Қазақстан Республикасы Ғылым және жоғары білім министрлігі

Абай атындағы Қазақ ұлттық педагогикалық университеті

Сагимбаева А.Е., Жаксылыков А.Е.

GENAI ТЕХНОЛОГИЯСЫН ҚОЛДАНУ НЕГІЗІНДЕ БАҒДАРЛАМАЛАУ БОЙЫНША БІЛІМДІ БАҚЫЛАУДЫҢ БЕЙІМДЕЛГЕН КРИТЕРИЙЛЕРІ ТУРАЛЫ ҚҰЖАТТАМА

Бұл зерттеуді Қазақстан Республикасы Ғылым және жоғары білім министрлігі Ғылым комитеті қаржыландырды (Грант № AP23490592).

Алматы, 2025

МАЗМҰНЫ

КІРІСПЕ

Қазіргі цифрлық қоғамда заманауи білім беру жүйесі технологиялық өзгерістер мен жаңа әдістемелік көзқарастардың ықпалында қарқынды дамып келеді. Әсіресе, жасанды интеллект (AI) және генеративті жасанды интеллект (GenAI) технологиялары білім сапасын арттырудың жаңа мүмкіндіктерін ашты. Бұл технологиялар тек оқыту процесінде ғана емес, студенттердің білімін бақылау мен бағалау кезеңдерінде де маңызды рөл атқара бастады.

ЖОО-да оқытылып жатқан программалау пәндері заманауи қоғам мүшелерінің цифрлық сауаттылығын қалыптастырудың негізгі өзегі болып табылады. Бұл пәндер логикалық ойлау, алгоритм құру, шешім қабылдау және шығармашылық қабілеттерді дамытуға бағытталған. Алайда, дәстүрлі бақылау жүйелері (мысалы, тест немесе жазбаша тапсырмалар) студенттердің бұл қабілеттерін толық өлшей алмайды.

Міне, осы тұста GenAI технологияларының көмегімен бейімделген бақылау жүйесін құру өзекті мәселеге айналып отыр. Бұл тәсіл әр студенттің жеке оқу траекториясын ескеріп, тапсырманың күрделілігін, мазмұнын және бағалау критерийлерін автоматты түрде реттеуге мүмкіндік береді.

Мысалы, егер студент программалау тапсырмасын дұрыс шешсе, GenAI жүйесі оған келесі, біршама күрделі тапсырманы ұсына алады. Ал қателескен жағдайда, жүйе оған жеңілдетілген, бағыттаушы тапсырма немесе қосымша түсіндірме береді. Осылайша, бақылау үдерісі бейімделген және интерактивті сипат алады.

Сонымен қатар, GenAI технологиялары оқытушының уақытын үнемдеп, бағалаудағы субъективтілікті азайтады. Студенттің жұмысын нақты критерийлер бойынша автоматты талдау (мысалы, код дұрыстығы, тиімділігі, логикалық байланысы) студенттердің білімін объективті бағалауға мүмкіндік береді.

Қазіргі таңда программалаудан студенттердің білімін бақылауда қолданылып жүрген кейбір GenAI құралдары мыналар:

- ChatGPT, Gemini, Copilot – кодты талдау және түсіндіру;
- Replit AI, CodeWhisperer – программаны автоматты толықтыру;
- GitHub Copilot Labs – оқыту және тест құрастыру;
- AI feedback жүйелері – студентке жеке түсіндірме мен кеңес беру.

Мұндай құралдар оқытушының рөлін жоққа шығармайды, керісінше оның әдістемелік жұмысын күшейтіп, студенттерге бағытталған оқу ортасын қалыптастырады.

Осы әдістемелік нұсқаулықтың негізгі мақсаты оқытушыларға GenAI технологиясын қолданып, программалаудан білімді бейімделген бақылау жүйесін құруға арналған нақты нұсқаулық пен критерийлерді ұсыну болып табылады.

Құралдың мазмұны оқытушыларға GenAI құралдарының мүмкіндіктерін танып білуге; программалау пәні бойынша бейімделген бағалау критерийлерін құруға; нәтижелерді автоматты талдау мен визуализациялауға; академиялық адалдық пен этиканы сақтауға көмектеседі.

1. GenAI технологиясын программалау пәнінде қолдану мүмкіндіктері

Жасанды интеллекттің жаңа буыны – Generative AI (GenAI) – білім беру саласына үлкен өзгеріс әкелді. Бұл технология адамның тапсырмасы (prompt) бойынша мәтін, код, сурет, есеп, сұрақ немесе шешім ұсына алады. GenAI жүйелерінің ерекшелігі – олар дайын жауап бермей, оқушының сұранысына қарай бейімделіп, бағыттап, түсіндіру мүмкіндігін береді.

GenAI технологиялары білім беру үдерісін жеңілдетіп қана қоймай, оны **дербестендірілген**, бейімделген және интерактивті етеді. Программалау пәні мысалында бұл технологиялар келесі бағыттарда тиімді қолданылады (1-кесте).

№	Қолдану бағыты	GenAI мүмкіндігі	Нәтиже
1	Код жазу	ChatGPT, Copilot көмегімен код құрылымын ұсыну	Кодтың сапасы артады
2	Қате талдау	GenAI кодтағы синтаксистік және логикалық қателерді анықтайды	Қателер азаяды
3	Тапсырмаларды бейімдеу	Студент деңгейіне қарай күрделілік таңдау	Жеке траектория қалыптасады
4	Кері байланыс	AI студентке не дұрыс, не бұрыс екенін түсіндіреді	Өз бетінше үйрену қалыптасады
5	Тест құру	GenAI автоматты түрде тест сұрақтарын жасайды	Уақыт үнемделеді

Программалауды оқыту барысында GenAI қолдануды нақты мысалдармен қарастырайық.

