УДК 372.8
**DOI:10.58494/esai.23(7).2023.20**

*Турдубаева Кандалатхан Ташполотовна,*
*ОшМПУ, п.и.к., доцент*
*kandalathanturdubaeva@gmail.com*
*Маатова Гүлжамал Максатовна*
*ОшМПУ, магистрант*

## PYTHON ПРОГРАММАЛОО ТИЛИНДЕ КОД ЖАЗУУ

***Аннотация.*** *Мектептерде Python программалоо тилин үйрөтүү үчүн коддун мисалдарын окутуу зарыл. Коддун мисалдары окуучуларга программалоо концепцияларынын визуалдык чагылдырылышын камсыз кылат. Pythondун таза жана окула турган синтаксиси окуучуларга тилдин негизги курулуш блокторун түшүнүүгө жардам берип, коддун структурасын көрүүгө мүмкүнчүлүк берет. Бул мисалдар негизги программалоо түшүнүктөрүн камтыйт жана окуучулар үчүн Python тилин үйрөнүү үчүн баштапкы чекит катары кызмат кыла алат. Бул макалада бул суроого Pythondун негизги артыкчылыктары, анын ар кандай тармактарда колдонулушу жана тилди үйрөнүү үчүн ресурстары ачылып жооп берилет.*

***Түйүндүү сөздөр:*** *Python программалоо тили, коддун структурасы, тилди үйрөнүү ресурстары, арифметикалык операциялар, берилгендердин типтери, функциялар*

## НАПИСАНИЕ КОДА НА ЯЗЫКЕ ПРОГРАММИРОВАНИЯ PYTHON

***Аннотация.*** *Чтобы преподавать язык программирования Python в школах, необходимо преподавать примеры кода. Примеры кода предоставляют учащимся визуальное представление концепций программирования. Чистый и читаемый синтаксис Python помогает учащимся понять основные строительные блоки языка и увидеть структуру кода. Эти примеры охватывают основные концепции программирования и могут служить отправной точкой для изучения Python. Данная статья отвечает на этот вопрос, раскрывая основные преимущества Python, его применения в различных областях и ресурсы для изучения языка.*

***Ключевые слова:*** *язык программирования Python, структура кода, ресурсы изучения языка, арифметические операции, типы данных, функции.*

## WRITING CODE IN PYTHON PROGRAMMING LANGUAGE

***Annotation.*** *To teach the Python programming language in schools, it is necessary to teach code examples. Code examples provide students with a visual representation of programming concepts. Python's clean, readable syntax helps students understand the basic building blocks of the language and see the structure of the code. These examples cover basic programming concepts and can serve as a starting point for learning Python. This article answers this question by revealing the main benefits of Python, its applications in various fields, and resources for learning the language.*

***Keywords:*** *Python programming language, code structure, language learning resources, arithmetic operations, data types, functions.*

Python - бул үйрөнүүгө жана колдонууга бир топ жеңил болгон программалоо тили, ал көбүнчө кодду иштеп чыгуучулардын, ар кандай тиркемелерди жана тилдерде жазылган программалардын өндүрүмдүүлүгүн жогорулатуу үчүн колдонулат [1, 2].

Python, ар тараптуу жана кубаттуу программалоо тили, анын жөнөкөйлүгү, окула тургандыгы жана кеңири китепканалары менен иштеп чыгуучулардын сүйүктүү тили болуп калды. Python программалоо тилинде кодун жазуу программалоо тилин өздөштүрүүчүлөр

үчүн пайдалуу тажрыйба. Бул макалада биз Python программалоонун негизги каражаттары менен эффективдүү код жазуу үчүн тажрыйбаларга кайрылабыз [3].

