



Python® for Programmers

*with introductory
AI case studies*

- ▶ Natural Language Processing
- ▶ Data Mining Twitter®
- ▶ IBM® Watson™
- ▶ Machine Learning with scikit-learn®
- ▶ Deep Learning with Keras
- ▶ Big Data with Hadoop®,
Spark™, NoSQL and the Cloud
- ▶ Internet of Things (IoT)
- ▶ Python Standard Library
- ▶ Data Science Libraries:
NumPy, Pandas, SciPy,
NLTK, TextBlob, Tweepy,
Matplotlib, Seaborn,
Folium and more

PAUL DEITEL • HARVEY DEITEL

www.EBooksWorld.ir

টোরি

ছবি

প্রোগ্রামারদের জন্য পাইথন®

আর্নিং পাথ

ফার্স্ট এবং ডিল

আলো

জিনিসপত্র

সমর্থন

সাইন আউট

প্লেলিস্ট
ডিটেল ডেভেলপার সিরিজ[®]

গল্প

মতামত

প্রোগ্রামারদের জন্য পাইথন

উপার্জনের পৃষ্ঠা
প্লেলিস্ট

হার্ডে ডিটেল
অফার্স এবং ডিল



এটিৎ

সমর্থন

সাইন আউট



নির্মাতা এবং বিক্রেতারা তাদের অপিক পণ্যগুলিকে আলাদা করার জন্য যে সমস্ত পদবী ব্যবহার করেন সেগুলির অনেকগুলিকে ট্রেডমার্ক হিসেবে দাবি করা হয়। যেখানে এই বইটিতে সেই পদবীগুলি উল্লেখ করা হয়েছে এবং প্রকাশক ট্রেডমার্ক দাবি সম্পর্কে অবগত ছিলেন, সেখানে পদবীগুলি "আর্নিং পাথস" এর প্রাথমিক বড় অক্ষরে বা সমস্ত বড় অক্ষরে মুদ্রিত হয়েছে।

অফার এবং ডিল

লেখক এবং প্রকাশক এই বইটি তৈরিতে যত্নবান হয়েছেন, তবে কোনও ধরণের স্পষ্ট বা অন্তর্নিহিত ওয়ারেন্টি দেননি এবং ক্রটি বা আলোকপাতের ভুলের জন্য কোনও দায় গ্রহণ করেননি। এখানে উল্লেখিত তথ্য বা প্রোগ্রামগুলির ব্যবহারের সাথে সম্পর্কিত বা এর ফলে উদ্ভূত আনুষঙ্গিক বা ফলস্বরূপ ক্ষতির জন্য কোনও দায়বদ্ধতা গ্রহণ করা হবে না।

সমর্থন

এই শিরোনামটি প্রচুর পরিমাণে কেনার বিষয়ে তথ্যের জন্য, অথবা বিশেষ বিক্রয় সুযোগের জন্য (যার মধ্যে ইলেকট্রনিক সংস্করণ; কাস্টম সাইন ক্রস্টেক্স ডিজাইন অন্তর্ভুক্ত থাকতে পারে; এবং

আপনার ব্যবসা, প্রশিক্ষণের লক্ষ্য, বিপণন কেন্দ্রবিন্দু, অথবা ব্র্যান্ডিং আগ্রহের জন্য নির্দিষ্ট বিষয়বস্তু), অনুগ্রহ করে আমাদের কর্পোরেট বিক্রয় বিভাগের সাথে orpsales@pearsoned.com অথবা (800) 3823419 নম্বরে যোগাযোগ করুন।

সরকারি বিক্রয় সংক্রান্ত অনুসন্ধানের জন্য, অনুগ্রহ করে overnmentsales@pearsoned.com-এ যোগাযোগ করুন।

মার্কিন যুক্তরাষ্ট্রের বাইরে বিক্রয় সম্পর্কে প্রশ্নের জন্য, অনুগ্রহ করে ntlcs@pearson.com এ যোগাযোগ করুন।

ওয়েবে আমাদের সাথে যোগাযোগ করুন: informit.com

লাইব্রেরি অফ কংগ্রেস কন্ট্রোল নম্বর: 2019933267

কপিরাইট © ২০১৯ পিয়ারসন এডুকেশন, ইনকর্পোরেটেড।

সর্বস্বত্ত্ব সংরক্ষিত। এই প্রকাশনাটি কপিরাইট দ্বারা সুরক্ষিত, এবং যেকোনো নিষিদ্ধ পুনরুৎপাদন, পুনরুদ্ধার ব্যবস্থায় সংরক্ষণ, অথবা যেকোনো আকারে বা যেকোনো উপায়ে, ইলেকট্রনিক, যান্ত্রিক, ফটোকপি, রেকর্ডিং, অথবা অনুরূপভাবে ট্রান্সমিশন করার আগে প্রকাশকের কাছ থেকে অনুমতি নিতে হবে। অনুমতি, অনুরোধ ফর্ম এবং পিয়ারসন এডুকেশন প্লোবাল রাইটস অ্যান্ড পারমিশনস ডিপার্টমেন্টের মধ্যে উপযুক্ত যোগাযোগ সম্পর্কিত তথ্যের জন্য, অনুগ্রহ করে www.pearsoned.com/permissions/dept।

eitel এবং doublethumbsup বাগ হল Deitel এর নিবন্ধিত ট্রেডমার্ক এবং

অ্যাসোসিয়েটস, ইনকর্পোরেটেড।

পাইথন লোগোটি পাইথন সফটওয়্যার ফাউন্ডেশনের সৌজন্যে।

প্রচ্ছদ নকশা করেছেন পল ডেইটেল, হার্ভে ডেইটেল এবং চুটি প্রাসার্টসিথ।

অ্যাগস্যান্ড্রু/শাটারস্টকের প্রচ্ছদ চিত্রকর্ম

আইএসবিএন১৩: ৯৭৮০১৩৫২২৪৩০৫

আইএসবিএন১০: ০১৩৫২২৪৩০০

ইলিস্ট

অরি

পুনরুদ্ধার করা

"ওদের মধ্যে সোনা আছে থার পাহাড়!"^১

উৎস অজানা, প্রায়শই ভুলভাবে মার্ক টোয়েনের নামে দারী করা হয়।

পথগুলি

টাকা এবং ডিল

প্রোগ্রামারদের জন্য পাইথনে আপনাকে স্বাগতম! এই বইটিতে, আপনি আজকের সবচেয়ে আকর্ষণীয়, অগ্রণী কম্পিউটিং প্রযুক্তির সাথে হাতে-কলমে শিখবেন এবং আপনি পাইথনে প্রোগ্রামিং করবেন—যা বিশ্বের সবচেয়ে জনপ্রিয় ভাষাগুলির মধ্যে একটি এবং তাদের মধ্যে দ্রুত বর্ধনশীল ভাষা।

আলো

ডেভেলপাররা প্রায়শই দ্রুত আবিষ্কার করে যে তারা পাইথন পছন্দ করে। তারা এর প্রকাশ ক্ষমতা, পঠনযোগ্যতা, সংক্ষিপ্ততা এবং ইন্টারঅ্যাক্সিভিটির প্রশংসন। তারা ওপেনসোর্স সফ্টওয়্যার ডেভেলপমেন্টের জগৎ পছন্দ করে যা বিভিন্ন অ্যাপ্লিকেশন ক্ষেত্রের জন্য পুনর্ব্যবহারযোগ্য সফ্টওয়্যারের দ্রুত বর্ধনশীল ভিত্তি তৈরি করছে।

সমর্থন

সাইন আউট

বহু দশক ধরে, কিছু শক্তিশালী প্রবণতা বিদ্যমান। কম্পিউটার হার্ডওয়্যার দ্রুত, সন্তো এবং ছোট হয়ে আসছে। ইন্টারনেট ব্যাক্টুইথ দ্রুত বৃহত্তর এবং সন্তো হয়ে উঠেছে। এবং "ওপেন সোর্স" আন্দোলনের মাধ্যমে মানসম্পন্ন কম্পিউটার সফ্টওয়্যার আরও প্রচুর এবং মূলত বিনামূল্যে বা প্রায় বিনামূল্যে হয়ে উঠেছে। সীমাই, "ইন্টারনেট অফ থিংস" প্রতিটি কল্পনাপ্রসূত ধরণের কোটি কোটি ডিভাইসকে সংযুক্ত করবে। এগুলি দ্রুত বর্ধনশীল গতি এবং পরিমাণে বিপুল পরিমাণে ডেটা তৈরি করবে।

আজকের কম্পিউটিং-এ, সর্বশেষ উন্নতবনগুলি হল "সবকিছু সম্পর্কে তথ্য" - ডেটা বিজ্ঞান, ডেটা বিশ্লেষণ, বিগ ডেটা, রিলেশনাল ডাটাবেস (SQL), এবং NoSQL এবং NewSQL ডাটাবেস, যার প্রতিটি আমরা পাইথন প্রোগ্রামিংয়ের একটি উন্নতবনী পদ্ধতির সাথে সম্মৌখন করি।

তথ্য বিজ্ঞান দক্ষতার জন্য প্রয়োজনীয় চাকরি

২০১১ সালে, ম্যাককিনসে ফ্লোরাল ইনসিটিউট তাদের প্রতিবেদন প্রকাশ করে, "বিগ ডেটা: উন্নতবন, প্রতিযোগিতা এবং উৎপাদনশীলতার পরবর্তী সীমানা।" এতে তারা বলেছে, "শুধুমাত্র মার্কিন যুক্তরাষ্ট্রেই গভীর বিশ্লেষণাত্মক দক্ষতা সম্পর্ক ১৪০,০০০ থেকে ১৯০,০০০ লোকের অভাব রয়েছে এবং সেই সাথে বড় ডেটা বিশ্লেষণ করে তাদের ফলাফলের ভিত্তিতে সিদ্ধান্ত নেওয়ার জন্য ১.৫ মিলিয়ন ব্যবস্থাপক এবং বিশ্লেষকের অভাব রয়েছে।"