- ChatGPT немесе Gemini — студентке программаның идеясын түсіндіру, синтаксис бойынша көмек бере алады.
- GitHub Copilot — Visual Studio Code ішінде кодтың жалғасын автоматты түрде ұсынады.
- Replit AI — студенттің қате кодын талдап, қадамдық нұсқаумен түзету жолдарын көрсетеді.
- AI Quiz Generator — тақырып бойынша жылдам тест пен жаттығу құрастырады.

Мысалы, студентке “Пайдаланушы енгізген санның жай сан екенін тексеретін код жазыңыз” деп тапсырма берілсе, GenAI мұнда бірнеше көмек түрін ұсына алады:

- алгоритмдік логиканы түсіндіреді;
- Python тілінде мысал код ұсынады;
- қате болса, қай жерден кеткенін көрсетеді;
- әрі қарай күрделенген нұсқасын ұсынады (мысалы, жай сандарды диапазон бойынша табу).

Программалаудан студенттердің білімін бағалау барысында GenAI технологиясын қолданудың ең үлкен артықшылығы уақыт үнемдеу және бағалау сапасын арттыру болып табылады.

Мысалы:

- Әр студентке жеке тапсырма беру мүмкіндігі артады;
- Оқытушы кодтың дұрыстығын тез тексеріп, кері байланыс бере алады;
- Топтағы үлгерім статистикасын AI өзі құрастырады;
- Бейімделген бақылау үшін критерийлерді жүйе автоматты түрде қолдана алады.

Дегенмен, басқа технологиялар сияқты, GenAI да шексіз емес. Оны тиімді пайдалану үшін келесі шектеулерді ескерген жөн:

- AI шешімі әрқашан 100% дұрыс бола бермейді, сондықтан оқытушы тексеруі қажет;
- Дайын кодты көшіру академиялық адалдықты бұзады, сондықтан студент түсініп қолдануы тиіс;
- Prompt жазу дағдылары болуы қажет, себебі ол AI-мен жұмыс істеудің негізгі кілті болып табылады;
- Тілдік шектеу — жүйелер көбіне ағылшын тілін жақсырақ түсінеді, сондықтан аралас тілді қолдану тиімді.

Қорыта келе, GenAI оқытушының орнын басатын құрал емес, ол оқу үдерісін жандандыратын интеллектуалды көмекші болып табылады. GenAI программалауды оқыту барысында дұрыс қолдану арқылы:

- студенттің жеке даму қарқыны сақталады;
- бағалау әділ әрі бейімделген болады;
- оқыту сапасы артады.

2. Бейімделген (adaptive) бақылаудың мәні

Білім алушыны әділ және тиімді бағалау оқыту сапасының басты көрсеткіші болып табылады. Дәстүрлі бақылау жүйесі барлық студенттерге бірдей тапсырма ұсынса, бейімделген бақылау (adaptive assessment) әр студенттің жеке деңгейіне, оқу қарқынына және алдыңғы нәтижелеріне сүйеніп, тапсырмаларды бейімдеп береді.

Бейімделген бақылаудың мақсаты — студенттің шынайы қабілетін анықтау.

Мысалы, егер студент тақырыпты жақсы меңгерсе, жүйе күрделірек тапсырма ұсынады; егер қателессе — жеңілдетілген немесе түсіндірулі тапсырма береді.

Бұл тәсіл:

- студентті мотивациялауға;
- оқу үдерісін жекелешелендіруге;
- бағалауды объективтілігін қамтамасыз етуге көмектеседі.

Енді GenAI көмегімен бейімделген бақылаудың қалай жүзеге асатынына талдау жасап көрейік.

GenAI жүйесі студенттің әрекеттерін (код жазу уақыты, қатені түзету саны, тапсырма нәтижесі) талдай отырып, оқыту траекториясын автоматты реттейді.

Мысалы:

- Студент алғашқы тапсырманы дұрыс шешті → келесі күрделі деңгей ұсынылады.
- Қате жіберді → жүйе түсіндіру мен қосымша нұсқаулық береді.
- Бірнеше әрекеттен кейін де нәтиже төмен → жүйе тапсырманы қарапайым мысалмен түсіндіреді.

Нәтижесінде, әр студент өз деңгейінде жетістікке жете алады.

Қорыта келе, бейімделген бағалаудың артықшылықтары 2-кестеде сипатталды.

2-кесте. Бейімделген бақылаудың артықшылықтары

Артықшылықтары	**Түсініктеме**
Дербестендіру	Әр студенттің деңгейі мен қарқыны ескеріледі
Объективтілік	Бағалау нақты критерийлер мен деректерге негізделеді
Мотивация	Студент өз жетістігін көреді, үнемі прогресс байқалады
Үнемділік	Оқытушының уақыты қысқарады, бағалау автоматтандырылады
Деректерге негізделу	Нәтижелерді талдау арқылы оқу сапасы артады

Жоғарыдағы айтылғандарды негізге ала отырып, программалаудан студенттердің білімін бейімделген бақылау үлгісі 3-кестеде келтірілді.

3-кесте. Программалаудан студенттердің білімін бейімделген бақылау үлгісі

Студент деңгейі	Тапсырма сипаттамасы	Кері байланыс түрі
Бастапқы	Қарапайым есептер (цикл, шарт)	Қате болса, қадамдық түсіндірме
Орта	Функция, рекурсия, тізімдер	Алгоритм тиімділігі бойынша кеңес
Жоғары	Жоба құру, модульдік код	Код құрылымы мен логика талданады

GenAI көмегімен бейімделген бақылау оқытушыларға мынадай мүмкіндіктер ұсынады:

- Бағалау әділ әрі бірізді болады;
- Әр студенттің прогресін нақты деректермен дәлелдеуге болады;
- Бақылау интерактивті сипат алады (GenAI сұрақ қойып, кері байланыс береді);
- Нәтижелерді автоматты түрде визуализациялау мүмкіндігі пайда болады.