Python тилинин синтаксиси так жана кыска болуп иштелип чыккан. Pythondo чегинүү абдан маанилүү, анткени ал код блоктору үчүн салттуу кашааларды алмаштырат. Туура чегинүү окууну жакшыртат жана ырааттуу коддоо стилин камсыз кылат.

```
if x>0:
    print(“on sandar”)
else:
    print(“ters sandar”)
```

Python тили динамикалык түрдө типтештирилген, демек, берилгендердин тибин ачык жарыялоонун кереги жок. Бүтүн сандар, жылып жүрүүчү чекиттер, саптар, тизмелер жана сөздүктөр сыяктуу ар кандай берилгендердин типтерин түшүнүү абдан маанилүү. Коддун ачыктыгын жакшыртуу үчүн өзгөрмөлөрдүн аталыштары баяндалган болушу керек.

```
a=25

b= “asan”
```

Python башкаруунун стандарттык агымынын структураларын колдойт, мисалы if билдирүүлөрү, циклдер жана функциялардын аныктамалары. Үчтүк оператор кыска шарттуу туюнтмаларды түзүүгө мүмкүндүк берет.

Python программалоо тилинин программасын жүктөгөндө монитордо>>> белгиси менен терезе ачылат. Бул жердеги>>> белгисинен кийин керектүү тапшырмаларды берсе болот.

Enterди басканда - ал тапшырманын натыйжасы чыгат.

Мисалы:

```
>>> 15+14  #кошуу амалы
29
>>> 15-3 #кемитүү амалы
12
>>>5*3 #көбөйтүү амалы
15
>>>5**2#Даражага көтөрүү
25
>>> 10/3#бөлүү амалы
3.3333333333335
>>>10//3#Бөлүүнүн бүтүн бөлүгүн чыгаруу
3
>>>10%3 #Бөлүүнүн калдыктуу бөлүгүн чыгаруу
1
```

Бул жерде # - белгисинен кийин жазылган маалымат – комментарий болуп эсептелинип, программа менен аткарылбайт.

Pythondo өтө чоң сандар менен да иштөө оңой. Мисалы:

```
>>>2**50  # 2нин 50 даражасы
1125899906842624
```

Дагы бир мисал «67 санынын цифраларынын суммасынын тап» деген тапшырма берилсе:

>>>67//10+67%10 көрүнүшүндө аткаруу мүмкүн.

Натыйжасы: 13

Же «123 санынын цифраларынын суммасын тапкыла?» десе

>>> 123//100+123%100//10+123%10 көрүнүшүндө жазылат.

Натыйжасы: 6

print () функциясы консолго текстти чыгаруу үчүн колдонулган Pythondo орнотулган функция. Бул жерде операторлор менен да иштөөгө мүмкүн.

>>>print('Salam aalam') #Enter ди басканда жообу чыгат:

Salam aalam Python тилинде программа түзүү үчүн төмөнкүчө амалдар аткарылат. Негизги менюдан **file** басычын басып, чыккан панелден **New file** же болбосо **ctrl+N** баскычтары терилет. Натыйжада жаңы терезе пайда болот. Ушул терезеге керектүү тапшырмаларды жазып программа түзөбүз.

```
a=15
b=13
f=a+b
print('f ')
```

Андан соң программага ат берип сактап коёбуз.

Ал үчүн: **File** бөлүмүнө **saveAs**... же **ctrl+shift+s** баскычтарын басып керектүү папканын ичине файлга ат коюп сакталат. Кийин **F5** баскычы же менюдагы **Run**бөлүмүндөгү**RunModuleF5** баскычы басылат. Натыйжа болсо төмөнкүчө чыгат:

Бул жердеги f=18 алынган керектүү натыйжа чыгат.

Эми берилген төрт орундуу сандын суммасын табуунун программасын түзүп көрөбүз.

```
a=1981
b=a//1000+a//100%10+a//10%10+a%10
print("Жообу=",b)
```

Программаны сактап F5 баскычын басабыз.

```
>>>
Жообу=19
```

Бул код менен жогорудагы ыкмалардын айырмасы, бул жерде берилген a=1981 мааниси компьютердин эсинде сакталып калат. Ал эми берилген 1981 санынын ордуна каалаган башка сандарды да колдонсо болот [4].

```
a = int ( input() )
b=a//1000+a//100%10+a//10%10+a%10
print("Жообу=",b)
print ( c )
```

Pythondo башка программалоо тилиндей математикалык операцияларды аткарууга болот, мисалы:

```
print(2 + 4)
print(10 - 6)
print(2 * 8)
print(10 / 3)
```

тиешелеш түрдө төмөнкүдөй жыйынтыктарды алабыз:

```
6
4
16
3.3333333333333335
```

Математикалык операцияларды түздөн-түз өзгөрмө чондуктарга да аткарууга болорун билүү маанилүү, мисалы:

```
a = 10
b = 3
print(a + b)
print(a - b)
print(a * b)
print(a / b)
print(a / a)
```

**Өзгөрмөлөр жана маалымат түрлөрү**. Өзгөрмө-бул аталышка ээ болгон эс тутумдун эң кичине бөлүкчөсү.Андагы сакталган берилиштер өзгөрмөнүн мааниси болот.