এই অবস্থা এখনও অব্যাহত রয়েছে। ২০১৮ সালের আগস্টের "লিঙ্কডইন ওয়ার্কহোর্স রিপোর্ট" বলছে যে মার্কিন যুক্তরাষ্ট্রে ডেটা সায়েলে দক্ষতা সম্পর্ক ১,৫০,০০০ এরও বেশি লোকের ঘাসতি রয়েছে। আরুইএম, বার্নিং প্লাস টেকনোলজিস এবং বিজনেসহায়ার এভুকেশন ফোরামের ২০১৭ সালের একটি প্রতিবেদনে বলা হয়েছে যে ২০২০ সালের মধ্যে মার্কিন যুক্তরাষ্ট্রে ডেটা সায়েলে দক্ষতার প্রয়োজন এমন লক্ষ লক্ষ নতুন চাকরি তৈরি হবে।

৮

২

<https://www.mckinsey.com/~media/McKinsey/Business%20Functions/McKinsey%20Digital/Our%20P�tch%20Digital%20Strategies%20in%20the%20New%20Normal%20-%20McKinsey%20View%20from%20the%20CEO%20Office.pdf>

বলা

^৯ <https://economicgraph.linkedin.com/resources/linkedinworkforce-employment-trends-2018>

^{১০} https://www.burningglass.com/wp-content/uploads/The_Quant_Crunch.pdf (পৃষ্ঠা ৩)

মডিউলার স্থাপত্য

বইটির মডুলার স্থাপত্য (বইয়ের ভিতরের প্রচদে সূচিপত্রের প্রাফিকটি দেখুন) আমাদের বিভিন্ন পেশাদার দর্শকদের বিভিন্ন চাহিদা পূরণে সহায়তা করে।

হ্যাপ্টার্স ১-০ পাইথন প্রোগ্রামিং কভার করে। এই প্রতিটি অধ্যায়ে একটি সংক্ষিপ্ত ডুমিকা রয়েছে

ডেটা সায়েল বিভাগটিতে কৃতিম বুদ্ধিমতা, মৌলিক বর্ণনামূলক পরিসংখ্যান, কেন্টীয় প্রবণতা এবং বিচ্ছুরণের পরিমাপ, সিমুলেশন, স্ট্যাটিক এবং ডায়নামিক

ভিজ্যালাইজেশন, CSV ফাইলগুলির সাথে কাজ করা, ডেটা অব্রেগ এবং ডেটা র্যাসলিং এর জন্য পার্সা, টাইম সিরিজ এবং www.EBooksWorld.ir এর সাথে পরিচয়

করিয়ে দেওয়া হয়েছে।

লিনিয়ার রিপ্রেশন প্রয়োগ করুন। এগুলি আপনাকে ১-৬ অধ্যায়ে ডেটা সায়েন্স, এআই, বিগ ডেটা এবং ক্লাউড কেস স্টাডিওর জন্য প্রস্তুত করতে সাহায্য করবে, যা আপনাকে সম্পূর্ণ কেস স্টাডিওতে বাস্তব জগতের ডেটাসেট ব্যবহারের সুযোগ প্রদান করে।

পাইথন হ্যাপ্টার ১-৫ এবং হ্যাপ্টার ৬-৭ এর কয়েকটি গুরুত্বপূর্ণ অংশ কভার করার পরে, আপনি সক্ষম হবেন

হ্যাপ্টার ১১-৬ - এ কেস স্টাডিও উল্লেখযোগ্য অংশগুলি পরিচালনা করুন। "অধ্যায়

এই ভূমিকার "নির্ভরতা" বিভাগটি প্রশিক্ষকদের তাদের পেশাদার কোর্স পরিকল্পনা করতে সাহায্য করবে
বইটির অন্য স্থাপত্যের প্রেক্ষাপট।

হ্যাপ্টার ১১-৬ অসাধারণ, শক্তিশালী, সমসাময়িক উদাহরণে ভরপুর। তারা হাতে-কলমে উপস্থাপন করে-

প্রাকৃতিক ভাষা প্রক্রিয়াকরণ, তথ্যের মতো বিষয়ের উপর বাস্তবায়ন কেস স্টাডিও উপর

তুটোর মাইনিং, আইবিএমের ওয়াটসনের সাথে জ্ঞানীয় কম্পিউটিং, প্রেশিভিনাস এবং রিপ্রেশন সহ তত্ত্বাবধানে থাকা মেশিন লার্নিং,
ক্লাস্টারিং সহ তত্ত্বাবধানহীন মেশিন লার্নিং, কনভেলিউশনাল নিউরাল নেটওয়ার্ক সহ গভীর শিক্ষা, পুনরাবৃত্ত নিউরাল নেটওয়ার্ক সহ গভীর
শিক্ষা, হ্যাডোপ, স্পার্ক এবং নোএসকিউএল ডাটাবেস সহ বিগ ডেটা, ইন্টারেন্ট অফ থিংস এবং আরও অনেক কিছু। পথের সাথে,
আপনি ডেটা সায়েন্সের পদ এবং ধারণাগুলির একটি বিস্তৃত সাক্ষরতা অর্জন করবেন, সংক্ষিপ্ত সংজ্ঞা থেকে শুরু করে ছেট,

মাঝারি এবং বৃহৎ প্রোগ্রামগুলিতে ধারণাগুলির ব্যবহার পর্যবেক্ষণ। বইয়ের বিস্তারিত সূচীপত্র এবং সূচী ব্রাউজ করলে আপনি
কভারেজের বিস্তৃত সম্পর্কে ধারণা পাবেন।

মূল বৈশিষ্ট্য

KIS (সহজ রাখুন), KIS (ছোট রাখুন), KIT (প্রসঙ্গিক রাখুন)

- সহজ রাখুন—বইয়ের প্রতিটি দিকেই আমরা সরলতা এবং স্পষ্টতার জন্য চেষ্টা করি। উদাহরণস্বরূপ, যখন আমরা প্রাকৃতিক ভাষা
প্রক্রিয়াকরণ উপস্থাপন করি, তখন আমরা আরও জটিল NLTK-এর পরিবর্তে সহজ এবং স্বজ্ঞাত TextBlob লাইব্রেরি ব্যবহার
করি। আমাদের গভীর শিক্ষায়
- উপস্থাপনায়, আমরা টেনসরফোর দিয়ে কেরাসকে বেশি পছন্দ করি। সাধারণভাবে, যখন একাধিক লাইব্রেরি পারে
একই ধরণের কাজ সম্পাদনের জন্য ব্যবহার করা হয়, আমরা সবচেয়ে সহজে সহজে ব্যবহার করি।
- ছোট রাখো—বইটির ৫০টি উদাহরণের বেশিরভাগই ছোট—প্রায়শই মাত্র কয়েকটি লাইন
কোড, তাৎক্ষণিক ইন্টারেক্ষিভ আইপিথন প্রতিক্রিয়া সহ। আমরা 40 টি বৃহত্তর স্ক্রিপ্টও অন্তর্ভুক্ত করি
এবং গভীর কেস স্টাডি।
- বিষয়বিত্তিক থাকুন—আমরা সাম্প্রতিক পাইথন প্রোগ্রামিং এবং ডেটা সায়েন্সের বই পড়েছি এবং প্রায় ১৫,০০০ সাম্প্রতিক
প্রবন্ধ, গবেষণাপত্র, শ্বেতপত্র, ডিপিও, ব্লগ পোস্ট, ফোরাম পোস্ট এবং ডকুমেন্টশনের টুকরো ব্রাউজ করেছি,
পড়েছি বা দেখেছি। এর ফলে আমরা পাইথন, কম্পিউটার বিজ্ঞান, ডেটা সায়েন্স, এআই, বিগ ডেটা এবং ক্লাউডের
"নাচি" বুঝতে পেরেছি।
সম্প্রদায়।

তাৎক্ষণিক প্রতিক্রিয়া: আইপিথন অর্বেষণ, আবিষ্কার এবং পরীক্ষা-নিরীক্ষা

- এই বইটি থেকে শেখার আদর্শ উপায় হল এটি পড়া এবং কোড উদাহরণগুলি সমান্তরালভাবে চালানো।
বইটি জুড়ে, আমরা IPython ইন্টারপ্রেটার ব্যবহার করি, যা পাইথন এবং এর বিস্তৃত লাইব্রেরিগুলির সাথে দ্রুত অর্বেষণ,
আবিষ্কার এবং পরীক্ষা-নিরীক্ষার জন্য একটি বৰুৱাপূর্ণ, তাৎক্ষণিক প্রতিক্রিয়া ইন্টারেক্ষিভ মোড প্রদান করে।
- বেশিরভাগ কোড ছোট, ইন্টারেক্ষিভ আইপিথন সেশনে উপস্থাপন করা হয়। আপনার লেখা প্রতিটি কোড মিপেটের জন্য,
আইপিথন তাৎক্ষণিকভাবে এটি পড়ে, মূল্যায়ন করে এবং ফলাফল প্রিন্ট করে। এই তাৎক্ষণিক প্রতিক্রিয়া আপনার
মনোযোগ ধরে রাখে, শেখার গতি বাড়ায়, দ্রুত প্রোটোইপিং সহজতর করে এবং সংক্ষেপ্যার ডেভেলপমেন্ট
প্রক্রিয়াকে দ্রুততর করে।
- আমাদের বিশেষ সর্বদা লাইভকোড পদ্ধতির উপর জোর দেয়, লাইভ ইনপুট এবং আউটপুট সহ সম্পূর্ণ, কার্যকরী
প্রোগ্রামগুলির উপর দৃষ্টি নিবন্ধন করে। আইপিথনের "জানু" হল এটি এমনকি মিপেটগুলিকে কোডে পরিণত করে যা
প্রতিটি লাইনে প্রবেশ করার সাথে "জীবন্ত হয়ে ওঠে"। এটি শেখার প্রচার করে এবং পরীক্ষা-নিরীক্ষাকে
উৎসাহিত করে।

