки бирлаштириш, айириш
<< >> >>>	Иккиталик силжишлар

&	Иккиталик ВА
^	Соқит этувчи иккиталик ЁКИ
/	Иккиталик ЁКИ
={оператор}=	Оддий ёки мураккаб қайд этишлар

Бир хил устуворликка эга бўлган операторлар чапдан ўнгга қараб бажарилади.

Махсус символлар

JavaScript сатрли ифодалари махсус символларни–ифодага бошқача тарзда киритиб бўлмайдиган муайян хизматга доир ёки чоп этилмайдиган символларни ифодаловчи белгилар комбинациясини ўз ичига олади. JavaScript тилида ўқиладиган барча махсус символлар 5 - жадвалда келтирилган.

5 – жадвал. Махсус символлар.

Символ	Тавсифи
/b	Ажратма (backspace)
/ f	Янги саҳифа (Саҳифани ўтказиш)
/ n	Янги сатр (сатрни ўтказиш)
/ r	Кареткани қайтариш
/ t	Горизонтал табуляция
/ ′	Апостроф
/ ″	Қавс
/ /	Тескари слэш

Энди, айтайлик, сатрга кареткани қайтариш символини киритмоқчи бўлсангиз, қуйидаги ифодани ёзинг:

“Some string with carriage return/n”

Шартли операторлар, цикллар ва шарҳлар

Шартли операторлар маълумотларга ишлов бериш билан эмас, балки дастур ишини бошқариш билан шуғулланганликлари учун улар алоҳида гуруҳга ажратилган. Шартли оператор муайян шарт берилган (ёки берилмаган) ҳолда коднинг муайян қисмини бажариши (ёки, аксинча, бажармаслиги) мумкин. Мантиқий ўзгарувчи ифодаси ёки мантиқий ифодани ҳисоблаш натижаси мана шундай шарт бўлиб хизмат қилади.

Бу ерда мантиқий ифода тушунчасига таъриф бериб ўтиш жоиз. Мантиқий ифода JavaScript тилининг одатдаги ифодасидир, бироқ уни ҳисоблаш натижаси *true* ёки *false* мантиқий катталиклари ҳисобланади. Мантиқий ифода қуйида кўриб чиқиладиган мантиқий таққослаш операторларига эга.

Блокли ифодалар

JavaScript бир нечта ифодани битта ифодага бирлаштириш имконини беради. Бундай ифода *блокли ифода* ёки *блок,* деб аталади. Бунинг учун керакли код фигурали қавс {} ичига олинади.

```
a = 11;
{
b = “12”;
c = a-b;
}
```

Блоклардан мураккаб ифодалар, хусусан, шартли операторларга эга ифодалар тузиш учун фойдаланилади.

if-else тармоқлаш оператори

Тармоқлаш оператори муайян ҳодиса содир бўлганида коднинг муайян қисмини бажариш имконини беради. Ўзгарувчига муайян ифоданинг берилиши ёки муайян ифодани ҳисоблашнинг у ёки бу тарзда алоҳида муносабат билдирилиши лозим бўлган натижаси мана шундай ҳодиса бўлиб хизмат қилиши мумкин. Тармоқлаш оператори қуйидаги форматга эга:

if ({Шарт}) ... “у ҳолда” блоки

[else

... “акс ҳолда” блоки]

Шарт–мантиқий ифода бўлиб, унга мувофиқ интерпретатор қайси блокни бажариш тўғрисида қарор қабул қилади. Агар шарт true («ҳақиқат») ифодасига эга бўлса, “у ҳолда” блоки бажарилади. Агар шарт false («ёлғон») ифодасига эга бўлса, “акс ҳолда” блоки бажарилади, (агар у мавжуд бўлса). Бордию, «акс ҳолда» блоки мавжуд бўлмаса, дастурнинг навбатдаги ифодаси бажарилади. Агар шартнинг натижаси null ёки undefined бўлса, тармоқлаш оператори falsega қандай муносабат билдирган бўлса, унга ҳам шундай муносабат билдиради.

Мана, бунга бир мисол:

```
if (x == 1){
f = 3;
h = 4;
}
else
{
f = 33;
h = 44;
}
```

Бу ерда биз, x ўзгарувчининг ифодасини бир (1)га таққослаймиз ва таққослаш натижаларига қараб f ва h ўзгарувчиларга ҳар хил ифодалар берамиз. Шартга эътибор беринг–*мантиқий таққослаш оператори* шундай топилади.

Таққослаш операторлари

Таққослаш операторлари 6 - жадвалда келтирилган.

6 – жадвал. Таққослаш операторлари.

Оператор	Тавсифи	Оператор	Тавсифи
<	Кичик	&&	Мантиқий ВА
>	Катта	//	Мантиқий ЁКИ
==	Тенг	!	Мантиқий ЭМАС
<=	Кичик ёки тенг	===	Қатъий тенг
>=	Катта ёки тенг		
! =	Тенг эмас	! ==	Қатъий тенг эмас

Дастлабки олти оператор билан ҳеч қандай муаммо туғилмаслиги керак–улар икки ифодани бир-бирига таққослайди ва шарт бажарилган бўлса, true ни, бажарилган бўлмаса–false ни қайтаради.

Кейинги уч оператор мураккаброқ мантиқий ифодалар тузиш имконини беради. Улар икки мантиқий ифода устида ВА, ЁКИ ва ЭМАС мантиқий операцияларини бажаради ва натижани қайтаради.

Охирги икки оператор–«қатъий тенг» ва «қатъий тенг эмас» операторларига келсак, улар қатъий тенглик ва қатъий тенгсизлик операторлари деб аталади. Гап шундаки, одатдаги «тенг» ва «тенг эмас» операторлари ҳар хил типдаги операндларга дуч келса, уларни муайян бир типга айлантиришга ҳаракат қилади. Қатъий тенглик ва қатъий тенгсизлик операторлари бундай қилмайди, операндлар ўзаро мос келмаган тақдирда falseни қайтаради. Бу баъзан, масалан, фойдаланувчининг киритмасини текширишда фойдалидир.

Биз, юқорида кўриб чиққан таққослаш операторлари ҳар хил устуворликка эга. Энг устувор оператор–! (мантиқий ЭМАС) оператори. У устуворлик жиҳатидан инкремент, декремент ва инверсия операторларига

тенг. <, >, <= ва >= операторлари битли силжиш операторлари орқасидан келади, уларнинг орқасидан эса–==, !=, === ва !== операторлари келади. Иккиталик ЁКИ кетидан мантиқий ВА, унинг кетидан эса–мантиқий ЁКИ келади. Улар энг паст даражали устуворликка эга.

Қатъий таққослаш операторлари Navigator да 4.06 версиясидан бошлаб қўлланилади.

Оператор ?

Бу яна бир шартли оператор.

{Шарт} ? {“у ҳолда» ифодаси} : {“акс ҳолда” ифодаси}

Бу оператор шарт ҳақиқий бўлган тақдирда “у ҳолда” ифодасини, шарт ҳақиқий бўлмаган тақдирда эса–“акс ҳолда” ифодасини қайтаради.

a = (f == 2) ? b : c + 2;

Агар f 2 га тенг бўлса, ифода а ўзгарувчига b ўзгарувчининг ифодасини, акс ҳолда эса–c + 2 ифодасининг натижасини жойлаштиради.

Бу устуворлик даражаси энг паст бўлган операторлардан биридир. Ундан қуйида фақат қайд этиш оператори туради.

Цикллар

Цикллар–бир неча марта бажарилувчи блоклар. Циклнинг такрорланиши муайян шарт берилишига қараб тўхтатилади. Бу муайян шарт берилганида коднинг муайян блоки бажариладиган ёки бажарилмайдиган шартли операторлар билан боғлиқ ҳолатга ўхшайди.

JavaScript дастурчиларга цикларнинг бир неча турларини таклиф қилади.

for цикли

Токи муайян шарт ҳақиқий бўлиб қолар экан, for цикли бажарилади, (true қайтарилади). У коднинг муайян фрагментини маълум марта бажариш учун қўлланилади. Бу ҳолда коднинг мазкур фрагменти неча марта

бажарилганлигини ҳисобловчи бутун сон *цикл ҳисоблагичи*, деб аталадиган ўзгарувчига жойлаштирилади. Коднинг циклда бажарилувчи блоки эса *цикл танаси* ҳисобланади.

for ({Инициализация ифодаси}; {Шарт};

{Инкремент ифодаси}) ... Цикл танаси

Инициализация ифодаси ўзгарувчи-ҳисоблагичнинг бошланғич ифодасини беради. Шундан кейин шарт текшириб кўрилади ва агар унинг ифодаси ҳақиқий бўлса, цикл танаси бажарилади. Шундан сўнг, инкремент ифодаси ҳисоблагичнинг ифодасини ўзгартиради ва яна шарт текшириб кўрилади. Токи, шарт ёлғон (false) тус олмагунича бу амал қайта-қайта бажарилаверади. Ҳисоблагич билан боғлиқ бундай цикллар дастурларда жуда кўп учрайди.

```
for (i = 1; i < 11; i++) {
a += 3;
b = i * 2 + 1;
}
```

Бу цикл 10 марта бажарилади. Инициализация ифодаси ва шартга эътибор беринг. Биз, i ҳисоблагичига 1 бошланғич ифодасини берамиз ва ҳар цикл танаси бажарилганидан кейин уни биттага кўпайтирамиз. Ҳисоблагич ифодаси 11га етганида цикл бажарилишдан тўхтайди ва шарт ёлғонга айланади.

Биз, цикл ҳисоблагичини цикл танасининг ифодаларидан бирида қўлладик–бундай қилиш мумкин. i ҳисоблагичи 1 дан 10 гача бўлган ифодаларга изчил ўсиб борадики, улардан ҳисоблашда фойдаланиш мумкин.