Python динамикалык түрдө өзгөрмөнүн маалымат түрүн анын маанисине жараша чыгарат. Мисалда ысым - сап, жаш - бүтүн сан, бийиктик - калкыма чекиттүү сан жана is_student - логикалык маани. Берилиштердин төмөнкү негизги типтерин бөлүп карайбыз:

**int**-бүтүн сан, **float**-чыныгы сан мааниле ри, **bool**-логикалык маанилер (**true**-ооба, **false**-жок ), **str**-бир же кош тырмакчага алынган символдордун жыйындысы

**Сандын квадратын эсептеген жөнөкөй функцияны карап көрөлү:**

```
def square(number):
return number ** 2 result = square(5)
print("Square of 5:", result)
```

**Шарттуу билдирүүлөр** (if, elif, else) белгилүү бир шарттардын негизинде программаңыздын агымын көзөмөлдөөгө мүмкүндүк берет. Бул мисалда ал сандын оң, терс же нөл экенин текшерет. Бул мисал окуучуларды шарттуу билдирүүлөр менен тааныштырат жана Pythonдо чечимди кантип кабыл алуу керек.

```
age = 16
if age >= 16:
   print("сиз документ тапшырсаңыз болот")
else:
    print("сиз 16 толгончо күтүп турасыз.")
```

**Тизмелер жана циклдер.** Тизмелер Pythonдо иреттелген жыйнактар. for цикли тизмедеги (сандар) ар бир элементтин үстүнөн кайталанат жана аларды басып чыгарат. Цикл элементтердин жыйнагы аркылуу итерациялоо процессин жөнөкөйлөтөт [4].

Бул мисал циклди көрсөтөт for, ал ырааттуулукту кайталоо үчүн колдонулат (мисалы, тизме).

```
for цикли менен кайталоону уюштуруу:
compania = ["Microsoft", "Google", "Oracle", "Apple"]
for item in compania:
print(item)
```

Ал эми бул программада окуучуларды тизмелер менен тааныштырат жана алар аркылуу кантип кайталоону көрсөтөт.

```
    compania = ["Microsoft", "Google", "Oracle", "Apple"]
for i in range(len(compania)):
print(compania [i])
```

Мында **len()** функциясын колдонуп тизменин элементтеринин узундугун алабыз.

**Функциялар.** Функциялар **def** ачкыч сөзү, андан кийин функциянын аты жана параметрлери аркылуу аныкталат. Кайтаруу билдирүүсү чалуучуга кайра маани жөнөтүү үчүн колдонулат. Функциялар коддун модулдуулугуна жана кайра колдонууга көмөктөшөт.

Бул мисалдар фундаменталдык программалоо түшүнүктөрүн камтыйт жана Pythonдо күчтүү пайдубалды куруу үчүн колдонулушу мүмкүн. Окуучулардын программалоо боюнча билимин өркүндөтүүгө ылайыкташтырылып, кеңейтилиши мүмкүн.

Бул фундаменталдык түшүнүктөрдү түшүнүү окуучуга Python боюнча бекем түшүнүк менен камсыз кылып, татаалыраак долбоорлор жана тиркемелер үчүн негиз түзөт. Түшүнүктү дагы тереңдетүү үчүн ар бир теманы андан ары изилдеп, өзүндүн кодуң менен эксперимент жасоо керек жана окуучуну да ошол багытка шыктандыруу зарыл.

Алгачкы версияларында Python негизинен скрипт тапшырмалары үчүн колдонулган, мисалы, күндөлүк тапшырмаларды автоматташтыруу, анын ичинде файлдар жана папкалар менен иштөө жана тексттик маалыматтарды иштетүү. Тил өнүккөн сайын адамдар Pythonду татаалыраак тапшырмалар үчүн колдоно башташты, анын ичинде веб-тиркемелерди иштеп чыгуу, илимий эсептөө, жасалма интеллект жана машина үйрөнүү.