Қорыта келе, GenAI көмегімен бейімделген бақылау әдісі оқытушы мен студент арасындағы өзара әрекетті деректерге негізделген интеллектуалды үдеріске айналдырады.

3. Бағалау критерийлері мен бейімделген рубрика

Программалау курсынан бағалау жүйесі студенттің кодты жаттап алу қабілетін емес, логикалық ойлауын, алгоритм құру білігін және GenAI құралдарын дұрыс қолдануын бағалауы тиіс.

Сондықтан бағалау критерийлері дәстүрлі “дұрыс/бұрыс” тәсілінен кеңірек болуы қажет.

GenAI технологиясы енгізілген жағдайда бағалау келесі қағидаттарға сүйенуі керек:

- Бейімделгіштік – тапсырмалар студенттің деңгейіне қарай автоматты өзгереді;
- Дәлдік – бағалау нақты критерийлерге сүйенеді, субъективтілік азаяды;
- Жеке кері байланыс – әр студент өз қатесін түсінеді және түзету жолын алады;

- Этика – GenAI-дің көмегі дұрыс пайдаланылған ба, әлде көшірме ме — бұл да ескеріледі.

Программалаудан бейімделген бағалауда мынадай негізгі критерийлер ұсынылады (4-кесте):

4-кесте. Программалаудан бағалау критерийлері

Бағалау критерийі	**Сипаттамасы**
Кодтың дұрыстығы	Нәтиже дұрыс шыға ма, алгоритм жұмыс істей ме
Алгоритмнің тиімділігі	Код қысқа әрі ресурсты үнемді пайдалана ма
Құрылым және түсініктілік	Кодтың оқылуы, айнымалы атауларының мағынасы
Алгоритмдік ойлау	Студент шешімге қалай келгенін дәлелдей ала ма
GenAI құралын қолдану сапасы	Студент AI көмегін дұрыс бағытта пайдалана ма
Шығармашылық және талдау қабілеті	Студент өз кодын жетілдіре ме, жаңа тәсіл ұсына ма

Бағалау критерийлері бейімделген, әділ және дамуға бағытталған түрде жасалу керек. Бұл тек техникалық дұрыстық пен нәтиже сапасын ғана емес, сонымен қатар студенттің ойлау қабілетін, мәселені шешу тәсілін және деңгейіне сай құзыреттілігін өлшейді. Бағалау критерийі екі негізгі бағытқа сүйенеді: пәннің мазмұнына және студенттің деңгейі мен тапсырманың деңгейіне негізделген бағалау критерийі. Бағалау тек нәтиженің дұрыстығын емес, сонымен бірге *ойлау барысын және үйрену үдерісін* де қамтиды. Программалаудан білім мен дағдыларды бағалауға арналған бейімделген чек-парақ (Кесте 1).

4.1-кесте. Программалаудан білім мен дағдыларды бағалауға арналған бейімделген чек-парақ

Бағалау критерийі	**Бастапқы деңгей (Beginner)**	**Орта деңгей (Intermediate)**	**Жоғары деңгей (Advanced)**
1.Негізгі ұғымдарды түсіну	Негізгі синтаксис пен деректер түрлерін таниды, қарапайым кодты жаза алады.	Тілдің құрылымын жақсы біледі, функциялар мен модульдерді қолдана алады.	Тілдің мүмкіндіктерін терең меңгерген, күрделі логикалық құрылымдар құра алады.

2. Алгоритмдік ойлау	Қарапайым алгоритмдерді (мысалы, цикл, шарт) қолдана алады.	Алгоритмді түрлендіре алады және тиімділігін талдайды.	Өз алгоритмін жобалайды, деректер құрылымдарын оңтайлы пайдаланады.
3. Код сапасы	Код жұмыс істейді, бірақ құрылымы ретсіз болуы мүмкін.	Код модульдік және түсініктемелермен берілген.	Код таза, тиімді, кәсіби стандарттарға сай.
4. Қате табу және жөндеу дағдысы (Debugging)	Қателерді мұғалім көмегімен табады.	Қателердің себебін талдай алады, шешім ұсынады.	Қателерді өз бетімен тез анықтап, күрделі логиканы түзете алады.
5. Мәселені шешу тәсілі	Дайын үлгілерге сүйенеді, шығармашылық аз.	Өз бетінше шешім табуға тырысады, бірақ кейде толық емес.	Мәселені бірнеше жолмен шешіп, ең тиімдісін таңдайды.
6. Жоба құрастыру (егер бар болса)	Қарапайым қосымшаны мұғалім көмегімен орындайды.	Функционалды жоба жасайды, құрылымын түсіндіре алады.	Күрделі, интеграцияланған жүйе жасап, өз идеясын жүзеге асырады.
7. Құжаттау және таныстыру	Минималды түсіндірме, кодқа түсініктеме сирек.	Құжатты негізгі бөлімдерімен толтырады.	Толық, кәсіби баяндау, техникалық және визуалды безендіру бар.
8. Шығармашылық пен зерттеу элементі	Тек нұсқаулыққа сүйенеді.	Өз идеясын ішінара қолданады.	Толықтай жаңа тәсіл ұсынады немесе зерттеу элементін қосады.

Осы чек-парақтың негізінде келесі программалау курстары бойынша бағалау критерийлері жасалды (Қосымшалар).