Оператор, (вергул)

JavaScript асосан, for цикллари инкрементининг ифодаларида қўлланувчи операторни таклиф қилади. Бу , (вергул) операторидир. У

инкрементнинг бир у, бир бу ифодасини кетма-кет бажариш имконини беради. У қуйидаги форматга эга:

{“Тоқ ифода”}, {“Жуфт ифода”}

Циклнинг биринчи ўтишида «тоқ» ифода, иккинчи ўтишида–«жуфт» ифода, учинчи ўтишида эса–яна «тоқ» ифода бажарилади ва ҳ.к.

for (i = 0; j < 11; i^{++}, j^{++}) {

a = i ∗ 2 + 1;

b = j / 2-3;

, (вергул) оператори устуворлик жиҳатидан барча операторлар орасида энг охирги ўринда туради.

do-while цикли

do-while цикли for циклига кўп жиҳатдан ўхшаш, бироқ унда ҳисоблагич қўлланилмайди. Токи, шарт ҳақиқий бўлиб қолар экан, do-while цикли бажарилаверади. Бунда шарт цикл танаси бажарилишидан олдин эмас, балки у бажарилганидан кейин текширилади. Шу боис, ҳатто шарт аввалбошдан сохта бўлган тақдирда ҳам, do-while цикли ҳеч бўлмаса бир марта бажарилади.

do

... Цикл танаси

while ({Шарт});

Сиз do-while циклини ҳар хил усулда қўллашингиз мумкин:

do {

a = a ∗ i + 2;

i = ++ i;

} while (a < 100);

Юқорида кўриб чиқилган мисолда ажратилган шартнинг берилиши текширилади.

while цикли

while цикли do-while циклини эслатади, аммо шарт цикл танаси бажарилишидан олдин текширилади. Шу боис, агар у аввалбошдан сохта бўлса, цикл бирор марта ҳам бажарилмайди.

```
while ({Шарт})
... Цикл танаси
```

Мисол учун:

```
while (a < 100) {
a = a * i + 2;
i = ++ i;
}
```

break ва continue операторлари

Баъзан циклнинг бажарилишини тўхтатиш талаб этилади. Бунинг учун JavaScript дастурчиларга break ва continue операторларини таклиф қилади.

break оператори циклнинг бажарилишини тўхтатиш ва ва ундан кейинги ифодага ўтиш имконини беради.

```
while (a < 100) {
a = a * i + 2;
if (a > 50) break;
i = ++ i;
}
```

Бу мисолда биз, агар а ўзгарувчининг ифодаси 50 дан ошмаса, циклнинг бажарилишини (аммо дастурнинг бажарилишини эмас) тўхтатамиз.

continue оператори циклни қайта юклаш, яъни цикл танасига кирувчи барча кейинги командаларни бажармасдан қолдириш ва унинг бошига қайтиш имконини беради. Шарт текшириб кўрилади (for цикли, бундан ташқари, инкремент ифодасини бажаради) ва, агар шарт ҳақиқий бўлса, цикл танаси бошидан бажарилади.

```
while (a < 100) {
```

```
i = ++ i;
if (i > a && i < 11) continue;
a = a * i + 2;
}
```

Бу ерда биз i нинг 10 дан 20 гача бўлган барча ифодалари учун a нинг ҳисобланган барча ифодаларини ўтказамиз.

break ва continue операторларининг тўлиқ формати қуйидаги кўринишга эга:

```
break / continue [{Белги}]
```

Бу ерда *белги* айни break ёки continue оператори қўлланилаётган циклни белгилайди. Белги қуйидаги форматга эга:

```
{Белги номи} : ... Ифода
```

Белги номи дастур доирасида бетактрор бўлиши лозим. Ўзгарувчиларнинг номларига нисбатан амал қилувчи қоидалар белги номига нисбатан ҳам амал қилади.

Белгилар Internet Explorer ва Navigator иловаларида 4.0 версияларидан бошлаб қўлланилмоқда. Бу дастурларнинг олдинги версияларида break ва continue операторлари белгилар кўрсатилмасдан қўлланилган.

Шарҳлар

JavaScript ихтиёримизга шарҳлар киритишнинг икки операторини беради.

```
// ... Шарҳ сатри
```

Бу оператор ифода охирига бир сатрли шарҳни киритиш имконини беради.

```
a = b + c; // Бу–бир сатрли шарҳ
```

Эътибор берган бўлсангиз, шарҳ ифода тугаганини кўрсатувчи нуқта ва вергул (;)дан кейин жойлашган.

```
/*
```

... шарҳ

/*

Бу оператор эса, дастур кодига ҳар қандай катталикдаги шарҳни киритиш имконини беради.

/*

Бу ифодада биз, икки ўзгарувчининг ифодасини қўшамиз ва натижани учинчи ўзгарувчига жойлаштирамиз.

/*

a = b + c;

Х У Л О С А

I боб Интернет технологиялари оламига саёҳат деб номланиб қуйидаги жами бўлиб 4 та бўлимлардан иборат. Улар 1 - бўлим Интернет – ҳалқаро глобал тармоғи, 2 – бўлим Web браузерлар, 3 – бўлим HTML – Web ҳужжатларни яратиш тили, 4 – бўлим JavaScript тили асослари деб номланган бўлиб берилган мавзунинг назарий қисмини ўзида жамлаган. Бундаги келтирилган барча маълумотлар Интернет технологиялари оламига саёҳат, Интернет – ҳалқаро глобал тармоғи, Web браузерлар, HTML – Web ҳужжатларни яратиш тили, JavaScript тили асослари ҳақидаги маълумотлар асосида берилган.

Ҳар бир бўлим номидан келиб чиқадиган бўлсак, қўйиган мавзусига мос материаллар билан берилган, бундан ташқари кўргазмали асосида ифодаланган бўлиб, жами олтита жадвал ва тўртта расм орқали берилган. Бу эса бўлимларни янада тушинарли ифодаланишига асос бўлади.

II БОБ. PHP WEB ДАСТУРЛАШ ТИЛИ ВА MYSQL МАЪЛУМОТЛАР ОМБОРИНИ БОШҚАРИШ ТИЗИМИ. WEB ДАСТУРНИ СОЗЛАШ ВА УНДАН ФОЙДАЛАНИШ.

2.1. PHP Web дастурлаш тили.

PHP – ўз номини етарлича танитиб улгурган, дастурлаш тили ҳисобланади. Гап шундаки, бошланишда бу унча қийин бўлмаган шахсий WEB – саҳифаларини яратиш учун мўлжалланган оддий макрослар тўпламидан иборат бўлган бўлиб, PHP-personal home page (шахсий уй саҳифаси) сўзларининг қисқартмасидан иборат.

Тез орада у фойдаланувчилар назарига туша бошлади, ҳамда тезлик билан такомиллашиб, оммалашиб борди. 1997 йилдан бу тил устида програмистлар гурухи иш олиб боради.

Меҳнатларнинг самарасида эса PHP3 кейинги версия яратилди. Бу PHP нинг такомиллашган ва замонавий версия бўлиб, унда матнларининг қайта ишлашнинг янги усуллари яратилди ва бу усуллар Зиб Зураски ва энди Гутианс (Zeev, Surasky, Andi Ceutmans) исмли прогромистлар томонидан яратилди. Шунингдек тилнинг синтаксисида бироз ўзгаришлар киритилиб, янги функциялар қўшилди. Янги верция шу даврда сарвер учун дастурлаш тилларининг энг зўри ҳисобланиб, жуда ҳам тез оммалашиб кетди.

MySQL маълумотлар базаси ва Apache сервери билан ишлаш учун PHP нинг имкониятлари янада кенгайиб борди. Apache сервери ҳозирги кунда дунёдаги энг кенг тарқалган Web -сервер ҳисобланади ва PHP тили Apache сервери учун модул кўринишида қўлланилиши мумкин. MySQL- бу замонавий маълумотлар базаси бўлиб пулсиз (текин) тарқатилади, шунинг учун ҳам PHP нинг барча функциялари шу базага боғланган. Тан олиш лозимки Apache, MySQL ва PHP ларнинг ўзаро бир-бири билан боғлиқ равишда ишлаши ўртадаги рақобатга барҳам беради.

Бу эса PHP бошқа МБСИ билан ишламайди дегани эмас. Бу технология жуда МБСИ ва Web серверлар билан ишлаш имкониятига эга.

WEB саҳифаларни ва тармоқни яратиш йўллари ўзгариши билан PHP ҳам такомиллаша борди. 1990 - йил ўрталарига келиб катта тармоқларда ҳам HTML да ёзилган юзлаб статик саҳифалар ишлатилар эди. ҳозир эса жараён ўзгариб бормоқда. WEB саҳифаларини яратувчилар маълумотлар базаси билан ишловчи WEB саҳифаларни яратиш имконига эга бўлиб, фойдаланувчиларни қайта ишлаш имконига эга бўлган WEB саҳифаларни яратмоқдалар.

Маълумотларни сақлаш ва маълумотларга мурожат қилиш учун маълумотлар базасидан фойдаланиш янада актуаллашиб, мобил телефонлар, рақамли телевидения ва ҳоказолар. Турли хил қурилмаларда маълумотларни узатишда сифатни ўсишига эришишмоқда.

Бу фикрлар асосида айтиш мумкинки, келгусида PHP тили янада такомиллашиб ўзининг ўта юқори даражадаги дастурлаш тили эканлигини намоён қилади.

PHP турли хил системаларда ишлай олади. У система Windows, Unix нинг кўплаб версиялари, шунингдек Linux ва ҳатто Macintosh бўлиши мумкин. PHP кўплаб тармоқ серверларида, хусусан Apache, Microsoft Internet Information Server, Web Site Pro, Iplanet Web Server ва Microsoft Personal Web Server – ларда ишлаши мумкин. Агар ўзимиз тузган дастурларимизни Windows тизимида текширишни хохласак охирги санаб ўтилган сервердан фойдаланишимиз мумкин, ҳатто Apache сервери Windows системаси бошқарувида ишласа ҳам.

PHP интерпретатори ёрдамида дастурни алоҳида мустақил кўринишда компилясия қилиш мумкин. У ҳолда дастурни мустақил ишга тушириш мумкин. PHP тилини яратишда маълумотлар базаси билан боғланиш талабларини алоҳида эътиборга олинган. Кўплаб маълумотлар базаларини PHP да ўқиш мумкин. Масалан буларга Adabas D, InternetBase, Golid, dBase,

mSQL, Sybase, Empress, MySQL, Velosic, FilePro, Oracle, Unixdbm, Informix ва ҳоказоларни келтиришимиз мумкин. Шунингдек, PHP ODBC стандартини ҳам ўқий олади. Ушбу қўлланмада эса биз Linux, Apahe ва MySQL серверлар асосида фикр юритамиз. Бу учта дастурлар мажмуалари кенг фойдаланиш мумкин бўлган

PHP4 ни http://www.php.net тармоғидан олиш мумкин. Бунинг учун эса, алохида кредит карточка талаб қилинмайди.