Заманбап технологиялар чөйрөсүндө Жасалма Интеллект (AI) жана Машиналарды үйрөнүү (ML) дүйнөбүздү кайра түзүүчү күчтөр катары өзгөчөлөнөт. Бул

жетишкендиктердин өзөгүндө код саптарынын ортосундагы татаал алгоритмдер жатат. Python өзүнүн жарашыктуу синтаксиси жана кеңири экосистемасы менен AI жана ML потенциалын колдонууну каалаган иштеп чыгуучулар үчүн артыкчылыктуу тил катары пайда болду. AI жана ML үчүн тандоо тили катары Pythondун көтөрүлүшү өзүм билемдик эмес. Анын окулушу, жөнөкөйлүгү жана ар тараптуулугу аны интеллектуалдык системаларды жасоо үчүн идеалдуу холст кылат. Көптөгөн китепканалардын жана алкактардын бекем колдоосу анын коддоо пейзажындагы позициясын бекемдейт.

AI жана MLдин татаалдыктарын терендетүү коддоо процессин терең түшүнүүнү талап кылат. Python'дун экспрессивдүү синтаксиси татаал алгоритмдерди аракетке жарамдуу кодго которууну жеңилдетет. Python'дун интерактивдүү табияты иштеп чыгуучуларга өз моделдеринде үзгүлтүксүз эксперимент жүргүзүүгө, өзгөртүүгө жана итерациялоого мүмкүндүк берет.

**Интеллектти иштеткен алкактар.** Нейрондук тармактарды ишке ашырууга келгенде, TensorFlow жана PyTorch моделдерди иштеп чыгуу жана окутуу үчүн күчтүү пайдубалды сунуштайт. Салттуу машина үйрөнүү үчүн классификацияны, регрессияны, класстерлөө жана башкаларды ишке ашыруу үчүн комплекстүү инструменттер топтомун камсыз кылуучу scikit-learn эң жогорку бийликке ээ.

**Жасалма интеллект (AI) үчүн Python**. AIнын кеңири чөйрөсү табигый тилди иштетүүнү, компьютердик көрүнүштү жана өркүндөтүлгөн текстти иштетүүнү камтыйт. Python'дун бай китепкана экосистемасы бул ар түрдүү муктаждыктарга жооп берет, анын ичинде NLTK, OpenCV жана spaCy негизги оюнчулар катары өзгөчөлөнөт. Иштеп чыгуучулар бул китепканаларды тилди түшүнгөн, сүрөттөрдү чечмелеген жана текстти кылдаттык менен иштеткен интеллектуалдык системаларды түзүү үчүн колдоно алышат.

AI үчүн коддоо машиналарга маалыматтарды кабыл алуу, ой жүгүртүү жана үйрөнүү жөндөмүн берген алгоритмдерди ишке ашырууну камтыйт. Python'дун шамдагайлыгы иштеп чыгуучуларга татаал AI концепцияларын аткарылуучу кодго оной которууга мүмкүндүк берет, бул үлгүлөрдү тааный турган, негизделген чечимдерди кабыл ала турган жана табигый тилде өз ара аракеттене ала турган системаларды түзүүгө көмөктөшөт.

**Machine Learning, AIнын бир бөлүгү**, тутумдарга маалыматтардан үйрөнүүгө жана жакшыртууга мүмкүндүк берген алгоритмдерге багытталган. Python'дун ML арсеналы, мисалы, scikit-learn сыяктуу китепканалар, маалыматтарды даярдоо процессин, алгоритмди тандоону жана моделди тактоо процессин жеңилдетет. Тилдин интерактивдүү табияты иштеп чыгуучуларга ар кандай ыкмалар менен эксперимент жүргүзүүгө мүмкүнчүлүк берип, финансылык божомолдордон баштап жекелештирилген сунуштарга чейинки чечимдерди иштеп чыгууга алып келет [5,6].