পাইথন প্রোগ্রামিং এর মৌলিক বিষয়সমূহ

- প্রথমত, এই বইটিতে পাইথনের উপর সমৃদ্ধ কভারেজ রয়েছে।
- আমরা পাইথনের প্রোগ্রামিং মডেলগুলি নিয়ে আলোচনা করব—প্রক্রিয়াগত প্রোগ্রামিং, কার্যকরী-

- আমরা সর্বোত্তম অনুশীলন ব্যবহার করি, বর্তমান প্রবাদের উপর জোর দিই।
- বই জুড়ে যথাযথভাবে ফাংশনালস্টাইল প্রোগ্রামিং ব্যবহার করা হয়েছে। অধ্যায় 8-এর একটি চার্টে পাইখনের বেশিরভাগ মূল ফাংশনালস্টাইল প্রোগ্রামিং ক্ষমতা এবং যেসব অধ্যায়ে আমরা প্রাথমিকভাবে বেশিরভাগই আলোচনা করব।

538 কোড উদাহরণ

- ৫০৮ এর মাধ্যমে আপনি পাইখনের একটি আকর্ষণীয়, চ্যালেঞ্জিং এবং বিনোদনমূলক ভূমিকা পাবেন।
বাস্তব জগতের উদাহরণ, যার মধ্যে রয়েছে পৃথক স্লিপেট থেকে শুরু করে সারগর্ড কল্পিউটার বিজ্ঞান, ডেটা বিজ্ঞান,
কৃত্রিম বুদ্ধিমত্তা এবং বিগ ডেটা কেস স্টাডি।
- আপনি AI, বিগ ডেটা এবং ক্লাউড প্রযুক্তি যেমন প্রাকৃতিক ভাষা প্রক্রিয়াকরণ, ডেটা মাইনিং টুইটার, মেশিন লার্নিং,
ডিপ লার্নিং, Hadoop, MapReduce, Spark, IBM Watson, কী ডেটা সায়েন্স লাইব্রেরি (NumPy, pandas,
SciPy, NLTK, TextBlob, spaCy, Textatistic, Tweepy, Scikitlearn, Keras), কী ডিজ্যুয়ালাইজেশন লাইব্রেরি
(Matplotlib, Seaborn, Folium) এবং আরও অনেক কিছু দিয়ে গুরুত্বপূর্ণ কাজগুলি আকর্মণ করবেন।

ইংরেজি ব্যাখ্যার পক্ষে ভারী গণিত এড়িয়ে চলুন

- আমরা গণিতের ধারণাগত সারাংশ ধরে রাখি এবং আমাদের উদাহরণগুলিতে এটি কাজে লাগাই। আমরা
পরিসংখ্যান, NumPy, SciPy, pandas এবং আরও অনেক লাইব্রেরি ব্যবহার করে এটি করি, যা গণিতিক জটিলতা লুকিয়ে
রাখে। সুতরাং, ঐতিহাসিক রিপ্রেজেন্টেশনের মতো গণিতিক কোশলগুলির অনেক সুবিধা আপনার পক্ষে তাদের পিছনের গণিত না
জেনেই পাওয়া সহজ। মেশিনলার্নিং এবং ডিপ-লার্নিং উদাহরণগুলিতে, আমরা এমন বস্তু তৈরি করার উপর ফোকাস করি
যা আপনার জন্য গণিত করে "

দৃশ্য।"

ডিজ্যুয়ালাইজেশন

- ৬৭টি স্ট্যাটিক, ডাইনামিক, অ্যানিমেটেড এবং ইন্টারেক্টিভ ডিজ্যুয়ালাইজেশন (চার্ট, গ্রাফ, ছবি, অ্যানিমেশন
ইত্যাদি) আপনাকে ধারণাগুলি বুবলতে সাহায্য করে।
- নিম্ন-স্তরের গ্রাফিক্স প্রোগ্রামিং এর চিকিৎসা অন্তর্ভুক্ত করার পরিবর্তে, আমরা Matplotlib, Seaborn, pandas এবং
Folium (ইন্টারেক্টিভ মানচিত্রের জন্য) দ্বারা উত্পাদিত উচ্চ-স্তরের ডিজ্যুয়ালাইজেশনের উপর মনোযোগ দিই।
- আমরা একটি শিক্ষামূলক হাতিয়ার হিসেবে ডিজ্যুয়ালাইজেশন ব্যবহার করি। উদাহরণস্বরূপ, আমরা একটি
গতিশীল ডাইরোলি সিমুলেশন এবং বার চার্টে বৃহৎ সংখ্যার সূত্রকে "জীবিত" করি। মোলের সংখ্যা বাড়ার সাথে
সাথে, আপনি দেখতে পাবেন মোট মোলের প্রতিটি মুখের শতাংশ বীরে শতাংশ বীরে 16.667% (1/6) এর কাছাকাছি চলে
আসছে এবং শতাংশের প্রতিনিধিত্বকারী বারগুলির আকার সমান হয়ে যাচ্ছে।
- তথ্য অনুসন্ধান এবং পুনরংপাদনযোগ্য গবেষণার ফলাফল যোগাযোগের জন্য বৃহৎ তথ্যে ডিজ্যুয়ালাইজেশন অত্যন্ত
গুরুত্বপূর্ণ, যেখানে তথ্যের সংখ্যা লক্ষ লক্ষ, লিলিয়ন বা তারও বেশি হতে পারে। একটি সাধারণ কথা হল যে একটি ছবি
হাজার শব্দের সমান - বৃহৎ তথ্যে, একটি ডিজ্যুয়ালাইজেশন একটি ডাটাবেসে লিলিয়ন, ট্রিলিয়ন বা তারও বেশি আইটেমের
সমান হতে পারে।
ডিজ্যুয়ালাইজেশন আপনাকে "ডেটা থেকে ৪০,০০০ ফুট উপরে উড়ে" "বড় আকারে" দেখতে এবং আপনার ডেটা সম্পর্কে
জানতে সক্ষম করে। বর্ণনামূলক পরিসংখ্যান সাহায্য করে কিন্তু বিআন্তিক হতে পারে। উদাহরণস্বরূপ,
অ্যানসকষ্টের চতুর্ভুজটি ডিজ্যুয়ালাইজেশনের মাধ্যমে দেখায় যে উল্লেখযোগ্যভাবে ডিন
ডেটাসেটগুলিতে প্রায় একই রকম বর্ণনামূলক পরিসংখ্যান থাকতে পারে।

৫

https://en.wikipedia.org/wiki/A_picture_is_worth_a_thousand_words.

৬ https://en.wikipedia.org/wiki/Anscombe%27s_quartet.

- আমরা ডিজ্যুয়ালাইজেশন এবং অ্যানিমেশন কোড দেখাই যাতে আপনি নিজেরটি বাস্তবায়ন করতে পারেন। আমরা
সোস্কোড ফাইল এবং জুপিটার নোটবুক হিসাবে অ্যানিমেশনগুলি সরবরাহ করি, যাতে আপনি সুবিধাজনকভাবে
কোড এবং অ্যানিমেশন প্যারামিটারগুলি কাস্টমাইজ করতে পারেন, অ্যানিমেশনগুলি পুনরায় কার্যকর করতে পারেন
এবং পরিবর্তনগুলির প্রভাব দেখতে পারেন।

- তুমি বাস্তব জগতের অনেক ডেটাসেট এবং ডেটা উৎসের সাথে কাজ করবে। অনলাইনে পরীক্ষা-নিরীক্ষার জন্য প্রচুর পরিমাণে বিনামূলের ওপেন ডেটাসেট পাওয়া যায়। এর মধ্যে কিছু আমরা যেসব সাইটের উল্লেখ করি, তাতে শত শত বা হাজার হাজার ডেটাসেট থাকে।
- আপনার ব্যবহার করা অনেক লাইব্রেরিতে পরীক্ষার জন্য জনপ্রিয় ডেটাসেট থাকে।
- আপনি তথ্য সংগ্রহ এবং বিশ্লেষণের জন্য প্রস্তুত করার জন্য প্রয়োজনীয় পদক্ষেপগুলি শিখবেন, সেই তথ্য বিশ্লেষণ করুন অনেক কৌশল ব্যবহার করে, আপনার মডেলগুলিকে সুর করুন এবং আপনার ফলাফলগুলি কার্যকরভাবে যোগাযোগ করুন, বিশেষ করে ডিজিয়ালাইজেশনের মাধ্যমে।

গিটহাব

- আপনার প্রকল্পগুলিতে অন্তর্ভুক্ত করার জন্য (এবং ওপেনসোর্স সম্প্রদায়ে আপনার কোড অবদান রাখার জন্য) ওপেনসোর্স কোড খুঁজে বের করার জন্য গিটহাব একটি দুর্দান্ত স্থান। এটি সফ্টওয়্যার ডেভেলপারদের অন্তর্গারের একটি গুরুত্বপূর্ণ উপাদান যার সংস্করণ নিয়ন্ত্রণ সরঞ্জাম রয়েছে যা ডেভেলপারদের দলগুলিকে ওপেনসোর্স (এবং ব্যক্তিগত) প্রকল্প পরিচালনা করতে সহায়তা করে।
- আপনি আসাধারণ পরিসরের ফ্রি এবং ওপেনসোর্স পাইথন এবং ডেটা সায়েন্স লাইব্রেরি এবং বিনামূল্যে, ফ্রিট্রায়াল এবং ফ্রিমিয়াম সফ্টওয়্যার এবং ক্লাউড পরিষেবা ব্যবহার করবেন। অনেক লাইব্রেরি GitHub-এ হোস্ট করা আছে।