PHP ни web-тармоғи прогромистлар учун ажойиб манба ҳисобланади. http://www.php.net/manual адресида бошқа прогромистларнинг изоҳлари, фикрлаш ва танқидий фикрлари келтирилган маълумотларни ва қўлланмаларни олиш мумкин. Бу қўлланмаларни турли хил форматларда олиш мумкин.

PHP.ini файли. PHP ни ўрнатиб бўлгандан сўнг, унинг настройкасини PHP.ini файли ёрдамида ўзгартириш мумкин. Агар компютерга UNIX операцион системаси ўрнатилган бўлса, бу файл /usr/local/lib катологида, агар Windows операцион системаси ўрнатилган бўлса, Windows катологида жойлашган бўлади.

Агар дастурлаш тили омма учун (текин) бўлса, албатта ёрдамни Internet дан етарлича топиш мумкин. PHP тили бўйича сиз билан фикр алмаша олувчи яна бир яхши манба бўлиб, у PHP билимлар базаси, яни knowledge Base деб номланади, ва уни http://www.fagts/com/knowledge-base/index.phtml деб номланади.

Биринчи дастур.

Биз юқорида PHP ни ўрнатиш ва ўрганиш бўйича бошланғич маълумотларни олдик. Энди эса, PHP да биринчи дастуримизни яратишга ҳаракат қиламиз. PHP да дастур тузиш учун HTML даги каби WEB-саҳифа яратишни билишимиз лозим.

Ушбу дастурда эса, куйидагиларни ўрганишга ҳаракат қиламиз:

PHP-программаларини яратиш, серверга нусхасини кўчириш ва ишга тушириш PHP командалари ва HTML матнини битта ҳужжатга боғлаш, Дастур матнини тушинтириш учун изоҳ келтириш

PHP-дастурни тузиш учун ихтиёрий матн редакторини ишга туширамиз. PHP – дастури HTML- ҳужжатлари каби оддий матндан ташкил топади. Шунинг учун дастурни ихтиёрий матн редакторида, агар unix бўлса, VI-ёки Emacs да ҳам ёзиш мумкин. қуйидаги дастур матнини киритайлик ва уни first.php деб сақлайлик:

```
< ? php
  print “hello web”;
?>
```

ушбу дастур кенгайтмаси албатта. PHP бўлиши шарт. Агар дастур сервер эмас, балки клетн компютерида тузилган бўлса, дастур файлини серверга киритиш учун FTP сервисдан фойдаланишга тўғри келади. Агар дастурда хатоликлар мавжуд бўлмаса, у ҳолда натижани браузер ойнасида кўриш мумкин.

Агар дастурда ёки унинг кенгайтмасида хатолик мавжуд бўлса, браузер ойнасида дастур матнининг ўзи намоён бўлади.

Жадвалда келтирилган стандарт ва дастурий теглар PHP нинг ихтиёрий конфигурациясида ишлайди. қисқа ва ASP теглар ишлаши учун php.ini файлида аниқ келтирилган бўлиши лозим. қисқа теглар фойдаланилиши учун php.ini файлида қуйидаги директива ёзилган бўлиши лозим:

Php.ini файлини тахрир қилиб, ихтиёрий директивадан фойдаланиш мумкин. қуйидаги барча теглар учун ёзилган дастур матнини кўрайлик:

```
<?
print “hello web”;
?>
<? php
print “hello web”;
```

```
?>
<%
print “hello web”;
%>
<script languageк”php”>
print “hello web”;
</script>
```

print () функцияси

Бу функция маълумотларни экранга чоп қилиш учун фойдаланилади. Экранга бериладиган маълумотлар () ичига берилади. Экранга бериладиган ҳар бир маълумот “ ” ичига берилиши лозим. Шунингдек Print дан кейин () қўйиш ҳам, қўймаслик ҳам мумкин.

HTML ва PHP нинг биргаликда ишлатилиши. Қуйида келтириб ўтиладиган дастур фақат PHP командаларидан ташкил топган. Лекин очувчи ва ёпувчи теглар қўшиш орқали HTML - текст қўшиб HTML ва PHP нинг аралаш ҳужжатнинг яратиш мумкин:

```
<html>
<head>
  <litle> php ва html ҳужжати </litle>
</head>
<body>
<b>
<?php>
  print “hello worlb!”;
?>
</b>
</body>
</html>
```

Агар дастурни сақлаб (.PHP) уни браузер орқали ишга туширсак, экранда (жирнўй) hello world! ҳосил бўлади.

PHP-дастурида коментариялар (изоҳлар)

Маълумки, дастурни тузиш жараёнида ҳамма нарса тушунарли ва оддий дастурни ҳам тушуниб олиш қийин бўлиб қолади. Шунинг учун ҳам, дастур ёзиш жараёнида албатта изоҳ келтириб бориш зарур, чунки сизни дастурингиздан фойдаланувчига бироз осонроқ бўлади.

Коментария (изоҳ) - бу интерпретторда қайта ишланмайдиган, дастурдаги оддий матн ҳисобланади. Изоҳ дастурни тушиниш учун енгиллик яратиш мақсадида фойдаланилади.

Ўзгармаслар. Ўгарувчилардан фойдланиш маълумотларни сақлаш ва қайта ишлашнинг энг ажойиб усулларидан бири ҳисобланади. Биз бу ўзгарувчилар қийматини, ҳаттоки типини хохлаган вақтимизда ўзгартира олишимиз мумкин. Лекин дастур яратиш жараёнида шундай ҳолат бўлишини истаб қоламизки, бунда бизни берган қийтимиз ёки маълумотимиз дастурни яратиш ва фойдаланиш жараёнида, керак бўлса тасодифан ўзгариб кетишдан ҳоли бўлсин, яъни ўзгармасин. Бунинг учун ўзгармаслардан фойдаланишимиз зарур. PHP4 да ўзгармаслар билан ишловчи define() функцияси манашу максадда ишлатилади. Бу функция ёрдамида яратилган ўзгармас (константа) дастурни яратиш ва фойдаланиш жараёнида ўзгартирилмайди. Ўзгармасни ташкил этиш учун define() функциясида ўзгармас номи ва унинг қиймати берилади.

Define (“nomi”,42)

Ўзгармас қабул қиладиган маълумот қиймат ва символ бўлиши мумкин. Ўзгармас номи эса факат катта щарфлар билан берилади. Ўзгармасдан фойдаланишда фақат катта ҳарфлар билан $ белгисини олдига қўймасдан ёзилади.

Куйидаги ўзгармасдан фойдаланиш дастурда қандай ишлатилишига мисол сифатида келтириб ўтамиз:

```
<html>
<head>
<title>Uzgarmaslarni tashkil etish</title>
</head>
<body>
<?php
define("USER",Gerald);
print "Welcome".USER;
?>
</body>
</html>
```

Агар дастурга аҳамият берсак, «Welcome» ёзувидан сўнг конкатенация (.) белгисидан фойдаландик.

PHPда яна бир имконият борки, у фойдаланувчига автоматик равишда бир нечта ўзгармас ташкил этиб беради. Масалан, _FILE_ ўзгармаси факат фойдаланилаётган актив файл номини ўзида сақлайди. _LINE_ ишлатилаётган файлдаги актив каторни аниклаб беради ва х.к.

Маълумки, жараёнлар чизикли, такрорланувчи ва тармокланувчи бўлиши мумкин. Хозиргача урганган ва курган мисолларимиз факат чизикли жараёнга тегишли эди. Бу мавзунинг асосий максади эса, PHP4 тилида тармокланувчи ва такрорланувчи жараёнилар учун қандай функция, процедура ва инструкциялардан фойдаланилиши билан таништиришдан иборат. из асосан куйидагиларни урганамиз:

if инструкцияси ёрдамида дастурдаги баъзи бир шартларни қандай текширилиши ва бажарилиши;

Шарт бажарилмаган кисм учун дастурнинг алтернатив бўлимини =андай яратиш;

Switch ёрдамида баъзи бир ифодаларни қийматини эътиборга олган ҳолда дастур қисмини қандай яратиш ;

While ёрдамида циклик жараённи ташкил этиш;

For ёрдамида циклик жараёниларни ташкил этиш;

Циклик жараёнларни тухтатиш қандай бажарилади;

Бир неча циклик жараёнларни биргаликда қандай ташкил этиш.

PHP4 да шартли жараёниларни ифодалаш. Жуда куп дастурлар маълум бир шартлар сабабли узининг ҳолатини ўзгартириб туради. Маълум бир аник шартлар асосида дастурнинг такомиллашувчи яратилаётган Web саҳифаларнинг динамиклигини янада мустахкамлайди. Бошка дастурлаш тиллари каби PHP4 тили ҳам шартли инструкциялардан фойдаланади. Бу инструкция PHP4 да if ёрдамида амалга оширилади.

IF операторидан фойдаланиш. IF инструкциясидан фойдаланилаёт-ганда ҳисоблаш жараёни учун шарт кавслар ичида текширилади. Агар шарт бажарилса (true бўлса) блок ичидаги ифода бажарилади, акс ҳолда блок ичидаги ифода тулик ташлаб юборилади. IF инструкциясининг умумий куринишдаги схемасини куйидагича келтириш мумкин:

```
IF (шарт)
{
// бу ердаги ифода агар берилаётган шарт рост бўлса бажарилади
}
```

Келтириб утилган умумий схемани янада аникрок тушуниб олиш учун куйидаги мисолни келтириб ўтайлик.

```
<html>
<head>
<title>IF ёрдамида шартли ифодаларни куллаш </title>
</head>
<body>
<?php
$mod="sad";
if ($mod="happy")
```

```
{
print "Менинг кайфиятим жуда ҳам яхши!"
}
?>
</body>
</html>
```

Оператор ёрдамида $mood Ўзгарувчи =ийматини “happy” катори билан солиштирамиз. Агар шарт бажарилса, блок ичидаги ёзув экранга берилади, акс ҳолда берилмайди. Келтириб утилган мисолда блок ичидаги инструкция куп эмас, шунинг учун бу блокни янада ихчамрок куринишда куйидагича келтириш мумкин:

```
if ($mod="happy")
print "Менинг кайфиятим жуда ҳам яхши!";
```

IF инструкциясининг else блоки. IF инструкциясини келтириш жараёнида бериладиган шарт бажарилмаса альтернатив блок келтиришга мажбур бўлиб колинади. Буннинг учун эса, if шартдан кейинги блокдан сунг else блокини ҳам бериш лозим. Умумий ҳолда бу ҳолатни куйидагича келтириш мумкин:

```
If (шарт)
{
//бу блокдаги ифодалар шартни текшириш натижаси рост булганда бажарилади
}
else
{
// бу блокдаги ифодалар шартни текшириш натижаси рост булганда бажарилади
}
```

Келтириб утилган умумий схемани янада мустахкамлаш учун куйидаги мисолни куриб ўтайлик:

```
<html>
<head>
<title>IF ёрдамида шартли ифодаларни куллаш ва else блокини текшириш </title>
</head>
<body>
<?php
$mod="sad";
if ($mod="happy")
{
print "Менинг кайфиятим жуда ҳам яхши!"
}
else
{
print “Менда $mood”;
}
?>
</body>
</html>
```

$mood Ўзгарувчиси бу мисолда sad каторини кабул килади, бу эса “happy” катори билан бир хил эмас, шунинг учун ҳам шарт бажарилмайди. Бу эса, биринчи блокни бажарилмаслигини билдириб, кейинги блокда келтириб утилган ифодани экранга берилиши билдиради.