Python өзүнүн жөнөкөйлүгү менен белгилүү, бул аны программалоону жаңы баштагандар үчүн өзгөчө жагымдуу кылат. Анын негизги артыкчылыктары жөнөкөй синтаксисин, кыска кодду жана окулушуна басым жасоону камтыйт.

Python синтаксисинин жөнөкөйлүгү, бул кодду окууга мүмкүнчүлүк берет. Python иштеп чыгуучуларынын негизги максаты - тилди мүмкүн болушунча жеткиликтүү жана окула турган кылуу. Бул бизге түшүнүктүү жана так семантикасы бар тилди түзүүгө мүмкүндүк берди, бул кодду түшүнүү процессин абдан жөнөкөйлөтөт.

Python тармал кашаалардын ордуна код блокторун аныктоо үчүн боштуктарды колдонот, бул Java же C++ сыяктуу көптөгөн башка программалоо тилдеринде кеңири таралган. Бул коддун тазалыгын жана түзүлүшүн өбөлгө түзөт. Python жана башка программалоо тилдериндеги код мисалдары. Келгиле, Python жана Java тилдеринде "ар бир үчүн" циклинин жөнөкөй мисалын карап көрөлү.

Python:

```
numbers = [1, 2, 3, 4, 5]
For num in numbers:For  (int num: numbers ) {
print  (num)
  }
```

Java:

```
int[] numbers  = {1, 2, 3, 4, 5};

System.out.println(сан)
```

Эки мисалда биз сандар массивинин элементтерин чыгарып жатабыз. Бирок, Python туюнтмасы кыскараак жана окууга оңой. Python маалымат түрүнүн декларациясын талап кылбайт жана ар бир цикл табигыйраак сезилет, бул Pythonду жазууну жана үйрөнүүнү өзгөчө женилдетет.

Pythondун ар тараптуулугу, колдонуунун жөнөкөйлүгү жана кубаттуу китепканалары аны ар түрдүү домендер боюнча окуучулар үчүн идеалдуу тандоо болуп саналат. Маалыматтарды манипуляциялоодон машина үйрөнүүсүнө чейин, Python татаал берилиштер топтомдорунан түшүнүктөрдү ачуу үчүн зарыл болгон куралдарды жана ресурстарды камсыз кылат. Технология өнүккөн сайын, илимдин жана аналитиканын келечегин түзүүдө Pythondун ролу ого бетер маанилүү болуп калат. Тажрыйбалуу мугалимсизби же жаңыдан келе жаткан окуучусузбу, Pythondун потенциалын колдонуу ишиңизди жаңы бийиктиктерге көтөрүп, инновациялардын жана ачылыштардын жаны доорун ачат.

## Корутунду

Акылдуу системаларга суроо-талап өсүп жаткандыктан, бул домендер үчүн Pythonду өздөштүрүү жөн гана артыкчылык эмес, зарылчылык болуп саналат. Окуучулардын жасалма интеллекттин жана машинаны үйрөнүүнүн тереңдигине сүңгүп кирүүсү кубандырат жана Python алар үчүн бул жолду ачкан ишенимдүү шериги боло алат . Ошентип, коддоо инструментиңизди алыңыз, алгоритмдер дүйнөсүнө чөмүлүңүз жана Python сиздин жан жолдошуңуз болсун.

## Пайдаланылган адабияттар:

1. Python программалоо тилин үйрөнүү үчүн окуу куралы / Л. Самыкбаева,А. Беляев,А.Палитаев,И.Ташиев, С.Маматов–«Сорос-Кыргызстан» Фонду. -2019. – 80 б.
2. Информатиканын предметтик стандартын ишке ашыруунун методикалык маселелери Ибирайым кызы Айжан Бишкек. -2017
3. Информатика боюнча олимпиадалык маселер жыйнагы Н.К Аркабаев П.С. Панков Ош. - 2019
4. Python программалоо тилин үйрөнөбүз Турдубаева К.Т., Кудуев А.Ж., Жусупбек кызы Ж, Ырысбаева А Ош. -2021
5. https://gb.ru/blog/python/ Язык программирования Python: особенности и перспективы
6. https://infourok.ru/statya-na-temu-yazyk-programmirovaniya-python-6867703.html Язык программирования Python.