Студенттердің программалаудан білімдерін бағалау критерийлеріне сәйкес бағалау рубрикасын былайша жасауға болады (5-кесте)

5-кесте Бағалау рубрикасының үлгісі

Бағалау өлшемі	1-деңгей (төмен)	2-деңгей (орташа)	3-деңгей (жоғары)
Дұрыстық	Код жұмыс істемейді немесе қате нәтиже береді	Барлық тесттен өтеді	Қосымша тесттерді өзі құрастырады
Тиімділік	Код артық әрекеттермен жазылған	Кейбір тиімді әдістер қолданылған. Алгоритм оптималды	Уақыт пен жадыны үнемдеп, үздік әдіс қолданған
Құрылым	Код түсініксіз, ретсіз	Айнымалы атаулары бірізді емес. Құрылым сақталған	Модульдік, толық түсінікті код
GenAI қолдану	Дайын код көшірілген	Prompt дұрыс қолданылмаған. GenAI нәтижесін өңдеп жетілдірген	GenAI көмегімен кодты жетілдіріп, рефлексия жасаған
Логикалық ойлау	Қадамдар байланыссыз	Логика бар. Алгоритм толық құрастырылған.	Бірнеше тәсілді салыстырып, тиімдісін таңдаған
Шығармашылық	Стандартты шешім	Кейбір жаңа идея және өздік шешім бар	Ерекше, баламалы шешім ұсынған

Енді осы бағалау рубрикасын қолдануды мысалмен көрсетейік:

Мысалы, студент “Массив элементтерінің орташа мәнін табу” есебін шешу кодын жазды. GenAI жүйесі мен оқытушы былайша бағалай алады (6-кесте)

6-кесте. GenAI жүйесі мен оқытушының бағалауы

Критерий	Нәтиже	Бағалау деңгейі
Дұрыстық	Барлық тесттен өтті	2-деңгей
Тиімділік	for циклмен дұрыс жазылған	2-деңгей
Құрылым	Код түсінікті, айнымалы атаулары нақты	3-деңгей
GenAI қолдану	ChatGPT көмегімен түсіндірме алған	2-деңгей
Шығармашылық	Қосымша функция қосты	3-деңгей

Қорытынды балл: орташа 2,3 → “Орташа” деңгей

GenAI-да бейімделген бағалау үдерісі былайша жүзеге асырылады: GenAI жүйесі студенттің алдыңғы нәтижелерін есте сақтап, бағалау кезінде соған сүйенеді.

Мысалы:

- Егер студент соңғы 3 тапсырмадан 80% нәтиже алса → күрделірек тапсырма беріледі;
- Егер 50%-дан төмен болса → жүйе жеңілдетілген мысалдармен бекітеді;
- Егер 100%-ға жетсе → GenAI шығармашылық тапсырма ұсынады.

Сондықтан да, бағалау кезінде студенттің тек соңғы нәтижесіне емес, даму динамикасына назар аудару керек; GenAI жүйесінің берген бағасын оқытушы міндетті түрде тексеріп, соңғы шешімді өзі қабылдауы керек. Рубриканы әр тақырыпқа бейімдеп өзгертуге болады (мысалы, “Циклдар”, “Функциялар”, “Жобалық жұмыс” т.б.).

4. GenAI негізінде программалаудан тапсырма үлгілері

Сонымен, бейімделген тапсырма әр студенттің деңгейіне, алдыңғы нәтижелеріне және оқу қарқынына қарай өзгеретін тапсырма. GenAI технологиясы мұны автоматты түрде іске асыра алады. GenAI студенттің алдыңғы кодына қарап, келесі тапсырманың күрделілігін өзі реттейді:

- егер студент қате көп жіберсе, жүйе *қарапайым мысал* береді;
- егер дұрыс орындаса, жүйе *қосымша шарттармен күрделендірілген тапсырма* ұсынады.

GenAI негізіндегі тапсырмаларды үш негізгі топқа бөлуге болады (7-кесте):

7-кесте. GenAI негізіндегі тапсырмалардың түрлері

Тапсырма түрі	Мақсаты	GenAI қолдану тәсілі
Бейімделген тест	Білім деңгейін анықтау	GenAI сұрақ күрделілігін өзгертеді
Код жазу тапсырмасы	Алгоритмдік ойлау мен логиканы дамыту	AI кодқа түзету мен кері байланыс береді
Жоба немесе кейс	Шығармашылық және аналитикалық қабілетті дамыту	AI идея, құрылым және талдау ұсынады

1-тапсырма (бастапқы деңгей). Пайдаланушы енгізген екі санның қосындысын есептейтін программа құрыңыз.

GenAI қолдауы:

- Жүйе студенттің жазған кодын тексеріп, input(), int(), print() функцияларын қолдануын бақылайды.

- Егер нәтиже дұрыс болмаса, "Айнымалы түрін түрлендіру қажет болуы мүмкін" деген кеңес береді.

2-тапсырма (орта деңгей). Берілген тізімдегі ең үлкен және ең кіші элементті табатын код жазыңыз.

GenAI кеңесі:

- Егер студент for циклін қолданса, GenAI max() және min() функцияларын ұсынады.

- Егер кодта логикалық қате болса, жүйе нақты қатені көрсетіп, мысалмен түсіндіреді.

3-тапсырма (жоғарғы деңгей). Берілген мәтіннен әріптер жиілігін санайтын және нәтижені сөздік (dictionary) түрінде шығаратын код жазыңыз.

GenAI көмекші әрекеті:

- Студенттің шешімін бағалап, тиімді нұсқасын ұсынады;

- Егер код тым күрделі болса, оңтайландыру мысалын береді;

- Қосымша ұсыныс: "Осы кодты файлдан оқу арқылы орындап көріңіз" деген кеңес береді.

Енді GenAI көмегімен тапсырма бейімдеу сценарийін қарастырайық (8-кесте)

8-кесте. GenAI көмегімен тапсырма бейімдеу сценарийі

Кезең	GenAI әрекеті	Нәтиже
1	Студентке бастапқы тапсырма береді	Деңгей анықталады
2	Нәтижені талдайды	Тапсырма күрделенеді немесе жеңілдейді
3	Кері байланыс ұсынады	Қателер түзетіледі
4	Қорытынды нәтижені шығарады	Бағалау жүргізіледі

GenAI-ге арналған дұрыс prompt (тапсырма мәтінін) құру маңызды.