IF инструкцияда келтириладиган else блоки мураккаб масалаларни ечиш имкониятини беради. PHP4 тилида эса, янада мураккаброк булган шартларни текшириш имкониятлари мавжуд бўлиб бу шартларни биз кейинги мавзуларда келтириб ўтамиз.

Функция. Функция – бу яхши ишланган дастурнинг энг асосий компоненти ҳисобланади. Улар яратилаётган дастурнинг тушунарлилигини жуда ҳам яхши бўлишига ва фойдаланишга қулай бўлишига катта ёрдам беради. Ихтиёрий яратилаётган дастур унинг ёрдамисиз бирор катта натижага эришган эмас. Айтмоқчимизки, функциялар дастурлаш билан шуғулланишни бошлаганлар ва давом эттираётган учун жуда ҳам зарур ва қулай ҳисобланади. Функциялар ва уларнинг ёрдамида кўп ишлатилувчи бир хил типдаги ўзгарувчилар устида кўп марталик ҳисоблашларни қандай қилиб ихчамлаштириш мумкинлигини ўрганамиз.

Функцияларни қандай яратиш ва уларга мурожаат қилиш, функция параметрларига қийматларни қандай узатиш ва қайтарилган қийматларни қабул қилиб олиш, матнли ўзгарувчида унинг номини сақлаган ҳолда қандай қилиб функцияга динамик мурожаат қилиш , глобал ўзгарувчиларга функциядан туриб қандай мурожаат қилиш, «Эслаш» функцияларини қуриш, кўрсатма бўйича функция аргументини узатиш каби нарсаларни ўрганамиз.

Функция нима? Функцияни шундай қурол сифатида тасаввур қилиш мумкинки, биз унга бошланғич маълумотларни яъни хом ашёларни бериб тайёр махсулот олишимиз мумкин. Функция биздан бошланғич бир ёки бир неча маълумотлар, қийматлар олади ҳамда ўзида уларни биз қўйган мақсадимизга қараб қийматларни натижа сифатида қайтаради. Шунинг учун ҳам функция мақсадга боғлиқ равишда ихтиёрий вазифани бажарувчи инструмент ҳисобланади.

Масалан, оддий мисол сифатида уйда печенье пиширишни олиб қарайлик. Агар бир-икки марта пиширсак уни қўлда тайёрлашимиз мумкин. Лекин, агар бозор учун хар куни тайёрлайдиган бўлсак, у ҳолда унга мос равишда автомат қидириб қоламиз. Ушбу мисол асосида шундай савол туғилади. «Бу функция бизни бажармоқчи бўлган ишимиздаги такрорланишлар учун қандай аҳамиятга эга?». Бу эса жараённи қонуний ва кўп марта такрорланиши билан боғлиқ.

У ҳолда функцияни дастуримизда мурожаат қилиш мумкин бўлагн алоҳида номдаги дастур сифатида тасаввур қилишимиз мумкин. Бу дастурий функция фақат мурожаат қилингандагина ишлайди. Функция қайтарган қийматаларни биз натижа сифатида чоп қилишимиз мумкин ёки дастур давомида фойдаланишимиз мумкин.

Янги термин Функция – бу интсрукция ва операторлардан иборат қисм дастур бўлиб, у асосий дастурларда зарур вақтларда фойдаланилади. Функциялар тайёр қурилган ёки дастурчи томонидан қурилган бўлиши мумкин. Улар баъзи бир бошланғич маълумотларни ўзида фойдаланиш учун сўраши ва натижавий қийматни қайтариши мумкин.

Функцияларга мурожаат. Функция икки хил бўлиши мумкин. Булар тилнинг компиляторида мавжуд функциялар ҳамда дастурчи томонидан яратиладиган функциялар. Биринчи кўринишдаги функция хақидаги деярли маълумотга эгамиз. Чунки юқорида келтириб ўтган print (“Hello, Web”); функиямиз компиляторга тегишли. Лекин print() функцияси оддий функция ҳисобланмайди, чунки бу функция ўзининг аргументи учун қавсларни қўйиш шартини қаттиқ талаб қилмайди. У холда print (“Hello,Web”) ва print “Hello,Web” функциялар эквивалент конструкциялар ҳисобланади. Лекин фақат шу функциягина бундай кўринишга эга бўла олади. Қолган барча функциялар қавслар қўйилишини ҳатто аргумент талаб қилмаганда ҳам талаб қилади.

Функция албатта ўзининг номига эга бўлиши лозим. Юқорида келтириб ўтган функциямиз номи эса print ва унинг аргументи қавс ичида берилади. Функцияда бир ёки бир неча аргументларнинг бўлиши мумкин. Агар кўп бўлса улар бир-биридан вергал(,) билан ажратиб ёзилади. Уларни умумий ҳолда қуидагича келтиришимиз мумкин:

Some-function ($argument-1, $argument-2); ихтиёрий функцияалбатта қиймат қайтаради. Бунга сабаб эса, функция бажарган ишнинг муваффақиятли якунини аниқлаш бўлади.

Масалан, abs() функцияси соннинг абсолют қийматини қайтаради. Бунинг учун у мианфий сон қабул қилиб, мусбат қиймат қайтаради. Бу функцияга эса, қуйидаги дастур мисол сифатида келтиришимиз мумкин.

Мисол:

```
<html>
<head>
<title> abs() функциясини чакириш</title>
</head>
<body>
<?php
  $num=-321;
  $newnum=abs($num);
print $newnum;   // натижа «321»
?>
</body>
</html>
```

Бу мисолда $ пит ўзгарувчисига – 321 қийматни ўзлаштирамиз. Кейин эса, abs() функция аргументига узатиб, шу функция олган натижа $ new питни ўзгарувчига узатилади. Бу ўзгарувчидаги натижани print() орқали чоп қилиб оламиз. Аслида эса, дастурни янада ихчамроқ бўлиши учун $newnum ўзгарувчисидан фойдаланмай print(abs(-321)) беришимиз ҳам мумкин.

Янги термин Функция аргументи - бу сиз функцияга мурожаат қилганимиздаги берадиган қийматларингиз бўлиб, улар қавс ичида берилади. Функцияни яратишда аргумент номи шарли равишда берилади, сўнгра эса, бу аргументдан функция ўзагида фойдаланиш мумкин.

Функцияларни яратиш. Функцияларни яратиш учун function калит сўзидан фойдаланилади, ҳамда унинг умумий схемаси қуйидагича бўлади:

```
Function some_func (#argument_1, #argument_2);
{
```

```
  //функция узаги
}
```

Функция номи function сўзидан кейин берилади ва қавслар ичида аргументлар келтирилади. Агар яратилаётган функция щеч қандай аргумент талаб қилмаса ҳам қавслар қўйилиши шарт.

Келтириб ўтган фикрларимизни янада мустащкамлаш учун қуйидаги мисолни келтириб ўтайлик:

Мисол:

```
<html>
<head>
<title> функцияни аниклаш</title>
</head>
<body>
<?php
function bighello()
{
  print “<h1>HELLO</h>”;
}
bighello()
?>
</body>
</html>
```

Бу дастур браузер ойнасига фақат «HELLO» ёзувини чиқариб беради. Дастурда эса, функция номини bighello() деб номладик, ҳамда бу функция щеч қандай аргумент талаб қилмайди. Бу мисолда келтириб ўтилган функция барча талабларга тўғри келади.

Энди эса, аргумент талаб қилувчи қуйидаги мисолни келтириб ўтайлик:

Мисол:

```
<html>
```

```
<head>
<title> Аргумент талаб килувчи функцияга мисол</title>
</head>
<body>
<?php
function printBR($txt);
  {
  print (“txt”<br>\n);
  }
printBR(“Бу катор”);
printBR(“Бу кейинги катор”);
printBR(“Бу бир кейинги катор”);
?>
</body>
</html>
```

PrintBR() функцияси аргумент киритилишини талаб қилади. Аргумент типи эса, матнли бўлади ва унинг номи $txt деб аталган. Энди биз бу функция ёрдамида ихтиёрий ёзувни браузерга чоп қилишимиз мумкин.

Қиймат қайтарувчи функциялар яратиш. Функция қиймат қайтариши ёки return оператори ёрдамида объект қайтариши мумкин. Бу оператор функция ишини тугатади ва қайтарилиши лозим бўлган қийматни чақирилган бош дастурга қайтаради.

Бу фикрларни мустахкамлаш мақсадида 2 соннинг йиғиндисини ҳисоблашга доир қуйидаги мисолни келтириб ўтайлик.