হাতে-কলমে ক্লাউড কম্পিউটিং

- বেশিরভাগ বিগ ডেটা অ্যানালিটিক্স ক্লাউডে ঘটে, যেখানে আপনার অ্যাপ্লিকেশনগুলির প্রয়োজনীয় হার্ডওয়্যার এবং সফ্টওয়্যারের পরিমাণ গতিশীলভাবে পরিমাপ করা সহজ। আপনি বিভিন্ন ক্লাউড-ভিত্তিক পরিষেবাগুলির সাথে কাজ করবেন (কিছু প্রত্যক্ষভাবে এবং কিছু পরোক্ষভাবে), যার মধ্যে রয়েছে টুইটার, গুগল ট্রালেন্ট, আইবিএম ওয়াটসন, মাইক্রোসফ্ট অ্যাজুরে, ওপেনম্যাপকোয়েস্ট, জিওপি, ডিওয়েট.আইও এবং পাবনাব।
- আমরা আপনাকে বিনামূল্যে ট্র্যায়াল অথবা ফ্রিমিয়াম ক্লাউড পরিষেবা ব্যবহার করতে উৎসাহিত করি। আমরা এমন পরিষেবা পছন্দ করি যেখানে কেউকে কার্ডের প্রয়োজন হয় না কারণ আপনি দুর্ঘটনাক্রমে বড় বিলের ঝুঁকি নিতে চান না। যদি আপনি এমন কোনও পরিষেবা ব্যবহার করার সিদ্ধান্ত নেন যার জন্য ক্রেডিট কার্ডের প্রয়োজন হয়, তাহলে নিশ্চিত করুন যে আপনি যে স্ট্রাট বিনামূল্যে ব্যবহার করছেন তা স্বয়ংক্রিয়ভাবে একটি অর্থপ্রদানকারী স্তরে চলে যাবে না।

ডাটাবেস, বিগ ডেটা এবং বিগ ডেটা অবকাঠামো

- আইবিএম (নভেম্বর ২০১৬) অনুসারে, বিশ্বের ১০% তথ্য গত দুই বছরে তৈরি করা হয়েছে।
১ প্রমাণ ইঙ্গিত দেয় যে তথ্য তৈরির গতি দ্রুততর হচ্ছে।

৭

<https://public.dhe.ibm.com/common/ssi/ecm/wr/en/wrl12345usen/watson->

গ্রাহকসংশ্লিষ্টাওয়াটসনবিপণনকাগজপত্রএবংপ্রতিবেদন-

r12345usen20170719.pdf.

৮

- ২০১৬ সালের মার্চ মাসে অ্যানালিটিক্স টাইকের একটি নিবন্ধ অনুসারে, পাঁচ বছরের মধ্যে ৫০ বিলিয়নেরও বেশি ডিডাইস ইন্টারনেটের সাথে সংযুক্ত হবে এবং ২০২০ সালের মধ্যে আমরা গ্রহের প্রতিটি ব্যক্তির জন্য প্রতি সেকেন্ডে ১.৭ মেগাবাইট নতুন ডেটা তৈরি করব!

৮ <https://analyticsweek.com/content/bigdatafacts/>.

- আমরা SQLite এর সাথে রিলেশনাল ডাটাবেস এবং SQL এর একটি চিকিৎসা অন্তর্ভুক্ত করি।
- ডেটাবেসগুলি হল আপনার প্রক্রিয়াজাতকরণের জন্য বিশাল পরিমাণ ডেটা সংরক্ষণ এবং পরিচালনা করার জন্য গুরুত্বপূর্ণ বৃহৎ ডেটা অবকাঠামো। রিলেশনাল ডেটাবেসগুলি কাঠামোগত ডেটা প্রক্রিয়া করে - এগুলি বৃহৎ ডেটা অ্যাপ্লিকেশনগুলিতে অসংগঠিত এবং আধা-সংগঠিত ডেটার জন্য তৈরি নয়। তাই, বৃহৎ ডেটা বিবর্তনের সাথে সাথে, এই ধরনের ডেটা দক্ষতার সাথে পরিচালনা করার জন্য NoSQL এবং NewSQL ডাটাবেস তৈরি করা হয়েছিল। আমরা একটি NoSQL এবং NewSQL ওভারভিউ এবং একটি MongoDB JSON ড্রুমেট ডাটাবেসের সাথে একটি যান্ত্রিক কেস স্টাডি অন্তর্ভুক্ত করেছি। MongoDB হল সবচেয়ে জনপ্রিয় NoSQL। ডাটাবেস।

- আমরা chapter 16, www.EBooksWorld.ir- এ বিগ ডেটা হার্ডওয়্যার এবং সফ্টওয়্যার অবকাঠামো নিয়ে আলোচনা করব।

"

অধ্যায়

কৃতিম বুদ্ধিমত্তা কেস স্টাডিও

- কেস স্টাডিওর ১-৫ নম্বর অধ্যায়ে, আমরা কৃতিম বুদ্ধিমত্তাৰ বিষয়গুলি উপস্থাপন কৰি, যাৰ মধ্যে রয়েছে প্ৰাকৃতিক ভাষা প্ৰক্ৰিয়াকৰণ, ডেটা মাইনিং টুইটোৱ সেন্টিমেন্ট সম্পাদন কৰাৰে বিশ্লেষণ, আইবিএম ওয়াটসনেৰ সাহায্যে জ্ঞানীয় কম্পিউটিং, তত্ত্বাবধানে থাকা মেশিন শেখা, তত্ত্বাবধানবিহীন মেশিন লাৰ্নিং এবং গতীৱ শিক্ষা। chapter 16 উপস্থাপনা বিগ ডেটা হাৰ্ডওয়্যার এবং সফ্টওয়্যার অৰকাঠামো যা কম্পিউটোৱ বিজ্ঞানীদেৱ এবং তথ্য বিজ্ঞানীদেৱ জন্য অগ্ৰণী কৃতিম বুদ্ধিমত্তা ভিত্তিক সমাধান বাস্তবায়ন।

অন্তনির্মিত সংগ্রহ: তালিকা, টুপল, সেট, অডিধান

- আজকাল বেশিৱভাগ অ্যাপ্লিকেশন ডেভেলপারদেৱ কাস্টম ডেটা স্ট্রাকচাৱ তৈৰি কৰাৰ খুব একটা কাৰণ নেই। বইটিতে পাইথনেৰ অন্তনির্মিত ডেটা স্ট্রাকচাৱ - তালিকা, টিপল, অডিধান এবং সেট - এৱ একটি সমৃদ্ধ দুটি অধ্যায়েৰ ট্ৰিটমেন্ট রয়েছে যাৰ সাহায্যে বেশিৱভাগ ডেটা-স্ট্রাকচাৱিৰ কাজ সম্পন্ন কৰা যেতে পাৰে।

NumPy অ্যারে এবং পার্সা সহ অ্যারে-ওৱিয়েন্টেড প্ৰোগ্ৰামিং সিৱিজ/ডেটাফ্ৰেম

- আমৰা ওপেনসোৰ্স লাইভ্ৰেি থেকে তিনটি মূল ডেটা স্ট্রাকচাৱেৰ উপৰও মনোযোগ দিই - NumPy অ্যারে, পার্সা সিৱিজ এবং পার্সা ডেটাফ্ৰেম। এগুলি ডেটা সায়েন্স, কম্পিউটোৱ সায়েন্স, কৃতিম বুদ্ধিমত্তা এবং বিগ ডেটাতে ব্যাপকভাৱে ব্যবহৃত হয়। NumPy বিল্ট-ইন পাইথন তালিকাৰ তুলনায় দুই সুৰেৱ উচ্চতৰ কৰ্মক্ষমতা প্ৰদান কৰে।
- আমৰা chapter 7- এ NumPy অ্যারেৰ একটি সমৃদ্ধ ট্ৰিটমেন্ট অন্তৰ্ভুক্ত কৰেছি। অনেক লাইভ্ৰেি, যেমন পার্সা, NumPy-তে তৈৰি। অধ্যায় ৭-৯- এ ডেটা সায়েন্স বিভাগগুলিৰ ভূমিকা পার্সা সিৱিজ এবং ডেটাফ্ৰেম প্ৰাৰ্থন কৰুন, যা NumPy অ্যারেগুলিৰ সাথে থাকে বাকি অধ্যায়গুলিতে ব্যবহৃত।

ফাইল প্ৰক্ৰিয়াকৰণ এবং সিৱিয়ালাইজেশন

- হ্যাপটাৱ ৯ টেক্স্টফাইল প্ৰক্ৰিয়াকৰণ উপস্থাপন কৰে, তাৰপৰ বস্তুগুলিকে কীভাৱে সিৱিয়ালাইজ কৰতে হয় তা প্ৰদৰ্শন কৰে জনপ্ৰিয় JSON (জাভাস্ক্রিপ্ট অৰজেক্ট মোটেশন) ফৰ্ম্যাট ব্যবহাৱ কৰে। ডেটা সায়েন্স অধ্যায়গুলিতে JSON প্ৰায়শই ব্যবহৃত হয়।
- অনেক ডেটা সায়েন্স লাইভ্ৰেি আপনাৰ পাইথন প্ৰোগ্ৰামে ডেটাসেট লোড কৰাৰ জন্য বিল্ট-ইন ফাইল প্ৰসেসিং ক্ষমতা প্ৰদান কৰে। প্লাইন টেক্স্ট ফাইল ছাড়াও, আমৰা পাইথন স্ট্যান্ডাৰ্ড লাইভ্ৰেিৰ CSV মডিউল এবং পার্সা ডেটা সায়েন্স লাইভ্ৰেিৰ ক্ষমতা ব্যবহাৱ কৰে জনপ্ৰিয় CSV (commaseparated value) ফৰ্ম্যাটে ফাইল প্ৰক্ৰিয়া কৰি।