Мысалы: "Студентке арналған Python тілінен тапсырма жаса.

Деңгей: орта. Тақырып: циклдар.

Мақсат: қайталау операторын қолдану.

Әр тапсырманың соңында студентке “не үйрендің?” немесе “нені жақсартуға болады?” деген рефлексия сұрақтарын енгізу.

GenAI студентке тек дайын шешім бермей, ойлануға итермелейтін сұрақтар қойсын.

GenAI негізінде құрылған тапсырмалар студентті пассивті орындаушы емес, белсенді зерттеуші деңгейіне көтереді. Ол қателіктерді үйрену мүмкіндігі ретінде қабылдап, логикалық ойлауын дамытады.

GenAI негізінде жасалған тапсырмаларға мысалдар келтірейік.

Мысалы, “Рекурсия және функциялар” тақырыбы. Бұл тақырып студенттің логикалық ойлауын, алгоритмдік түсінігін және функциялар арасындағы байланысты түсінуін дамытады.

Оқыту мақсаты:

- Рекурсия ұғымын түсіну;
- Рекурсивті функция құра білу;
- Қарапайым және күрделі есептерге рекурсивті шешім қолдану;
- GenAI көмегімен кодты жетілдіру және бағалау.

1-тапсырма (бастапқы деңгей). Берілген санның факториалын есептейтін функция жазыңыз.

Мысалы:

```
def factorial(n):
    if n == 1:
        return 1
    else:
        return n * factorial(n - 1)

num = int(input("Санды енгізіңіз: "))
print(f"{num}! = {factorial(num)}")
```

GenAI бағалау логикасы:

- Кодта синтаксистік қате жоқ па, тексереді;
- Рекурсия дұрыс жұмыс істей ме, бақылайды;
- Нәтиже мысалмен салыстырылады (мысалы, 5! = 120);
- Нәтиже дұрыс болса, “жақсы деңгей”, қате болса “бастапқы деңгей” деп бағалайды.

2-тапсырма (орта деңгей). Рекурсия қолданбай факториал есептейтін (итерациялық тәсіл) код жазыңыз. Екі тәсілді салыстырып, қайсысы тиімді екенін түсіндіріңіз.

Мысалы:

```
def factorial_iter(n):
    result = 1
```

```
    for i in range(1, n+1):
        result *= i
    return result
```

GenAI әрекеті:

- Екі кодтың орындалу уақытын салыстырады;
- Студентке “итерациялық тәсіл жадыны аз пайдаланады” деген кеңес береді;
- Күрделілігін талдап, O(n) және O(n!) айырмашылығын түсіндіреді.

3-тапсырма (жоғарғы деңгей). Рекурсия мен итерация тәсілдерін бір бағдарламада біріктіріп,пайдаланушыға қай тәсілді таңдайтынын ұсынатын жүйе жасаңыз.

Мысалы:

```
def factorial(n, method="iter"):
    if method == "rec":
        if n == 1:
            return 1
        return n * factorial(n - 1, "rec")
    else:
        result = 1
        for i in range(1, n + 1):
            result *= i
        return result

choice = input("Әдісті таңдаңыз (rec/iter): ")
num = int(input("Санды енгізіңіз: "))
print(f"Нәтиже: {factorial(num, choice)}")
```

GenAI кері байланысы:

- код құрылымының дұрыстығын бағалайды;
- Функция параметрлерін қолдануды оң бағалайды;
- Қосымша ұсыныс береді: “кодқа енгізу қатесін өңдеу (try/except) қосыңыз.”

Соңында бейімделген бағалау нәтижесін ұсынады (9-кесте)

9-кесте. Бейімделген бағалау нәтижесі

Критерий	**Нәтиже**	**Баға деңгейі**
Код дұрыстығы	Барлық нұсқа жұмыс істейді	4
Алгоритм тиімділігі	Итерация жадыны аз пайдаланады	4
Құрылым	Түсінікті, модульдік код	4
AI қолдану	Prompt арқылы кеңес алған, жетілдірген	3
Жалпы баға	Орташа 3,75 – жоғары деңгей	✓

ҚОРЫТЫНДЫ

Қазіргі білім беру жүйесінде GenAI технологияларын қолдану оқыту мен бағалау сапасын арттырудың жаңа кезеңі болып табылады.

GenAI технологиялары студенттің жеке оқу траекториясын қалыптастыруға; бейімделген бақылау жүйесін автоматтандыруға; объективті бағалау мен уақыт үнемдеуге және академиялық әділдікті сақтауға мүмкіндік береді.

Программалау курсында GenAI көмегімен бақылау жүргізу арқылы студент тек код жазып қана қоймайды, сонымен қатар өз шешімін талдауға, тиімділігін бағалауға және қателерін түзетуге дағдыланады.

Программалаудан білімді бағалауда GenAI ды қолданудың артықшылықтары мынада: тапсырмалар әр студенттің деңгейі мен қабілетіне қарай реттеліп беріледі; Код сапасы, логикасы мен тиімділігі AI арқылы тексеріледі; интерактивті кері байланыс жасау арқылы студент нақты, жекеленген түсіндірме алады.

Бейімделген бағалау жүйесін енгізу бойынша мынадай ұсыныстар беруге болады: бірінші кезеңде қрапайымнан бастап, тек бір модуль немесе тақырып бойынша GenAI қолданып көру керек (мысалы, “Циклдар” тақырыбы); Excel немесе Google Sheets арқылы студент нәтижелерін тіркеп отыру керек AI оларды талдауға пайдалана алады; Әр тақырыпқа жеке критерийлер жасаған дұрыс (мысалы, логика, тиімділік, стиль). GenAI-ден әр студентке жеке түсініктеме беруді қамтамасыз ету керек *(Prompt мысалы: “Осы студенттің коды бойынша қысқаша кері байланыс жаз.”)*.