Мисол: қиймат қайтарувчи функция

```
<html>
<head>
<title> +иймат қайтарувчи функция </title>
</head>
```

```
<body>
<?php
function addNums ($firstnum, $secondnum);
  {
  result=$firstnum+$secondnum);
  }
print addNums(3,5);  // натижа 8 чикади
?>
</body>
</html>
```

Бу дастурнинг бажарилиши натижасида браузерда 8 чоп қилинади. Чунки add nums() функциясининг аргументлари 3 ва 5 қийматларни қабул қилади. Аввалроқ айтиб ўтганимиздек, дастурни янада ихчамроқ кўринишга келтиришимиз мумкин. У ҳолда функциянинг ўзаги қуйидаги кўринишга эга бўлади.

```
{
return ($firstnum+#secondnum);
}
```

return оператори қиймат, объект қайтариши ёки хеч раса қайтармаслиги ҳам мумкин. Return оператори ёрдамида қиймат қайтаришнинг бир неча усуллари мавжуд бўлиб улар:

```
return4;                                -ўзгармас;
return($a/$b);                          -ифода ёки
return(function($argument))           - бўлиши мумкин.
```

Функцияларга динамик мурожаат қилиш. Функция номини бирор матнли типга эга бўлган ўзгарувчига ўзлаштириш мумкин, ҳамда худди шу ўзгарувчига худди функция мурожаат қилгандек ишлаш мумкин.

Фикримизни янада аниқроқ тасаввур қилишимиз учун эса, қуйидаги мисолни келтириб ўтайлик:

Мисол:

```
<html>
<head>
<title> Функцияларга динамик мурожаат</title>
</head>
<body>
<?php
function sayHello( );
  {
  print “Hello <br>”;
  }
$function_holder=”sayHello”;
$function_holder();
?>
</body>
</html>
```

Бу ерда $function-holder ўзгарувчиси матнли типга эга ва у Sayhello() функцияси номини ўзлаштиради. Кейин эса, худди функцияга мурожаат қилгандек $function-folder() га мурожаат қилишимиз мумкин.

Албатта, табиий савол туғилади: «Функцияга мурожаат қилишнинг бу усулидан нима наф бор?» Бу мисолда биз ортиқча иш бажардик. Лекин бошқа катта дастурларда бу усулнинг ютуғи катта бўлади. Масалан, фойдаланувчи талабига асосан дастур холатини ўзгартиришга тўғри келади. Шунда биз қаторли параметр (ўзгарувчига) ўзлаштирилган функиця номи бўйича ўзгартиришимиз мумкин.

2.2. MySQL маълумотлар омборини бошқариш тизими.

PHP тилининг энг ажойиб томонларидан бири, дастурчи маълумотлар базаси билан ишлашда жуда ҳам кўплаб қулайликларга эга бўлади. Бу маъруза давомида биз асосан MySQL системаси ҳақида фикр юритамиз. Лекин, албатта PHPнинг бошқа маълумотлар базаси асосида ишловчи системалар билан алоқасини ҳам бироз бўлсада кўриб ўтамиз. «Нима учун айнан PHP билан MySQL ўртасида ўзаро алоқани ўрганишимиз лозим.Бу система маълумоларни бошқариш системаси ҳисобланиб, у PHPнинг энг асосий манбаси ҳисобланади, ҳамда бу система бепул тарқатилади ва фойдаланишга ҳамда ўрганишга жуда ҳам қулай ҳисобланади. Булардан ташқари MySQL база билан ишловчи системани жуда ҳам кўплаб платформалари мавжуд. Агар бу система компьютеримизда мавжуд бўлмаса, биз уни http:/www/mysql.org. тармоғидан олишимиз мумкин. Мавзуни ўзлаштириш давомида эса, асосан қуйидагиларни ўрганамиз:

SQL инглиз тилидаги Structured Query Language ёки ўзбек тилидаги сўровлар асосидаги структурали тил сўзларининг қисқартмасидан ташкил топган. Бу тил турли хилдаги маълумотлар базасига мурожаат қилиш учун стандарт имкониятларга эга бўлган тил ҳисобланади. Кўплаб базалар бу тил асосида ўзининг версияларига ҳам эга. Лекин шунга қарамасдан SQL турли хил кўринишдаги маълумотлар базалари билан ишлаш имконини бера олади. Бу маърузада биз SQL билан ишлашнинг деярли барча имкониятларини кўриб ўтмасакда, асосий тушунчалари, SQL тилининг бир қисми ҳисобланган MySQL маълумотлар базаси билан ишлашни ўрганамиз. MySQL тизими бошқа (удаленный) компьютерлардаги фойдаланувчилар билан боғланиши мумкин бўлган серверга эга бўлади. Серверга боғланиб биз ўзимиз учун керак бўлган базани топишимиз ва у билан ишлай олишимиз мумкин.

Базадаги маълумотлар ҳамиша қандайдир жадвалга эга бўлган кўринишда бўлади. Ҳар бир жадвал эса, устун ва қаторлардан иборат бўлиб, ҳар бир ячейкада маълумотлар сақланади. Ҳар бир устунда эса, битта типга

тегишли маълумотларни ёзилади. Масалан, бутун сонлар (тип INT) ёки қаторли типдан (тип VARCHAR) иборат бўлиши ҳам мумкин.

Танланган маълумотлар базасида янги жадвал ҳосил қилиш учун қуйидагича SQL сўровдан фойдаланиш лозим:

CREATE TABLE mytable (first_name VARCHAR(30), second_name VARCHAR(30), age INT);

Бу янги жадвалда учта устун ҳосил бўлади ёки қуйидаги номга эга бўлган майдонлар ҳосил қилинади: first_name ва second_name номли майдонлар 30 тагача символдан иборат элементларни олса, age майдонида эса, бутун сонлар сақланади.

Бу жадвалга маълумот қўшиш учун INSERT инструкциясидан фойдаланиш лозим:

INSERT INTO mytable (first_name, second_name, age) VALUES (‘John’, ‘Smith’, 36);

Биз қўшаётган маълумотларнинг майдони номлари биринчи қавсда, шу майдонга киритиладиган ёзувлар эса иккинчи қавсда берилади.

SELECT * FORM mytable;

Жадвалдаги барча маълумотларга мурожаат қила олиш учун SELECT инструкциясидан фойдаланиш лозим.

SELECT age FROM mytable;

* белгиси бюизга барча майдонлар кераклигини билдиради. Агар бизга фақат баъзи бир майдонлар лозим бўлса, SELECT инструкциясида шу номларни барчасини беришимиз лозим бўлади.

Жадвалдаги аввал ёзилган маълумотларни ёки қийматларни ўзгартириш учун UPDATE инструкциясидан фойдаланилади.

UPDATE mytable SET first_name=’Bert;

Натижада first_name майдонининг барча ёзувлар ўзгаради ва жадвалнинг барча қаторларида Bert ёзуви ёзилади. SELECT ва UPDATE

инструкцияси асосида ёзилган барча ёзувларни WHERE ёрдамида чиқариш мумкин:

SELECT * FROM mytable WHERE first_name=’Bert’;

Бу инструкция Bert ёзилган майдонларнинг барчасини аниқлаб беради. Қуйида келтириб ўтадиган мисолимизда агар иккинчи майдон second_name да ‘Baker’ ёзувчи ёзилган бўлса, биринчи майдондаги ёзувларни ўзгартириб беради.

UPDATE mytable SET first_name=’Bert’ WHERE second_name=”Baker”;

Маълумотлар базаси серверига боғланиш. Ўзимизнинг алоҳида маълумотлар базамиз билан ишлашимиз учун албатта серверга боғланишимиз лозим. Бунинг учунг PHPда mysql_connect() функциясидан фойдаланилади. Бу функция 3 та аргументга эга бўлиб, улар компьютер номи, фойдаланувчи номи ва пароль ҳисобланади. Агар бу аргументларни ташкаб юбормоқчи бўлсак, у ҳолда localhost компьютер талаб қилинади. Унда фойдаланувчи номи ва унинг пароли талаб қилинмайди, ҳамда бу компьютерда mysqluser ўрнатилмаган бўлади.

Албатта, бундай боғланиш катта аҳамиятга эга бўлмайди ва фақат сервер мавжудлигини аниқлаб олиш мумкин холос. Шунинг учун ҳам келгусида биз фойдаланувчи номи ва паролни бериб ишлашга ҳаракат қиламиз. Mysql_connect() функцияси агар барча ишлаш тўғри якунланса, боғланиш идентификаторини қайтаради. Бу идентификаторни бирор ўзгарувчида келгусида фойдаланиш учун сақлаб қўйиш мумкин.

Қуйидаги мисол маълумотлар базаси серверига боғланишга оид мисолни келтириб ўтамиз:

```
$link=mysql_connect(“localhost”, “root”, “toor”);
if (! $link)
    die(“Couldn’t connect to MySQL”);
```

Агар PHP ни Apache сервери билан бирга ишлаётган бўлса, сервердаги маълумотлар базаси билан боғланиш учун mysql_pconnect() функциясидан

фойдаланилиши мумкин. Дастурчи нуқтаи назаридан бу функцияларнинг вазифаси бир хил бўлсада, лекин бироз бўлсада фарқи мавжуд. Агар биз mysql_pconnect() функциясидан фойдаланаётган бўлсак, сервер билан боғланишда узилиш ҳисобига ҳеч нарса йўқотилмайди ёки mysql_close() функциясини ишлатмасак ҳам ортиқча йўқотишлар рўй бермайди.

Маълумотлар базасини танлаш. MySQL сервери билан боғланиб олингандан сўнг, ишламоқчи бўлган базамизни танлаб олишимиз лозим. Бунинг учун MySQL_select_db() функциясидан фойдаланилади. Бу функцияга базанинг номи ва иккинчи унча зарур бўлмаган аргумент, яъни серверга боғланиш идентификатори берилади. Агар иккинчи аргумент берилмаса, у ҳолда охирги боғланилган идентификатор ишлатилади. Агар функция «true» қиймат қайтарса, у ҳолда база мавжуд ва унга мрожаат қилиш мумкин. Қуйидаги мисолда sample номли базани танлаймиз:

$database=”sample”;

MySQL_select_db($sample) or die (“Couldn/t open $sample”)

Жадвалга маълумот қўшиш. Маълумотлар базасига мурожаат қилиш учун боғланиб бўлганимиздан сўнг, энди унга керакли маълумотни қўшиш имкониятига эга бўламиз. Буни мисол асосида кўриб ўтайлик. Айтайлик, фойдаланувчилар ўзлари учун домен сотиб олишлари имконини берувчи тармоқ яратмоқчимиз.