অৰজেক্ট-ভিত্তিক প্ৰোগ্ৰামিং

- আমৰা পাইথন ওপেনসোৰ্স কমিউনিটি যে বিপুল সংখ্যক মূল্যবান ক্লাসগুলিকে ইন্ডাস্ট্ৰি স্ট্যান্ডাৰ্ড ক্লাস লাইভ্ৰেিতে প্ৰক্ৰিয়া কৰেছে তাৰ উপৰ জোৱ দিই। আপনি কোন লাইভ্ৰেিগুলি আছে তা জানাৰ উপৰ মনোযোগ দেবেন, আপনাৰ অ্যাপগুলিৰ জন্য আপনাৰ প্ৰয়োজনীয় ক্লাসগুলি বেছে নেবেন, বিদ্যমান ক্লাসগুলি থেকে বস্তু তৈৰি কৰবেন (সাধাৱণত এক বা দুটি লাইন কোডে) এবং সেগুলিকে "লাফিয়ে পড়ুন, নাচুন এবং গান কৰুন"। এই অৰজেক্ট-ভিত্তিক প্ৰোগ্ৰামিং আপনাকে দ্রুত এবং সংক্ষিপ্তভাৱে চিতাৰকৰ্ষক অ্যাপ্লিকেশন তৈৰি কৰতে সক্ষম কৰে, যা পাইথনেৰ আবেদনেৰ একটি উল্লেখযোগ্য অংশ।
- এই পদ্ধতিৰ সাহায্যে, আপনি মেশিন লাৰ্নিং, ডিপ লাৰ্নিং এবং অন্যান্য এআই প্ৰযুক্তি ব্যবহাৱ কৰে বিভিন্ন ধৰণেৰ আকৰ্ষণীয় সমস্যাৰ দ্রুত সমাধান কৰতে পাৱবেন, যাৰ মধ্যে রয়েছে বকৃতা শীকৃতি এবং কম্পিউটোৱ দৃষ্টিভঙ্গিৰ মতো জ্ঞানীয় কম্পিউটিং চ্যালেঞ্জ।

অৰজেক্ট-ওৱিয়েন্টেড প্ৰোগ্ৰামিং

- কাস্টম ক্লাস ডেভেলপ কৰা একটি গুৰুত্বপূৰ্ণ অৰজেক্ট-ওৱিয়েন্টেড প্ৰোগ্ৰামিং দক্ষতা, উত্তৰাধিকাৰ, পলিমৱফিজম এবং ভাক টাইপিংয়েৰ সাথে। আমৰা পৰ্ব ১০-এ এগুলো নিয়ে আলোচনা এবং একটি মজাৰ কাৰ্ড অন্তৰ্ভুক্ত রয়েছে-
- পৰ্ব ১০-এ ডেক্সেটোৱ সাথে ইউনিট টেস্টিং নিয়ে আলোচনা এবং একটি মজাৰ কাৰ্ড অন্তৰ্ভুক্ত রয়েছে-

- হ্যাপ্টার্স ১১- ৬ এর জন্য কেবল কয়েকটি সহজবোধ্য কাস্টম ইনস সংজ্ঞা প্রয়োজন। পাইথনে,
আপনি সন্তুত পূর্ণাঙ্গ অবজেক্ট-ওরিয়েন্টেড প্রোগ্রামিংয়ের চেয়ে অবজেক্ট-ভিত্তিক প্রোগ্রামিং পদ্ধতি বেশি ব্যবহার
করবেন।

পুনরুৎপাদনযোগ্যতা

- সাধারণভাবে বিজ্ঞানে, এবং বিশেষ করে তথ্য বিজ্ঞানে, পুনরুৎপাদন করার প্রয়োজন আছে
পরীক্ষা-নিরীক্ষা এবং গবেষণার ফলাফল, এবং সেই ফলাফলগুলিকে কার্যকরভাবে জানানো। জুপিটার
এটি করার জন্য নেটবুক একটি পছন্দের মাধ্যম।
- আমরা বই জুড়ে জুপিটার নেটবুকস এবং ডকারের মতো প্রোগ্রামিং কৌশল এবং সফ্টওয়্যারের প্রেক্ষাপটে
অজননযোগ্যতা নিয়ে আলোচনা করেছি।

কর্মক্ষমতা

- একই কাজ সম্পাদনের জন্য বিভিন্ন পদ্ধতির কর্মক্ষমতা তুলনা করার জন্য আমরা বেশ কয়েকটি উদাহরণে %timeit
প্রোফাইলিং টুল ব্যবহার করি। অন্যান্য কর্মক্ষমতা সম্পর্কিত আলোচনার মধ্যে রয়েছে জেনারেটর
এরূপেশন, NumPy অ্যারে বনাম পাইথন তালিকা, মেশিনলার্নিং এবং ডিপলার্নিং মডেলের কর্মক্ষমতা এবং Hadoop
এবং Spark ডিস্ট্রিবিউটেড-কম্পিউটিং কর্মক্ষমতা।

বিগ ডেটা এবং সমান্তরালতা

- এই বইটিতে, আপনার নিজস্ব সমান্তরালকরণ কোড লেখার পরিবর্তে, আপনি টেনসরফ্লো-এর উপর চলমান কেরাসের
মতো লাইব্রেরি এবং হ্যাডোপ এবং স্পার্কের মতো বড় ডেটা সরঞ্জামগুলিকে আপনার জন্য সমান্তরাল ক্রিয়াকলাপ
পরিচালনা করতে দেবেন। এই বড় ডেটা/এআই যুগে, বিশাল ডেটা অ্যাপ্লিকেশনগুলির প্রক্রিয়াকরণের প্রয়োজনীয়তাগুলি
মাল্টিকোর দ্বারা প্রদত্ত প্রকৃত সমান্তরালতার সুবিধা গ্রহণের দাবি করে।
প্রসেসর, গ্রাফিক্স প্রসেসিং ইউনিট (GPU), টেনসর প্রসেসিং ইউনিট (TPU)
এবং ক্লাউড বিশাল কম্পিউটারের ক্লাউটার। কিছু বড় ডেটা টাক্স থাকতে পারে
বিপুল পরিমাণ তথ্য বিশ্লেষণের জন্য সমান্তরালভাবে কাজ করছে হাজার হাজার প্রসেসর
দ্রুততার সাথে।

অধ্যায় নির্ভরতা

আপনি যদি একজন প্রশিক্ষক হন যিনি পেশাদার প্রশিক্ষণ কোর্সের জন্য আপনার সিলেবাস পরিকল্পনা করছেন অথবা কোন
অধ্যায়গুলি পড়বেন তা নির্ধারণকারী একজন ডেভেলপার হন, তাহলে এই বিভাগটি আপনাকে সর্বোত্তম সিদ্ধান্ত নিতে সাহায্য
করবে। অনুগ্রহ করে বইয়ের ডিতের প্রচ্ছদে এক পৃষ্ঠার রঙিন সুচিপত্রটি পড়ুন—এটি আপনাকে বইয়ের অন্যন্য স্থাপত্যের
সাথে দ্রুত পরিচিত করবে। অধ্যায়গুলি ক্রমানুসারে শেখানো বা পড়া সবচেয়ে সহজ। তবে, ডেটা সায়েন্সের ভূমিকা বিভাগের
বেশিরভাগ বিষয়বস্তু

হ্যাপ্টার ১-০ এর শ্রেষ্ঠ প্রান্ত এবং হ্যাপ্টার ১১-৬ এর কেস স্টাডিতে শুধুমাত্র হ্যাপ্টার ১-৫ প্রয়োজন

এবং হ্যাপ্টার ৬-০ এর ছোট অংশ যেমনটি নিচে আলোচনা করা হয়েছে।

পর্ব ১: পাইথনের মৌলিক বিষয়গুলি দ্রুত শুরু

আমরা আপনাকে সমস্ত অধ্যায়গুলি ক্রমানুসারে পড়ার পরামর্শ দিচ্ছি:

- **হ্যাপ্টার ১, কম্পিউটার এবং পাইথনের ভিত্তি**
স্থাপনকারী ধারণাগুলি উপস্থাপন করে,
অধ্যায় ১১-৬ - এ কৃতিম বুদ্ধিমত্তা এবং ক্লাউডভিত্তিক কেস স্টাডি। অধ্যায়টি আরও
আইপিথন ইন্টারপ্রেটার এবং জুপিটার নেটবুকের টেস্টড্রাইভ অন্তর্ভুক্ত।
- **পর্ব ২, পাইথন প্রোগ্রামিং**
পাইথন প্রোগ্রামিং এর ভূমিকা, পাইথন প্রোগ্রামিং উপস্থাপন করে
মূল ভাষার বৈশিষ্ট্যগুলি চিন্তিত করে কোড উদাহরণ সহ মৌলিক বিষয়গুলি।
- **অধ্যায় ৩, নিয়ন্ত্রণ বিবৃতি**
পাইথনের নিয়ন্ত্রণ বিবৃতি উপস্থাপন করে এবং
মৌলিক তালিকা প্রক্রিয়াকরণ প্রবর্তন করে।
- **অধ্যায় ৪, ফাংশন**
কাস্টম ফাংশন প্রবর্তন করে, সিমুলেশন উপস্থাপন করে
র্যান্ডম সংখ্যা তৈরির কৌশল এবং টুপল প্রবর্তন করে
মৌলিক বিষয়।
- **অধ্যায় ৫, সিকেয়েল্স**: তালিকা এবং টিপলস, পাইথনের অন্তর্নির্মিত তালিকা এবং টিপল সংগ্রহগুলিকে
আরও বিশদে উপস্থাপন করে এবং ফাংশনালস্টাইল প্রোগ্রামিং প্রবর্তন শুরু করে।