Болашақта GenAI интеграцияланған LMS (Learning Management System) жүйелерін дамыту; Қазақ тіліндегі AI модельдерін жетілдіру (отандық контентке бейімдеу); AI негізінде оқыту аналитикасын (Learning Analytics) дамыту; Оқытушыларға арналған “AI этикасы және педагогикалық бейімделу” курстарын енгізу.

GenAI технологиялары білім беру саласының болашағын айқындайтын маңызды құрал болып табылады. Оқытушы бұл технологияны дұрыс, мақсатты және этикалық тұрғыдан қолдана білсе, ол оқыту процесін цифрлық деңгейге көтеріп, студенттің логикалық ойлауын, шығармашылығын және дербестігін дамытуға үлкен мүмкіндік ашады.

Қосымша 1

1. *«C++ программалау тілінің негіздері» пәнінен бағалау критерийлері*

Бағалау критерийі	**Бастапқы деңгей (Beginner)**	**Орта деңгей (Intermediate)**	**Жоғары деңгей (Advanced)**
1.Негізгі ұғымдарды түсіну	Айнымалыларды, деректер типтерін (int, double, char) және негізгі операторларды (+, -, *, /, if, for) біледі. программа құрылымын түсінеді, бірақ функцияларды қолдануда қиналады	Функциялар, массивтер (array), тізбектер (string) және енгізу/шығару (cin, cout) операцияларын дұрыс қолданады. Қарапайым құрылымдарды (struct) пайдалана алады.	C++ тілінің объектіге бағытталған мүмкіндіктерін (класс, мұрагерлік, инкапсуляция, полиморфизм) біледі және орынды қолданады. Кітапханаларды (<vector>, <algorithm>, <fstream>) еркін пайдалана алады.
2. Алгоритмдік ойлау	Қарапайым тізбектік алгоритмдерді жаза алады, бірақ шарттар мен циклдерді қолдануда жиі қате жібереді.	Циклдер мен шарт операторларын дұрыс пайдаланады, функциялар арқылы есептерді шешеді.	Күрделі есептерге тиімді алгоритмдер құрып, уақыт пен жад күрделілігін (O(n), O(log n)) ескереді.
3. Код сапасы	Кодты қолмен жазады, бірақ жиі қателіктер жібереді (нүктелі үтір, жақша, деректер типі қателері). Айнымалы атаулары түсініксіз болуы мүмкін.	Кодты оқуға ыңғайлы етіп шегініс (отступ) пен форматтау қолданады. Айнымалы мен функция атаулары мағыналы. Код құрылымында қайталануды азайтуға тырысады.	Таза, модульдік және стандарттарға сай код жазады. Функцияларды тиімді пайдаланады, класстар мен әдістерге нақты атау береді (мысалы, class Calculator, void printResult() сияқты). Код қайта қолдануға және кеңейтуге ыңғайлы.
4. Қате табу және жөндеу дағдысы (Debugging)	Бағдарламаны іске қосқанда пайда болатын синтаксистік қателерді (нүктелі үтір, жақша, айнымалы атауы т.б.) таба алады. Қате туралы компилятор хабарламаларын оқуға тырысады, бірақ	Қате орындарын анықтап, түзете алады. Қарапайым логикалық қателерді (қате шарт, цикл шексіздігі, айнымалы мәнін дұрыс бермеу) өздігінен таба алады. Программаны бірнеше рет сынап көру арқылы нәтижені жақсартады.	Қателерді тез және жүйелі түрде табады. Логикалық және орындау уақытындағы қателерді талдайды, отладка құралдарын (мысалы, Visual Studio Debugger, breakpoints) еркін пайдаланады. Кодты оңтайландырып,

	кейде түсінбей қиналады.		өнімділікті арттыру үшін себептерді анықтай алады.
5. Мәселені шешу тәсілі	Дайын үлгілер мен нұсқаулықтарға сүйеніп жұмыс істейді. Қарапайым программаларды (мысалы, екі санның қосындысын табу) жаза алады, бірақ күрделі есептерді шешу кезінде қиындық көреді.	Мәселені өздігінен талдап, бірнеше тәсілмен шешуге тырысады. Функциялар мен массивтерді қолданып орта деңгейлі есептерді шеше алады. Шешімін жетілдіруге ұмтылады.	Есепті тиімді әрі оңтайлы жолмен шешеді. Тапсырмаға креативті және логикалық тұрғыдан қарайды. Күрделі программалық құрылымдар мен алгоритмдерді (мысалы, сұрыптау, іздеу, рекурсия) өз бетінше жобалай алады.
6. Жоба құрастыру (егер бар болса)	Қарапайым бір файлдан тұратын шағын программа жаза алады (мысалы, калькулятор, температураны түрлендіру, санның факториалын есептеу). Код көбіне негізгі функцияның (main()) ішінде жазылады.	Бірнеше файл мен модульден тұратын шағын жоба жасай алады. Функциялар, құрылымдар (struct) және кластарды (class) пайдалана отырып, жобаны бөліктерге бөледі. Мысалы, деректерді өңдеу немесе қарапайым ойын программа сын жасай алады.	Толыққанды жоба құрып, оны модульдік түрде ұйымдастырады. Файлдық жүйе (fstream), деректер құрылымдары (vector, map) мен объектіге бағытталған тәсілдерді тиімді пайдаланады. Жобаны тестілей алады және жетілдіру жолдарын ұсынады.тестілеу және жетілдіру кезеңдерін орындайды.
7. Құжаттау және таныстыру	Кодқа түсініктеме (комментарий) жазуды біледі, бірақ сирек қолданады. Программаның не істейтінін қысқаша түсіндіре алады, бірақ құрылымын нақты сипаттай алмайды.	Кодтың маңызды және күрделі бөліктеріне түсініктеме жазып отырады. Өз жобасының жалпы құрылымын және логикасын түсіндіріп бере алады. Негізгі функциялар мен кластардың рөлін сипаттай алады.	Жобаның архитектурасын толық сипаттайтын құжаттама (мысалы, README файлы, блок-схема, UML диаграмма) жасайды. Кодты басқа әзірлеушілерге немесе студенттерге түсінікті етіп таныстыра алады, топтық жұмыста өз идеясын дәлелдеп жеткізеді.
8. Шығармашылық пен зерттеу элементі	Тапсырманы стандартты түрде орындайды, жаңа идеялар аз.	Сайтты барлық адамға ыңғайлы етуге (accessibility) көңіл бөледі.	Программада жаңашыл тәсілдер, зерттеу және талдау элементтері айқын көрінеді..