Биз sample номли маълумотлар базасида domains номли жадвал яратамиз. Жадвал майдонга эга бўлсин: янги ёзув қўшилганда қиймати автоматик равишда битта кўпаювчи id номли калит ҳисобланган биринчи майдон, VARCHAR типли қаторли ўзгарувчи ёзиладиган domain майдони, битта символдан иборат jins майдони ва фойдаланувчи адресини ўзида сақловчи mail майдонларига эга бўлади. Бу жадвални яратиш учун SQLнинг қуйидаги инструкцияси ишлатилади:

Create table domains (id INT NOT NULL AUTO_INCREMENT,

PRIMARY KEY(id),

```
Domain VARCHAR(20),
Jins CHAR(1)
Mail VARCHAR(20));
```

Бу жадвалга маълумот қўшиш учун SQL сўровдан фойдаланишимиз лозим. Бунинг учун PHP тилида махсус MySQL_query() функцияси мавжуд. Бу функцияга сўров қаторини ва унчалик катта аҳамиятга эга бўлмаган боғланиш идентификаторини берилиши лозим. Агар сўров тўғри бажариласа функция true қиймат қайтаради, акс ҳолда бундай сўров бажаришга рухсат берилмаган бўлса ёки қандайдир ҳатолик мавжуд бўлса false қиймат қайтаради. Сўровнинг тўғри бажарилиши жадвалдаги маълумотларни ўзгариб кетишини олдини олади. Қуйидаги мисолимизда аввал келтириб ўтилган мисол янада кенгайтирилади ва MySQL_query() функцияси sample маълумотлар базасидаги domains жадвалга маълумотларни қўйишда INSERT инструкцияси бажарилади.

Жадвалга ёзув қўшиш.

```
<html>
<head>
  <title> Жадвалга ёзув қўшиш </title>
</head>
<body>
<?php
$user=”harry";
$pass=”elbomonkey";
$db=”samle";
$link=mysql_connect("localhost", $user, $pass);
if (! $link)
die("MySQl база билан богланиш йук");
mysql_select_db($db,$link)
or die ("Базани очиб булмади:".mysql_error());
```

```
$query=”Insert into domains (domain,sex,mail)
values (123xyz.com,'F','sharp@adomain.com')";
my_sql_close($query,$link) or die ("Couldn`t add data to \"domains\"table:"
  .mysql_error());
  my_sql_close($link);
?>
</body>
</html>
```

Агар дастурга аҳамият берсак, унда биз id майдонига ҳеч қандай маълумот киритмаяпмиз, балки у ўзи автоматик равишда ошиб бормоқда.

Ҳақиқатан ҳам, бу дастурнинг ҳар сафар бажарилишида янги ёзув қўшилади ва ёзув биттага кўпаяди. Қуйидаги дастурда эса, фойдаланувчи томонидан киритилган маълумотларни жадвалга қўшиш бажарилади.

Келтириб ўтилган дастурда оддийлик ва ихчамлик учун бир зарур вазиятни ёзмай кетдик, яъни фойдаланувчи киритиган маълумотни текширмадик. Бундай қилиш эса, аслида ярамайди, яъни фойдаланувчи киритаётган маълумотга тўлиқ ишонмаслик олзим. Формага киритилган барча маълумотни албатта текшириш лозим.

Дастурда эса, асосан $domain, $jins ва $mail ўзгарувчилари мавжудлигини текширилади. Агар бу ўзгарувчилар мавжуд бўлса, демак add_to_database() функциясига мурожаат қилишимиз мумкин.

Add_to_database() функцияси туртта аргументга эга бўлиб, улар: фойдаланувчи томонидан берилган $domain, $jins ва $mail ўзгарувчилар ва $dberror қатор. Бу қаторга хатолик юз берса, шу хатолик ҳақидаги ахборот берилади. Шунинг учун бу ўзгарувчини йўналиш бўйича беришимиз мумкин. Функция ичидаги ўзгарувчи устида бўладиган барча ўзгаришлар унинг нусхасига эмас, балки фақат шу ўзгарувчининг ўзигагина тегишли бўлади. Дастурда биз MySQL сервери билан боғланамиз, агар боғланиш юз бермаса, false натижа қайтарилади ва дастур иши тўхтатилади. Агар боғлансак,

маълумотлар базасидан domains жадвали билан боғланамиз ва шу жадвалга маълумот кирита олувчи SQL – сўров ҳосил қиламиз. Бу сўров mysql_query() функцияси орқали бажарилади. Агар mysql_select_db() ёки mysql_query() функцияларидан бирортасида хатолик пайдо бўлса, $dberror функциясида бу хатолик ҳақидаги маълумот берилади ва false қиймат қайтарилади. Агар ҳаммаси жойида бўлса, функция true қиймат қайтаради.

Дастурни ўзида add_to_database() функцияси қайтарган қийматни текшириш имкони ҳам бор. Агар функция true қиймат қайтарса, демак базага маълумот қўшилганлигини билдиради. Агар false бўлса, демак қандайдир хатолик рўй берганлигини билдиради. У ҳолда хатолик ҳақидаги ахборотни браузеройнасига чоп қиламиз. Хатолик ҳақидаги ахборотни $dberror ўзгарувчида берилиши биламиз, шунинг учун бу ўзгарувчини маълумот чиқариш қаторига қўшиб қўямиз.

Агар дастур бошидаёқ келтириб ўтилган учта ўзгарувчилардан бирортаси мавжуд эмаслигини аниқласак, у ҳолда фойдаланувчи томонидан маълумот берилмаганлигини билдиради. Бундай ҳолатда write_form() функциясидан фойдаланамиз. Бу функция фойдаланувчи формасини браузер ойнасига чиқариб беради.

Автоматик ўзгарадиган майдондан қийматни олиш. Аввалги мисолларда жадвалга ёзувларни бемалол киритдик. Унда эса, id майдонидаги қийматлар ҳар сафар янги ёзув киритганимизда автоматик равишда ўзгариб борди. Энди шу id майдонидаги қийматларни формага SQL-сўровлар ёрдамида чиқаришимизлозим. Фақат қандай қилиб олиш мумкин? Бунинг учун PHP тилида mysql_insert_id() функцияси мавжуд бўлиб, бу функция охирги қўшилган ёзувнинг калит қийматини қайтаради.

У ҳолда фойдаланувчига унинг номери ҳақида ахборот беришимиз лозим бўлса, mysql_insert_id() функциясидан фойдаланиш лозим.

Масалан:

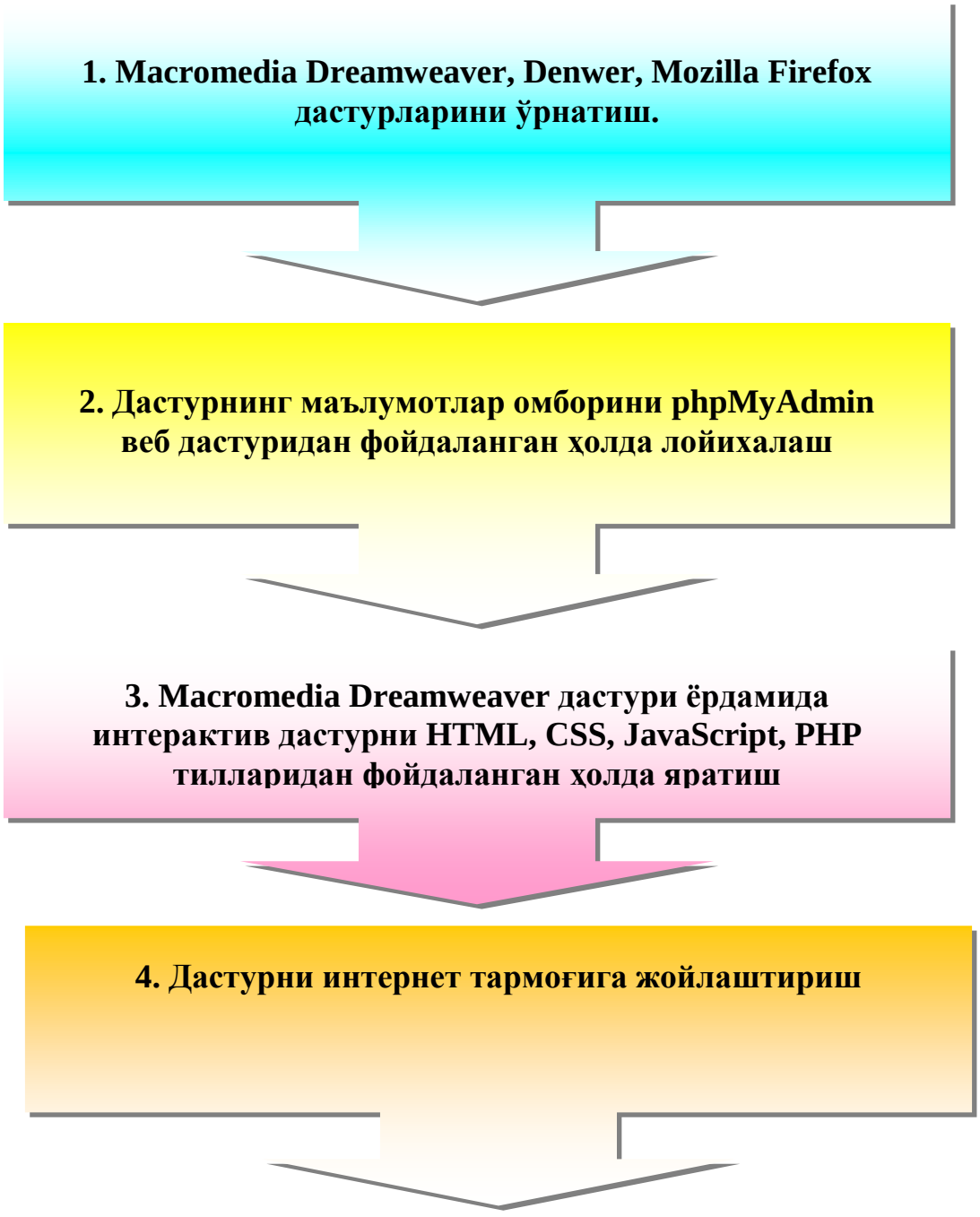
```
$query=”INSERT INTO domains(domain, jins, mail values (‘$domain’, ’jins’, ‘$mail’)”;
mysql_query($query,$link);
$id=mysql_insert_id();
print “Рахмат, сизнинг id номерингиз $id”;
```

Маълумотга мурожаат қилиш. Биз маълумотлар базасига ёзув қўшишни ўргандик, лекин уларни қандай ва қаердан ўқишни ҳам билишимиз лозим. Бунинг учун SELECT типида SQL-сўровдан фойдаланиш лозим. Mysql_query() функция сўровида хатолик рўй бермаса, у ҳолда маълумотни қайта ишлаш учун олинган идентификаторни бошқа функцияларга узатиш мумкин.