আর্ট ২: পাইথন ডেটা স্ট্রাকচার, স্ট্রিং এবং ফাইল

নিম্নলিখিতটি পাইথন হ্যাপ্টার ৬-৭ এর জন্য ইন্টারচ্যাপ্টর নির্ভরতাগুলির সারসংক্ষেপ এবং

ধরে নিচ্ছ যে আপনি chapters 1-5 পড়েছেন।

- অধ্যায় ৬, অভিধান এবং সেট—এই অধ্যায়ে ডেটা সায়েলের ভূমিকা অংশটি হল অধ্যয়ের বিষয়বস্তুর উপর নির্ভরশীল নয়।
- অধ্যায় ৭, NumPy সহ অ্যারে-ওরিয়েন্টেড প্রোগ্রামিং—ডেটা সায়েলের ভূমিকা বিভাগে অভিধান (অধ্যায় ৬) এবং অ্যারে (অধ্যায় ৭) প্রয়োজন।
- অধ্যায় ৮, স্ট্রিং: একটি গতীর দৃষ্টিভঙ্গ—ডেটা সায়েলের ভূমিকা বিভাগে কাঁচামাল প্রয়োজন স্ট্রিং এবং রেগুলার এক্সপ্রেশন (অ্যাকশন ৮.১১-১.১২), এবং পার্সাস সিরিজ এবং ডেটাফ্রেম বৈশিষ্ট্যগুলি অ্যাকশন ৭.১৪ এর ডেটা সায়েলের ভূমিকা থেকে।
- অধ্যায় ৯, ফাইল এবং ব্যতিক্রম—JSON সিরিয়ালাইজেশনের জন্য, অভিধানের মৌলিক বিষয়গুলি বোঝা দরকারী (অনুচ্ছেদ ৯.৩) এবং ইচ্ছাও, Intro to Data Science বিভাগে বিল্টইন ওপেন ফাংশন এবং with statement (অনুচ্ছেদ ৯.৩) এবং pandas DataFrame বৈশিষ্ট্যগুলি ection ৭.১৪ এর Intro to Data Science থেকে প্রয়োজন।

পার্ট ৩: পাইথন হাই-এন্ড টপিকস

নিম্নলিখিতটি পাইথন হ্যাপ্টার ১০ এর জন্য ইন্টারচ্যাপ্টর নির্ভরতাগুলির সারসংক্ষেপ করে এবং ধরে নেয়

যে তুমি পর্ব ১-৫ পড়েছো।

- অধ্যায় ১০, অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং—ডেটা সায়েলের ভূমিকা বিভাগ ইন্টেন্ট থেকে ডেটা সায়েল বিভাগ ৭.১৪ পর্যন্ত পার্সাস ডেটাফ্রেম বৈশিষ্ট্যগুলির প্রয়োজন। প্রশিক্ষকদের শুধুমাত্র ক্লাস এবং অবজেক্ট কভার করতে চাইলে, অনুচ্ছেদ ১০.১-০.৬ উপস্থাপন করতে পারেন। উত্তরাধিকার, পলিমরফিজম এবং ডাক টাইপিংয়ের মতো আরও উন্নত বিষয়গুলি কভার করতে চাইলে, অনুচ্ছেদ ১০.৭-০.৯ উপস্থাপন করতে পারেন। অনুচ্ছেদ ১০.১০-০.১৫ অতিরিক্ত উন্নত দৃষ্টিভঙ্গি প্রদান করে।

পর্ব ৪: এআই, ক্লাউড এবং বিগ ডেটা কেস স্টাডিজ

১১-৬ নম্বর পর্বের আন্তঃঅধ্যায় নির্ভরতার নিম্নলিখিত সারাংশ ধরে নেয় যে

তুমি হ্যাপ্টার্স ১-৫ পড়েছো। হ্যাপ্টার্স ১১-৬ এর বেশিরভাগেরই অভিধানের মৌলিক বিষয়গুলো প্রয়োজন।

বিভাগ ৬.২ থেকে।

- পর্ব ১১, ন্যাচারাল ল্যাঙ্গুয়েজ প্রসেসিং (NLP), পার্সাস ডেটাফ্রেম বৈশিষ্ট্য ব্যবহার করে বিভাগ ৭.১৪ এর ডেটা সায়েলের ভূমিকা থেকে।
- হ্যাপ্টার ১২, ডেটা মাইনিং টুইটার, ইক্ষণ থেকে পার্সাস ডেটাফ্রেম বৈশিষ্ট্য ব্যবহার করে ১.৪ এর ডেটা সায়েলের ভূমিকা, স্ট্রিং মেথড জয়েন (অনুচ্ছেদ ৮.৯), JSON মৌলিক বিষয়গুলি (ection 9.৫), TextBlob (ection 11.২) এবং Word ক্লাউড (ection 11.৩)! বেশ কিছু উদাহরণ উত্তরাধিকারের মাধ্যমে একটি শ্রেণী সংজ্ঞায়িত করার প্রয়োজন (অধ্যায় 10)।
- hapter 13, IBM Watson এবং Cognitive Computing, বিল্টইন ফাংশন open ব্যবহার করে এবং with বিরুতি (অংশ ৯.৩)।
- অধ্যায় ১৪, মেশিন লার্নিং: শ্রেণীবিভাগ, রিষ্ণেশন এবং ক্লাস্টারিং, ব্যবহার NumPy অ্যারের মৌলিক বিষয় এবং পদ্ধতি অনন্য (hapter 7), pandas ডেটাফ্রেম বৈশিষ্ট্যগুলি থেকে ection 7.14 এর Intro to Data Science এবং Matplotlib ফাংশন সাবপ্লট (অংশ ১০.৬)।
- hapter 15, ডিপ লার্নিং, NumPy অ্যারের মৌলিক বিষয়গুলি (hapter 7), স্ট্রিং প্রয়োজন পদ্ধতি যোগদান (অনুচ্ছেদ ৮.৯), হ্যাপ্টার 14 থেকে সাধারণ মেশিনলার্নিং ধারণা এবং হ্যাপ্টার 14 এর কেস স্টাডি থেকে বৈশিষ্ট্য: kNearest Neighbors এর সাথে শ্রেণীবিভাগ এবং সংখ্যা ডেটাসেট।
- hapter 16, ig ডেটা: Hadoop, Spark, NoSQL এবং IoT, স্ট্রিং পদ্ধতি স্প্লিট ব্যবহার করে (অনুচ্ছেদ 6.2.৭), ম্যাট্রিক্সের ফানকঅ্যানিমেশন, অনুচ্ছেদ 6.৪ এর ইন্টেন্ট টু ডেটা সায়েল, পার্সাস সিরিজ এবং ডেটাফ্রেম বৈশিষ্ট্য, অনুচ্ছেদ 7.14 এর ইন্টেন্ট টু ডেটা সায়েল, স্ট্রিং

method join (ection 8.9), json মডিউল (ection 9.5), NLTK stop words (ection 1.2.13) এবং hapter 12 থেকে, Twitter

প্রামাণীকরণ, Tweepy's StreamListener ক্লাস

স্ট্রিমিং টুইট এবং জিওগ্রাফি এবং ফোলিয়াম লাইব্রেরির জন্য। কয়েকটি উদাহরণের জন্য উত্তরাধিকারের মাধ্যমে একটি ক্লাস সংজ্ঞায়িত করা

প্রয়োজন (যাপ্টর ১০), তবে আপনি যাপ্টর ১০ না পড়েই আমাদের দেওয়া ক্লাস সংজ্ঞাগুলি অনুকরণ করতে পারেন।

জুপিটার নোটবুক

আপনার সুবিধার জন্য, আমরা কমান্ডলাইন IPython ইন্টারপ্রেটার এবং Jupyter Notebooks (.ipynb) ফাইলের সাথে ব্যবহারের জন্য Python সোর্স কোড (.py) ফাইলে বইটির কোড উদাহরণ প্রদান করছি যা আপনি আপনার যেৱে ব্রাউজারে লোড করে কার্যকর করতে পারবেন।

জুপিটার নোটবুকস একটি বিনামূল্যে, ওপেনসোর্স প্রকল্প যা আপনাকে যেৱে ব্রাউজারে ফ্রেট এবং সুবিধাজনকভাবে কোড প্রবেশ, সম্পাদনা, সম্পাদনা, ডিবাগিং এবং পরিবর্তন করার জন্য টেক্সট, গ্রাফিক্স, অডিও, ভিডিও এবং ইন্টারেক্ষনের কোডিং কার্যকৰিতা একত্রিত করতে সক্ষম করে।
"জুপিটার কী?" নিবন্ধ অনুসারে:

জুপিটার বৈজ্ঞানিক গবেষণা এবং তথ্য বিশ্লেষণের জন্য একটি মানদণ্ড পরিষ্ঠিত হয়েছে। এটি গণনা এবং যুক্তিকে একত্রিত করে, আপনাকে "গণনামূলক ব্যবহার" তৈরি করতে দেয়, এবং এটি সতীর্থ এবং সহযোগীদের মধ্যে কার্যকরী সংস্থানের বিতরণের সমস্যাটিকে সহজ করে তোলে।