Қосымша 2

«Web программалау» пәнінен HTML тілінен білімді бағалаудың бейімделген критерийлері.

Бағалау критерийі	**Бастапқы деңгей (Beginner)**	**Орта деңгей (Intermediate)**	**Жоғары деңгей (Advanced)**
HTML			
1. Негізгі ұғымдарды түсіну	Негізгі тегтерді біледі: <h1>, <p>, <a>, <img> (сурет), / (тізім). HTML-құжаттың неден тұратынын (<html>, <head>, <body>) түсінеді.	Мағынасы бар тегтерді (<header>, <footer>, <nav>) дұрыс қолданады. Сайтта формалар (<form>, <input>) жасай алады. Бетке видео мен аудио (<video>, <audio>) қоя алады.	HTML-дің сирек қолданылатын мүмкіндіктерін (Web Storage, Geolocation) біледі. Сайтты поисковиктерге (мысалы, Google) танымал ету үшін арнайы тегтерді қолданады.
2. Алгоритмдік ойлау	Берілген дизайн бойынша элементтерді бетке рет-ретімен қалай қоюды ойластырады.	Сайттың құрылымын алдын ала жоспарлап, оны бөліктерге бөле алады.	Сайттың тез жұмыс істеуін ойлап, HTML кодын оңтайландыруды жоспарлайды.
3. Код сапасы	Кодты қолмен жазады, бірақ кейде қателіктер жібереді. Кодты оқуға ыңғайлы болу үшін шегініс (отступ) жасайды.	Кодты қайталамауға тырысады. Класс аттарын мағынасына сай қояды (мысалы, BEM әдісі).	Таза, мағыналы және барлық стандартқа сай келетін HTML жазады.
4. Қате табу және жөндеу дағдысы (Debugging)	Браузердегі көрініске қарап, HTML-дегі жабылмаған тег сияқты оңай қателерді табады.	Браузердің DevTools құралын еркін қолданып, HTML-дегі кез келген элементті талдай алады.	Сайттың неге баяу жүктеліп жатқанын анықтап, себебін таба алады.
5. Мәселені шешу тәсілі	Дайын үлгілер мен нұсқаулықтарға сүйеніп жұмыс істейді. Қарапайым	Көптеген дайын блоктарды (карточка, меню) өз бетінше жасай алады.	Ерекше, стандартты емес дизайн элементтерін тек HTML және CSS арқылы жасай алады.

	бір беттік сайт жасай алады.		
6. Жоба құрастыру (егер бар болса)	Бір беттен тұратын жоба жасай алады (мысалы, өзі туралы парақша, рецепт беті).	Бірнеше беттен тұратын толыққанды сайт жасай алады (мысалы, портфолио, компания сайты).	Үлкен веб-қосымшалардың негізін (каркасын) жасай алады.
7. Құжаттау және таныстыру	Кодқа түсініктеме (комментарий) жазуды біледі, бірақ көп қолданбайды.	Кодтың күрделі жерлеріне түсініктеме жазып отырады. Өз жобасының құрылымын түсіндіріп бере алады.	Жобаның архитектурасын толық сипаттайтын құжаттама жасайды. Басқа әзірлеушілерге сабақ өткізе алады.
8. Шығармашылық пен зерттеу элементі	Интернеттен дайын кодтарды тауып, өз жобасына қоса алады.	Сайтты барлық адамға ыңғайлы етуді ойлайды (accessibility).	Веб-технологиялардың жаңалықтарын үнемі зерттеп, біліп отырады. Open-source жобаларға қатысады.

Қосымша 3

«Web программалау» пәнінен CSS тілінен білімді бағалаудың бейімделген критерийлері

Бағалау критерийі	**Бастапқы деңгей (Beginner)**	**Орта деңгей (Intermediate)**	**Жоғары деңгей (Advanced)**
1. Негізгі ұғымдарды түсіну	Қарапайым селекторларды (тег, класс, ID) біледі. Негізгі стильдерді бере алады: color (түс), font-size (әріп өлшемі), width (ен), height (биіктік).	Flexbox немесе Grid арқылы элементтерді басқара алады (мысалы, қатарға қою, ортаға жылжыту). Адаптивті дизайн жасай алады (сайт телефонда да, компьютерде де дұрыс көрінеді). Элементтің орнын (position) басқара алады.	Sass/SCSS сияқты препроцессорларды еркін қолданады. CSS-тің архитектурасын (ITCSS, SMACSS) түсінеді. Күрделі анимациялар, 3D-эффектілер жасай алады.
2. Алгоритмдік ойлау	Элементтерге стильдерді қалай қолдану керектігін түсінеді.	Алдымен сайттың телефонға арналған нұсқасын, содан кейін компьютерге арналған нұсқасын жасауды жоспарлайды.	Қайта қолдануға болатын, үлкен жобаларға арналған CSS компоненттер жүйесін құрастырады.
3. Код сапасы	Көбіне барлық стильдерді бір файлға ретсіз жазады. Бір кодты қайталай беруі мүмкін.	CSS кодын логикалық блоктарға бөліп, реттеп жазады. CSS айнымалыларын қолдана алады (мысалы, негізгі түсті бір жерде сақтап, бірнеше рет қолдану).	Кодты модульдерге бөліп жазады. Сайттың жылдамдығын арттыру үшін CSS-ті оңтайландыру (минимизация) әдістерін біледі.
4. Қате табу және жөндеу дағдысы (Debugging)	"Неге стилім жұмыс істемейді?" деп, мәндерін өзгертіп	DevTools арқылы стильдердің неге өзгермей	Браузердің Performance құралдарын