Сўровда топилган ёзувлар сонини аниқлаш. SELECT сўров бажарилиши натижасида ёзувлар сонини аниқлаш мумкин. Бунинг учун mysql_num_rows() функциясидан фойдаланилади. Бу функция сўров идентификатори берилади ва натижа сифатида эса ёзувлар сони қайтарилади. Қуйидаги мисолда domains жадвалидаги барча қаторлар сонини ва mysql_num_rows() функциясининг ўзи ёрдамида жадвалдаги аниқланган ҳажмни бериш келтирилади.

mysql_num_rows() функцияси сўров идентификатори натижасини ва шу жадвалдаги элементлар сонини аниқлаб, унинг соини қайтаради.

2.3. WEB ДАСТУРНИ СОЗЛАШ ВА УНДАН ФОЙДАЛАНИШ. PHP ВА MYSQL ЁРДАМИДА УНИВЕРСИТЕТ ТАЛАБАЛАРИ БИЛИМИНИ ТЕКШИРУВЧИ ИНТЕРАКТИВ ДАСТУРНИ ЯРАТИШ АЛГОРИТМИ



1. Macromedia Dreamweaver, Denwer, Mozilla Firefox дастурларини ўрнатиш.

Macromedia Dreamweaver дастурини шахсий компьютерга қуйидаги мақсадлар учун ўрнатамиз:

1. Динамик веб саҳифалар учун ягона дизанни ташкил этиш ва уни қўллаш.
2. Динамик веб сахифаларни кодлаштириш.
3. Динамик веб сахифаларни бошқариш ва уларни назорат қилиш.

Denwer дастури таркибига қуйидаги дастурий воситалар киради:

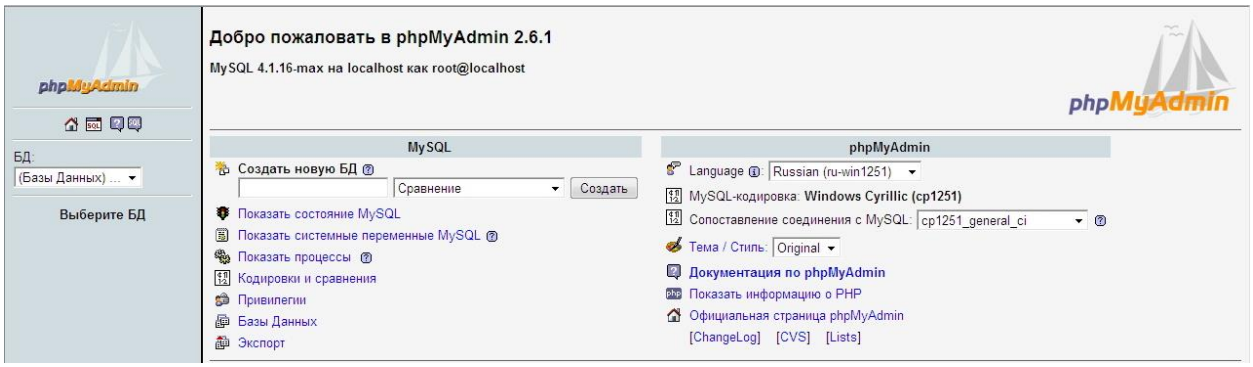
1. Apache Web Server
2. PHP Interpreter
3. MySQL Server
4. PERL Interpreter
5. PostgreSQL Server
6. Sendmail

Mozilla Firefox дастурини шахсий компьютерга қуйидаги мақсадлар учун ўрнатамиз:

1. Интерактив дастурдан фойдаланиш ва уни назорат қилиш.
2. Маълумотлар омборини бошқарувчи дастурдан фойдаланиш учун.

Denwer дастурининг охирги версиясини denwer.ru ва Mozilla Firefox дастурининг охирги версиясини www.mozilla.org веб сайтларидан юклаб олишингиз мумкин.

2. Дастурнинг маълумотлар омборини phpMyAdmin веб дастуридан фойдаланган ҳолда лойихалаш

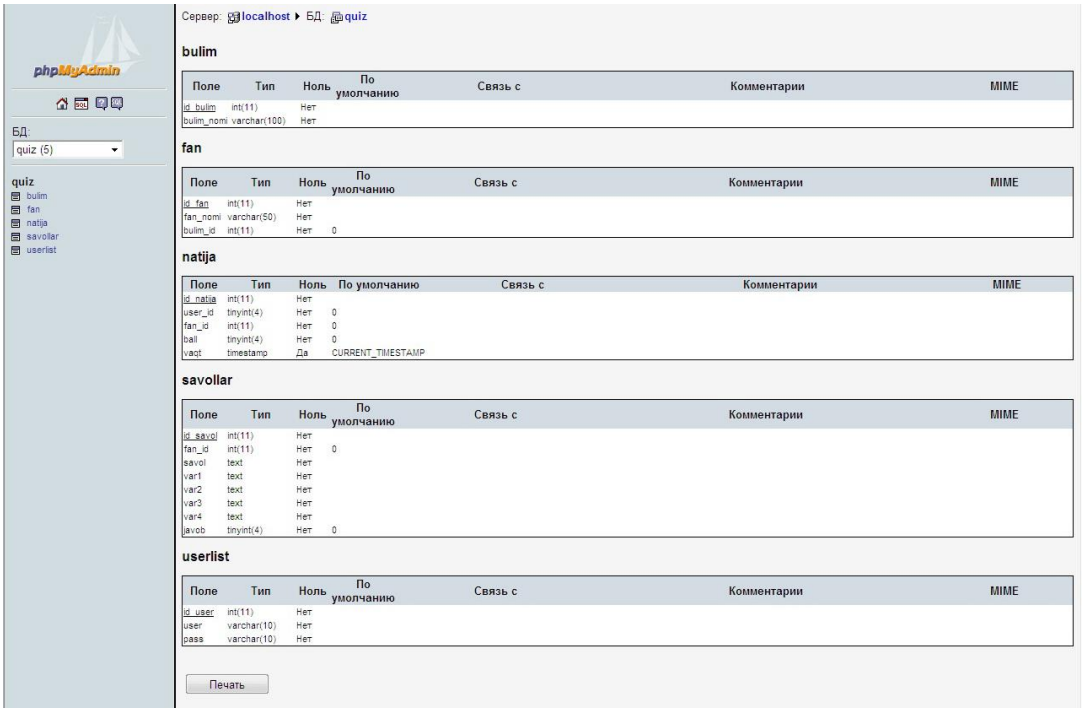


5 – расм. phpMyAdmin веб дастурининг бош саҳифаси

phpMyAdmin веб дастурини юклаш учун веб браузернинг манзил сатрига қуйидагини киритамиз:

http://localhost/denwer/Tools/phpMyAdmin/

Сўнгра quiz деб номланган маълумотлар омборини ташкил этамиз ва унда қуйидаги жадваллари ташкил этамиз:



6 – расм. Маълумотлар омбори схемаси.

3. Macromedia Dreamweaver дастури ёрдамида интерактив дастурни HTML, CSS, JavaScript, PHP тилларидан фойдаланган ҳолда яратиш

Дастурнинг асосини қуйидаги каталоглар ва файлларлар ташкил этади:
Admin (каталог)

IMG (каталог)

JS (каталог)

jquery-ui.css

bulim.php

check.php

conf.php

exit.php

fan.php

foot.php

head.php

index.php

login.php

reg.php

test.php

userbar.php

Асосий файллар листинги

bulim.php


```
<?php
require("login.php");
?>
<?php include "head.php"; ?>
<H1>Бўлимлар</H1>
<?php
include "conf.php";
$query=mysql_query("select * from bulim");
while($row=mysql_fetch_array($query))
echo "<p><a href='fan.php?bulim=".$row["id_bulim"].'">".$row["bulim_nomi"]."</a></p>";
include "userbar.php";
?>
<?php include "foot.php"; ?>
```

check.php


```
<?php
require("login.php");
?>
<?php include "head.php"; ?>
<h1>Натижа</h1>
<?php
include "conf.php";
if(isset($_POST["submit"])){
$query=mysql_query("select * from savollar where fan_id=1");
$ball=0;
while($row=mysql_fetch_array($query))
if($_POST[$row["id_savol"]] == $row["javob"])
$ball++;
echo "Балингиз: ". $ball."<br>";
include "userbar.php";
$q1=mysql_query("select id_user from userlist where user='".$_SERVER['PHP_AUTH_USER']."'");
$row=mysql_fetch_array($q1);
$q2=mysql_query("insert into natija (user_id,fan_id,ball) values (".$row["id_user"].",".$_POST["fan"].",".$ball.")");}
        ?>
```

<?php include "foot.php"; ?>

Conf.php

```
<?php
$db=mysql_connect("localhost","root","");
mysql_select_db("quiz");
mysql_query("set character_set_results=utf8;",    $db);
mysql_query("set character_set_connection=utf8;", $db);
mysql_query("set character_set_client=utf8;",     $db);
mysql_query("set character_set_database=utf8;",   $db);
?>
```

fan.php


```
<?php
require("login.php");
?>
<?php include "head.php"; ?>
<H1>Фанлар</H1>
<?php
include "conf.php";
$query=mysql_query("select * from fan where bulim_id=".$_GET["bulim"]);
while($row=mysql_fetch_array($query))
echo "<p><a href='test.php?fan=".$row["id_fan"].">".$row["fan_nomi"]."</a></p>";
```

```
include "userbar.php";
?>
<p><a href="bulim.php">Орта</a></p>
<?php include "foot.php"; ?>
```

head.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=uft-8">
<title>Untitled Document</title>
<style type="text/css">
<!--
body {
	background-image: url(IMG/fon.png);
	background-repeat:repeat-x;
}
.style1 {
	font-family: Arial, Helvetica, sans-serif;
	font-size: 24px;
	font-weight: bold;
}
.style2 {font-family: Arial, Helvetica, sans-serif; font-size: 36px; font-weight: bold; }

.content{
margin-left: 50px;
}
-->
```

```
</style>
</head>

<body>
  <table width="100%"  border="0" cellpadding="2">
    <tr>
      <td width="45%"><img src="IMG/2.gif" width="503" height="128"></td>
      <td width="55%"><span class="style1">PHP ва MYSQL ёрдамида университет талабалари<br>
билимини текширувчи интерактив дастурни яратиш</span></td>
    </tr>
  </table>
<p><div class="content">
```

foot.php