১ <https://www.oreilly.com/ideas/whatisjupyter>

আমাদের অভিজ্ঞতায়, এটি একটি চমৎকার শেখার পরিবেশ এবং ফ্রেট প্রোটোটাইপিং টুল। এই কারণে, আমরা Eclipse, Visual Studio, PyCharm বা Spyder এর মতো এতিয়বাহী IDE-এর পরিবর্তে Jupyter Notebooks ব্যবহার করি। শিক্ষাবিদ এবং পেশাদাররা ইতিমধ্যেই গবেষণার ফলাফল ভাগ করে নেওয়ার জন্য Jupyter Notebooks সমর্থন এতিয়বাহী ওপেনসোর্স কমিউনিটি মেকানিজমের মাধ্যমে প্রদান করা হয় (এই ভূমিকার পরে "Getting Jupyter Help" দেখুন)। সংস্থানের ইনস্টলেশনের বিশদ বিবরণের জন্য এই ভূমিকার পরে থাকা "Before You Begin" বিভাগটি দেখুন এবং বইয়ের উদাহরণগুলি চালানোর তথ্যের জন্য বিভাগ 1.5- এ টেক্সট্রাইভগুলি দেখুন।

০ <https://jupyter.org/community>

সহযোগিতা এবং ভাগাভাগি ফলাফল

শিল্প, সরকার বা শিক্ষাক্ষেত্রে ডেটা অ্যানালিস্টগুলি পদে বা স্থানান্তরিত ডেভেলপারদের জন্য দলগতভাবে কাজ করা এবং গবেষণার ফলাফলের সাথে যোগাযোগ করা উভয়ই গুরুত্বপূর্ণ:

- আপনার তৈরি নোটবুকগুলি কেবল ফাইলগুলি অনুলিপি করে বা GitHub এর মাধ্যমে দলের সদস্যদের মধ্যে ভাগ করে নেওয়া সহজ।

- কোড এবং অন্তর্দৃষ্টি সহ গবেষণার ফলাফলগুলি nbviewer (<https://nbviewer.jupyter.org>) এবং GitHub এর মতো টুলের মাধ্যমে স্ট্যাটিক ওয়েব পৃষ্ঠা হিসাবে ভাগ করা যেতে পারে - উভয়ই স্বয়ংক্রিয়ভাবে নোটবুকগুলিকে ওয়েব পৃষ্ঠা হিসাবে রেন্ডার করে।

পুনরুৎপাদনযোগ্যতা: জুপিটার নোটবুকের জন্য একটি শক্তিশালী কেস

তথ্য বিজ্ঞানে, এবং সাধারণভাবে বিজ্ঞানের ক্ষেত্রে, পরীক্ষা-নিরীক্ষা এবং অধ্যয়ন পুনরুৎপাদনযোগ্য হওয়া উচিত। বছ বছর ধরে সাহিত্যে এটি সম্পর্কে লেখা হয়েছে, যার মধ্যে রয়েছে

- ডোনাল্ড নুথের ১৯৯২ সালের কম্পিউটার বিজ্ঞান প্রকাশনা—লিটেরেট প্রোগ্রামিং।

—নুথ, ডি., "লিটেরেট প্রোগ্রামিং" (পিডিএফ), দ্য কম্পিউটার জার্নাল, ব্রিটিশ কম্পিউটার সোসাইটি, ১৯৯২।

- "LanguageAgnostic Reproducible Data Analysis Using Literate Programming"
প্রবন্ধটি বলে, "Lir (পিসিপিটি, পুনরুৎপাদনযোগ্য কম্পিউটিং) ডোনাল্ড নুথের প্রস্তাবিত সাক্ষর প্রোগ্রামিংয়ের ধারণার উপর ভিত্তি করে তৈরি।"

২ <http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0164023>

মূলত, প্রজননযোগ্যতা ফলাফল তৈরিতে ব্যবহৃত সম্পূর্ণ পরিবেশকে ধারণ করে — হার্ডওয়্যার, সংস্থানের যোগাযোগ, অ্যালগরিদম (বিশেষ করে কোড), ডেটা এবং ডেটার

ডকার

অধ্যায় ১৬-তে, আমরা ডকার ব্যবহার করব—একটি টুল যা সফ্টওয়্যারকে পাত্রে প্যাকেজ করার জন্য তৈরি করে যা বাস্তিল করে সেই সফ্টওয়্যারটি সুবিধাজনকভাবে, পুনরুৎপাদনযোগ্যভাবে এবং বহনযোগ্যভাবে চালানোর জন্য প্রয়োজনীয় সবকিছু প্লাটফর্ম। হ্যাপটার ১৬-তে আমরা যে কিছু সফটওয়্যার প্যাকেজ ব্যবহার করি তার জন্য জিটল সেটআপ প্রয়োজন এবং কনফিগারেশন। এর মধ্যে অনেকের জন্য, আপনি বিনামূলে আগে থেকে বিদ্যমান ডকার কটেইনার ডাউনলোড করতে পারবেন। এর ফলে আপনি জিটল ইনস্টলেশন সমস্যা এড়াতে পারবেন এবং আপনার ডিভাইসে স্থানীয়ভাবে সফ্টওয়্যার চালাতে পারবেন। ডেক্সটপ বা নোটবুক কম্পিউটার, ডকারকে নতুন কিছু শুরু করতে সাহায্য করার জন্য একটি দুর্দান্ত উপায় করে তালে দ্রুত এবং সুবিধাজনকভাবে প্রযুক্তি।

ডকার পুনরুৎপাদনযোগ্যতা বৃদ্ধিতেও সাহায্য করে। আপনি কাস্টম ডকার কটেইনার তৈরি করতে পারেন যা আপনার গবেষণায় ব্যবহৃত প্রতিটি সফ্টওয়্যার এবং প্রতিটি লাইব্রেরির সংস্করণের সাথে কনফিগার করা হবে। এটি অন্যান্য ডেভেলপারদের আপনার ব্যবহৃত পরিবেশ পুনরায় তৈরি করতে, তারপর আপনার কাজ পুনরুৎপাদন করতে এবং আপনার নিজস্ব ফ্লাফল পুনরুৎপাদন করতে সহায়তা করবে। অধ্যায় 16-এ, আপনি ডকার ব্যবহার করে একটি কটেইনার ডাউনলোড এবং কার্যকর করবেন যা আপনার জন্য জুপিটার নোটবুক ব্যবহার করে বিগ ডেটা স্পার্ক অ্যাপ্লিকেশন কোড এবং চালানোর জন্য পূর্বে কনফিগার করা আছে।

বিশেষ বৈশিষ্ট্য: আইবিএম ওয়াটসন বিশ্লেষণ এবং জ্ঞানীয় কম্পিউটিং

এই বইটির জন্য আমাদের গবেষণার শুরুতে, আমরা IBM-এর ওয়াটসনের প্রতি দ্রুত বর্ধনশীল আগ্রহকে স্থিরুত্তি দিয়েছিলাম। আমরা প্রতিযোগিতামূলক পরিষেবাগুলি অনুসন্ধান করে দেখেছি যে ওয়াটসনের "কোনও ক্রেডিট কার্ডের প্রয়োজন নেই" মৌলিক এর "মুক্ত স্তর"-এর জন্য আমাদের পাঠকদের জন্য সবচেয়ে বক্তৃতপূর্ণ।

আইবিএম ওয়াটসন একটি জ্ঞানীয় কম্পিউটিং প্ল্যাটফর্ম যা বাস্তব জগতের বিভিন্ন পারিস্থিতিতে ব্যবহৃত হয়। জ্ঞানীয় কম্পিউটিং সিস্টেমগুলি মানব মন্তব্যের প্যাটার্ন স্থিরুত্তি এবং সিদ্ধান্ত গ্রহণের ক্ষমতাগুলিকে অনুকরণ করে যাতে তারা আরও বেশি ডেটা ব্যবহার করে "শিখতে" পারে। আমরা একটি উরেখযোগ্য হ্যান্ডসন ওয়াটসন ট্রিটমেন্ট অনুরূপ করি। আমরা বিনামূলে ওয়াটসন ডেভেলপার ক্লাউড ব্যবহার করি: পাইথন এসডিকে, যা এমন API প্রদান করে যা আপনাকে ওয়াটসনের পরিষেবাগুলির সাথে প্রোগ্রাম্যাটিকভাবে ইন্টারঅ্যাক্ষেন্ট করতে সক্ষম করে। ওয়াটসন ব্যবহার করা মজাদার এবং আপনার সুজনশীল রস প্রবাহিত করার জন্য একটি দুর্দান্ত প্ল্যাটফর্ম। আপনি নিম্নলিখিত ওয়াটসন API গুলি ডেবো বা ব্যবহার করতে পারবেন:

কথোপকথন, আবিষ্কার, ভাষা অনুবাদক, প্রাকৃতিক ভাষা শেণিবন্ধকারী, প্রাকৃতিক ভাষা বোঝাপড়া, ব্যক্তিস্বত্ত্ব অন্তর্দৃষ্টি, স্পিচ টু টেক্সট, টেক্সট টু স্পিচ, টোন অ্যানালাইজার এবং ভিজুয়াল রিকগনিশন।

^১ <http://whatis.techtarget.com/definition/cognitivecomputing>.

^২ https://en.wikipedia.org/wiki/Cognitive_computing.