	көреді. Браузердің қарапайым DevTools құралын қолдана алады.	жатқанын, қай стиль маңыздырақ екенін (specificity) анықтай алады.	қолданып, анимацияның немесе беттің неге "қатып" қалатынын анықтай алады.
5. Мәселені шешу тәсілі	Дизайнда көрсетілген түс пен өлшемге сәйкес стильдерді қолданады.	Сайттың барлық браузерде бірдей көрінуін қамтамасыз ете алады.	Таза CSS-тің өзімен күрделі интерактивті элементтер (мысалы, ойын, график) жасай алады.
6. Жоба құрастыру (егер бар болса)	Жобада негізгі стильдерді (фон, мәтін түсі) қолдана алады.	Сайтты толықтай адаптивті ете алады. Элементтерге баяу қозғалыс, анимация (transition, animation) қоса алады.	Дизайн жүйесін (Design System) немесе UI компоненттер жинағын нөлден бастап құра алады.
7. Құжаттау және таныстыру	Өз жұмысын ауызша түсіндіріп бере алады.	Командадағыларға ыңғайлы болу үшін стильдер нұсқаулығын жасай алады.	Өзі жасаған компоненттер үшін толық сипаттама (мысалы, Storybook арқылы) дайындайды.
8. Шығармашылық пен зерттеу элементі	Басқа сайттарға қарап, ұқсас дизайндар жасауға тырысады	Жаңа CSS мүмкіндіктерін іздеп, өз жобаларында қолданып көреді.	CSS-тің мүмкіндіктерін толық қолданып, ерекше, инновациялық визуалды шешімдер жасайды.

Қосымша 4

“Визуалды және объектілі бағытталған программалау” пәнінен бейімделген бағалау критерийлері

Критерий	**Түсіндірме**
Функционалдық дұрыстық (Correctness)	Негізгі есептің дұрыс орындалуы, барлық тест нәтижесі сәйкес келуі
Қатеге қарсы тұрақтылық (Robustness)	Бағдарламаның дұрыс емес кірістерге (мысалы, бос мән, теріс сан, нөлге бөлу, т.б.) төзімді болуы
Код құрылымы мен архитектурасы (Code Design)	Класстар, объектілер, мұрагерлік, инкапсуляция, интерфейс принциптерінің сақталуы
Пайдаланушы интерфейсі мен визуалды шешім (UI/UX)	Пайдаланушыға ыңғайлы, қате туралы хабарламалар түсінікті түрде берілуі
Деректермен жұмыс	Деректер базасымен, файлмен, немесе API-мен дұрыс байланыс орнатылған ба?10
Құжаттама және түсіндірме (Documentation)	Кодтағы комментарийлер мен нұсқаулықтардың сапасы

Қосымша 5

“Мобильді қосымшалар жасау” пәнінен бейімделген бағалау критерийлері

“Мобильді қосымшалар жасау” пә*ні* бойынша әртүрлі тапсырмалар беріледі. Мысалы келесі тапсырмалар: Андроидта қарапайым бағдарлама жасау, Экранның бірнеше құрылымымен бағдарлама жасау, Таңдау компоненттерін пайдаланып көп терезелі бағдарлама жасау. Электрондық кітап жасау, Google Maps пайдаланып картографиялық бағдарлама жасау, SQLite мәліметтер қорымен жұмыс жасайтын программа жазу, JSON технологиясын пайдаланып прграмма жазу, JSOUP технологиясын пайдаланып прграмма жазу. Бағалау көбіне бағдарламаның бастапқы құрылымын (мысалы, кодтың кіріспе бөлігі мен синтаксисін) қаншалықты дұрыс қолдана алатындығын тексеріледі.

Критерий	**Түсіндірме**
Қолданбаның функционалдығы	Қосымша берілген тапсырманы орындай ма? Барлық негізгі функциялар дұрыс жұмыс істей ме?
Интерфейс және UX дизайны	UI элементтері логикалық орналасқан ба? Қолданушыға түсінікті және ыңғайлы ма?
Android компоненттерін пайдалану	Activity, Intent, Fragment, RecyclerView, Adapter сияқты компоненттер дұрыс және тиімді пайдаланылған ба?
Мәліметтермен жұмыс	SQLite, JSON, немесе JSOUP арқылы деректер дұрыс өңделіп, көрсетіле ме?
Google Services интеграциясы	Google Maps, Firebase, немесе басқа API дұрыс қосылған ба? Қателіксіз жұмыс істей ме?
Код құрылымы және сапасы	Код таза ма, құрылымдық па, қайта пайдалануға жарамды ма? Нейминг және класс ұйымдастыруы дұрыс па?
Қате өңдеу және тұрақтылық (Error handling)	try/catch, exception handling, validation, logging т.б. қолданылған ба?
Өнімділік пен тиімділік (Performance)	Қосымша жылдам жүктеле ме, артық ресурстар жұмсамай ма?
Шығармашылық және кеңейтілген функционал	Студент қосымша логика, жаңа модуль немесе ерекше дизайн элементін қосты ма?
Құжаттандыру және презентация	Қосымшаға README, түсіндірме (comments), немесе қысқаша бейне/слайд түрінде таныстырылым бар ма?