```
</div>
</p>
</body>
</html>
```

index.php

```
<?php
if(isset($_POST["submit"])){
header("Location: bulim.php");
}
?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Untitled Document</title>

<link href="jquery-ui.css" rel="stylesheet" type="text/css" />
<script type="text/javascript" src="JS/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="JS/jquery-ui.min.js"></script>

<style type="text/css">
<!--
body {
	background-image: url(IMG/fon.png);
}
.style1 {
	font-family: Arial, Helvetica, sans-serif;
	font-size: 24px;
	font-weight: bold;
}
.style2 {font-family: Arial, Helvetica, sans-serif; font-size: 36px; font-weight: bold; }
-->
</style>

<script language="JavaScript">
<!--
	$(function(){
	$("#loginform").dialog({autoOpen:false});
	$("#openme").click(function(){ $("#loginform").dialog('open');});
```

```
        });
//-->
</script>
</head>

<body>
<div align="center">
  <p><img src="IMG/2.gif" width="503" height="128">
</p>
  <p class="style2"> PHP ва MYSQL ёрдамида университет талабалари<br>
  билимини текширувчи интерактив дастурни яратиш</p>
  <p class="style1">Рахбар: Доц. Н.С. Сайидова<br>
    Битирувчи: Каюмова Гулшан</p>
  <p class="style1"> </p>
  <p class="style1"><a href="#" id="openme">тизимга кириш</a> | <a href="reg.php">регистрация</a></p>
</div>
<div id="loginform" title="Тизимга кириш">
<form action="" method="post" name="login">
Логин:<br>
<input name="user" type="text"><br>
Парол:<br>
<input name="pass" type="password"><br><br>
<input name="submit" type="submit" value="Кириш">
</form>
</div>
</body>
</html>
```

reg.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">
<title>Untitled Document</title>
<style type="text/css">
<!--
body {
	background-image: url(IMG/fon.png);
}
.style1 {
	font-family: Arial, Helvetica, sans-serif;
	font-size: 24px;
	font-weight: bold;
}
.style2 {font-family: Arial, Helvetica, sans-serif; font-size: 36px; font-weight: bold; }
.style3 {
	color: #FF0000;
	font-weight: bold;
}
-->
</style>
</head>

<body>
<div align="center">
```

```
 <p><img src="IMG/2.gif" width="503" height="128">
</p>
 <p class="style2"> PHP ва MYSQL ёрдамида университет талабалари<br>
 билимини текширувчи интерактив дастурни яратиш</p>
 <form name="form1" method="post" action="">
  <table width="319" border="0" cellpadding="3" cellspacing="2">
   <tr>
    <td colspan="2"><div align="right" class="style3">
     <div align="center">Регистрация</div>
    </div></td>
   </tr>
   <tr>
    <td width="94"><div align="left">Логин:</div></td>
    <td width="207"><div align="left">
     <input type="text" name="textfield">
    </div></td>
   </tr>
   <tr>
    <td><div align="left">Парол:</div></td>
    <td><div align="left">
     <input type="text" name="textfield">
    </div></td>
   </tr>
   <tr>
    <td><div align="left">Фамилия:</div></td>
    <td><div align="left">
     <input type="text" name="textfield">
    </div></td>
```

```
    </tr>
    <tr>
      <td><div align="left">Исм:</div></td>
      <td><div align="left">
        <input type="text" name="textfield">
      </div></td>
    </tr>
    <tr>
      <td><div align="left">Группа:</div></td>
      <td><div align="left">
        <select name="select">
          <option selected>4-07 AAT</option>
        </select>
      </div></td>
    </tr>
    <tr>
      <td> </td>
      <td> </td>
    </tr>
    <tr>
      <td> </td>
      <td><input type="submit" name="Submit" value="Юбориш"></td>
    </tr>
  </table>
 </form>
 <p class="style1"><a href="index.php">ортга</a></p>
 <p class="style1"> </p>
</div>
</body>
```

```
</html>
```

test.php

```
<?php
require("login.php");
?>
<?php include "head.php"; ?>
<h1>Тест саволлари</h1>
<form action="check.php" method="post">
<?php
include "conf.php";
$query=mysql_query("select * from savollar where fan_id=".$_GET["fan"]);
$i=1;
while($row=mysql_fetch_array($query)){
echo "<p>".$i.". ".$row["savol"]."<br>
<label><input type='radio' name='".$row["id_savol"]."' value='1' checked>".$row["var1"]."</label><br>
<label><input type='radio' name='".$row["id_savol"]."' value='2'>".$row["var2"]."</label><br>
<label><input type='radio' name='".$row["id_savol"]."' value='3'>".$row["var3"]."</label><br>
<label><input type='radio' name='".$row["id_savol"]."' value='4'>".$row["var4"]."</label></p>";
$i++;
}
?>
<input type="hidden" name="fan" value="<?php echo $_GET["fan"]; ?>">
<input type="submit" value="Якунлаш" name="submit">
```

```
</form>
<?php
include "userbar.php";
include "foot.php";
?>
```

4. Дастурни интернет тармоғига жойлаштириш

Дастурни интернет тармоғига жойлаштириш учун Total Commander, CuteFTP, FileZilla FTP Client, Windows Standart FTP Client дастурларидан фойдаланишингиз мумкин. Веб хостинг компанияси томонидан файлларни серверга юклаш учун веб интерфейс тақдим этилиши мумкин.

FTP протоколидан фойдаланиш учун юқорида айтиб ўтилган дастурларни компьютерга ўрнатиб, ўзингизга тегишли сервер манзили, логин, парол каби маълумотларни дастурга киритиб, FTP серверга уланиш мумкин.

Х У Л О С А

II боб PHP Web дастурлаш тили ва MySQL маълумотлар омборини бошқариш тизими. Веб дастурни созлаш ва ундан фойдаланиш деб номланиб қуйидаги жами бўлиб 3 та бўлимлардан иборат. Улар 1 - бўлим PHP Web дастурлаш тили, 2 – бўлим MySQL маълумотлар омборини бошқариш тизими, 3 – бўлим web дастурни созлаш ва ундан фойдаланиш деб номланган бўлиб берилган мавзунинг амалий қисмини ўзида жамлаган. Бундаги келтирилган барча маълумотлар замонавий дастурлаш тилларига тегишли бўлиб, PHP Web дастурлаш тили ва MySQL маълумотлар омборини бошқариш тизими. Веб дастурни созлаш ва ундан фойдаланиш, PHP Web дастурлаш тили, MySQL маълумотлар омборини бошқариш тизими, дастурни созлаш ва ундан фойдаланиш кетма-кетлиги батафсил келтирилган.

Ҳар бир бўлим номидан келиб чиқадиган бўлсак, қўйилган мавзусига мос материаллар, 2 та расм билан берилган. Хулоса қилиб айтадиган бўлсак,

масофавий таълим порталини яратишда Web дастурлаш технологиялари орасидан PHP дастурлаш тилини танлаб олдик. Бу дастурлаш тили билан қисқача танишдик. PHP дастурлаш тили маълумотларни сақлашда MYSQL маълумотлар омбори билан яхши ишлаганлиги учун маълумотлар базасини ҳам MYSQL да яратишга қарор қилдик. PHP компиляторини ишлатиш учун Apache Web сервердан фойдаланилади.

Х О Т И М А

Ҳозирги кунда интернет тармоғида мавжуд бўлган асосий веб сайтларни яратишда PHP веб дастурлаш тили ва MySQL маълумотлар омборини бошқариш тизимидан кенг фойдаланиб келинмоқда. Сабаби шундаки, PHP веб дастурлаш тили йиллар сайин янги имкониятлар ва қулайликларни веб дастурчиларга тақдим этмоқда. PHP веб дастурлаш тилида ёзилган веб иловалар тез ишлайди ва хавфсизлик жиҳатдан ижобий томонлари мавжуд.

Мен яратган PHP ва MySQL ёрдамида университет талабалари билимини текширувчи интерактив дастурда талабалар билимини олий таълим муассасида ўқитиладиган турли хил фанлар бўйича тестларни топшириш имконияти мавжуд. Талаба тестни топшириб бўлганидан сўнг тестлаш натижалари маълумотлар омборига киритилади ва ушбу натижалар дастур администратори томонидан назорат қилиниши мумкин. Дастурдан фақатгина рўйхатдан ўтган талабалар фойдаланиш ҳуқуқига эга. Рўйхатдан ўтган талабаларнинг ҳар бири ўзининг шахсий логин ва паролига эга бўлади. Дастур асосан 2 қисмга эга бўлади.

1. Фойдаланувчи қисми.
2. Администрарот қисми.

Администратор қисмида фан бўлимлари, фанларни, тест саволларини ва тест натижаларини бошқариш имконияти мавжуд.

Ушбу дастурдан барча ёшдаги инсонлар фойдаланишлари мумкин.

ФОЙДАЛАНИЛГАН АДАБИЁТЛАР РЎЙХАТИ

1. Лаура Томсон и ЛюкВеллинг. Разработка Web –приложений на PHP и MySQL. DiaSoft. Санкт-Петербург – 2003.

2. Н. Н. Куссуль, А.Ю. Шелестов. Самоучитель использование PHP. Диалектика Москва-Санкт-Петербург-Киев-2005.

3. Л.Аргерих, Д.Коггсхо, КЭгервари, М.Гейслер. Профессиональное PHP программирование. Второе издание. Санкт-Петербург-2003.

4. Д.М. Расулев, А.К. Машарипов, К.Т. Жумабаев, Э.У. Ибрагимов, М.А. Мусаева. Электронное учебное пособие по курсу «Проектирование Web - сайта». Ташкент- 2005 г.

5. А.К. Машарипов, Т.А. Закирова, А.А. Абидов, М.А. Мусаева. Электронное учебное пособие по курсу «Проектирование Web - сайта». Ташкент- 2006 г.

6. Ш.Т. Эрматов, Н.Х. Шоахмедова. HTML тилида Web саҳифани яратиш. − Т., 2005 й.

7. Т.А. Зокирова, А.К. Машарипов, Э.У. Иброҳимов, М.А. Мусаева. Web дастурлаш. Ўқув қўлланма. – Т.: ТДИУ, 2006. – 175 б.