^৩ <https://www.forbes.com/sites/bernardmarr/2016/03/23/whateveryone-actually-thinks-about-cognitive-computing>

জ্ঞানীয় কম্পিউটিং সম্পর্কে জানতে হবে।

ওয়াটসনের লাইট টিয়ার সার্ভিসেস এবং একটি দুর্দান্ত ওয়াটসন কেস স্টাডি

আইবিএম তার অনেকের জন্য বিনামূলে লাইট টিয়ার প্রদান করে শেখা এবং পরীক্ষা-নিরীক্ষাকে উৎসাহিত করে API গুলি। পর্ব ১০-এ, আপনি অনেক ওয়াটসন পরিষেবার ডেমো চেষ্টা করবেন। তারপর, আপনি "অর্থনীতি অনুবাদ অ্যাপ বাস্তবায়নের জন্য ওয়াটসনের টেক্সট টু স্পিচ, স্পিচ টু টেক্সট এবং ট্রাঙ্কলেট পরিষেবার লাইট স্তরগুলি ব্যবহার করবেন। আপনি ইংরেজিতে একটি প্রশ্ন বলবেন, তারপর আপনার বক্তৃতা ইংরেজি টেক্সটে ট্রাঙ্কলেট প্রক্রিয়া করবে, টেক্সটটি স্প্যানিশ ভাষায় অনুবাদ করবে এবং স্প্যানিশ টেক্সটটি বলবে। এরপর, আপনি একটি স্প্যানিশ উত্তর বলবেন (যদি আপনি স্প্যানিশ না বলতে পারেন, আমরা একটি অডিও ফাইল সরবরাহ করব যা আপনি ব্যবহার করতে পারেন)। তারপর, আপনাটি দ্রুত বক্তৃতাটি স্প্যানিশ টেক্সটে ট্রাঙ্কলেট প্রক্রিয়া করবে, টেক্সটটি ইংরেজিতে অনুবাদ করবে এবং ইংরেজি উত্তরটি বলবে। দারমন জিনিস!

^৪ সর্বদা IBM-এর ওয়েবসাইটে সর্বশেষ শর্তাবলী পরীক্ষা করুন, কারণ শর্তাবলী এবং পরিষেবাগুলি পরিবর্তিত হতে পারে।

^৫ <https://console.bluemix.net/catalog/>.

শিক্ষাদান পদ্ধতি

প্রোগ্রামারদের জন্য পাইথনে বিভিন্ন ক্ষেত্র থেকে সংগৃহীত উদাহরণের একটি সমৃদ্ধ সংগ্রহ রয়েছে।

আপনি বাস্তব জগতের ডোকেস্ট ব্যবহার করে আকর্ষণীয়, বাস্তব জগতের উদাহরণগুলির মাধ্যমে কাজ করবেন। বইটি ভালো সফটওয়্যার ইঞ্জিনিয়ারিংয়ের নীতিগুলির উপর মনোনিবেশ করে এবং www.EBooksWorld.ir প্রোগ্রামের উপর জোর দেয়।

জোর দেওয়ার জন্য ফটো ব্যবহার করা

আমরা সহজ রেফারেন্সের জন্য প্রতিটি সংজ্ঞায়িত ঘটনার জন্য মূল পদ এবং সূচকের পৃষ্ঠার রেফারেন্স **বোল্ড** চেক্সে রাখি।

আমরা বোল্ড হলডেটিকা ফটো (উদাহরণস্বরূপ, ফাইল মেনু) অনস্ক্রিন উপাদানগুলি উল্লেখ করি এবং পাইথন কোডের জন্য লুসিডা ফটো ব্যবহার করি (উদাহরণস্বরূপ, $x = 5$)।

সিনট্যাক্স রঙ

পঠনযোগ্যতার জন্য, আমরা সমস্ত কোড সিনট্যাক্স রঙ করি। আমাদের সিনট্যাক্স রঙ করার নিয়মগুলি নিম্নরূপ:

মন্তব্যগুলি সবুজ রঙে প্রদর্শিত হবে কীওয়ার্ডগুলি গাঢ়
নীল ধ্রুবকগুলিতে প্রদর্শিত হবে এবং আক্ষরিক মানগুলি
হালকা নীল রঙে প্রদর্শিত হবে ক্রটিগুলি লাল রঙে প্রদর্শিত হবে অন্যান্য সমস্ত কোড কালো রঙে প্রদর্শিত হবে

538 কোড উদাহরণ

বইটির ৫০৮টি উদাহরণে প্রায় ৪০০০ লাইন কোড রয়েছে। এই আকারের বইয়ের জন্য এটি তুলনামূলকভাবে কম সংখ্যা এবং এর কারণ হল পাইথন এতটাই অভিব্যক্তিপূর্ণ ভাষা। এছাড়াও, আমাদের কোডিং স্টাইল হল যেখানেই সম্ভব বেশিরভাগ কাজ করার জন্য শক্তিশালী ক্লাস লাইব্রেরি ব্যবহার করা।

১৬০টি টেবিল/চিত্র/ভিজ্যুয়ালাইজেশন

আমরা প্রচুর সারণী, লাইন অঙ্কন এবং ছবি, গতিশীল এবং ইন্টারেক্টিভ ভিজ্যুয়ালাইজেশন অন্তর্ভুক্ত করি।

প্রোগ্রামিং প্রজ্ঞা

আমরা লেখকদের নয় দশকের প্রোগ্রামিং এবং শিক্ষাদানের অভিজ্ঞতা থেকে প্রাপ্ত প্রোগ্রামিং জ্ঞানের আলোচনায় একীভূত করি, যার মধ্যে রয়েছে:

- ভালো প্রোগ্রামিং অনুশীলন এবং পছন্দের পাইথন বাগধারা যা আপনাকে আরও স্পষ্ট, আরও বোধগম্য এবং আরও রক্ষণাবেক্ষণযোগ্য প্রোগ্রাম তৈরি করতে সাহায্য করে।
- সাধারণ প্রোগ্রামিং ক্রটিগুলি যা আপনার হওয়ার সন্তান কমাতে পারে।
- ক্রটি প্রতিরোধের টিপস, আপনার প্রোগ্রাম থেকে বাগগুলি সনাক্ত এবং অপসারণের পরামর্শ সহ। এই টিপসগুলির মধ্যে অনেকগুলি প্রথমেই আপনার প্রোগ্রামগুলিতে বাগগুলি প্রবেশ করা থেকে বিরত রাখার কৌশল বর্ণনা করে।
- পারফরম্যান্স টিপস যা আপনার প্রোগ্রামগুলিকে দ্রুত চালানোর সুযোগগুলি তুলে ধরে অথবা তাদের মেমরির পরিমাণ কমিয়ে আনে।
- সফটওয়্যার ইঞ্জিনিয়ারিং পর্যবেক্ষণ যা সঠিক সফটওয়্যার নির্মাণের জন্য, বিশেষ করে বৃহত্তর সিস্টেমের জন্য, স্বাপত্য এবং নকশার সমস্যাগুলি তুলে ধরে।

বইটিতে ব্যবহৃত সক্ষওয়্যার

আমরা যে সফটওয়্যারটি ব্যবহার করি তা উইন্ডোজ, ম্যাকওএস এবং লিনাক্সের জন্য উপলব্ধ এবং ইন্টারনেট থেকে বিনামূল্যে ডাউনলোড করা যায়। আমরা বিনামূল্যে অ্যানাকোডা পাইথন বিতরণ ব্যবহার করে বইটির উদাহরণগুলি লিখেছি। এতে আপনার প্রয়োজনীয় বেশিরভাগ পাইথন, ভিজ্যুয়ালাইজেশন এবং ডেটা সায়েন্স লাইব্রেরি রয়েছে, সেইসাথে আইপিথন ইন্টারপ্রেটার, জুপিটার নোটবুকস এবং স্পাইডার, যা অন্যতম।

সেরা পাইথন ডেটা সায়েন্স আইডিই। আমরা প্রোগ্রামের জন্য শুধুমাত্র আইপিথন এবং জুপিটার নোটবুক ব্যবহার করি

বইটিতে উল্লেখ। এই ভূমিকার পরের "বিফোর ইউ বিগিন" অংশে আলোচনা করা হয়েছে

আমাদের উদাহরণগুলি ব্যবহার করার জন্য আপনার প্রয়োজন হবে Anaconda এবং আরও কিছু জিনিস ইনস্টল করা।

পাইথন ডকুমেন্টেশন

বইটি পড়ার সময় আপনি নিষ্পত্তিগ্রস্ত ডকুমেন্টেশনগুলি বিশেষভাবে সহায়ক পাবেন:

- পাইথন ভাষার রেফারেন্স:

<https://docs.python.org/3/reference/index.html>

- পাইথন স্ট্যান্ডার্ড লাইব্রেরি:

<https://docs.python.org/3/library/index.html>

- পাইথন ডকুমেন্টেশন তালিকা:

<https://ctcpi.es:/ডক্স.পাইথন.অর্গ/৩/>

আপনার প্রশ্নের উত্তর পাওয়া

জনপ্রিয় পাইথন এবং সাধারণ প্রোগ্রামিং অনলাইন ফোরামগুলির মধ্যে রয়েছে:

- পাইথনফোরাম.অর্জও

<https://www.dreamincode.net/forums/forum/29python/>

- স্ট্যাকওভারফ্লো.কম

এছাড়াও, অনেক বিক্রেতা তাদের সরঞ্জাম এবং লাইব্রেরির জন্য ফোরাম সরবরাহ করে। এই বইটিতে আপনি যে লাইব্রেরিগুলি ব্যবহার করবেন তার অনেকগুলি github.com এ পরিচালিত এবং রক্ষণাবেক্ষণ করা হয়। কিছু লাইব্রেরি রক্ষণাবেক্ষণকারী একটি নির্দিষ্ট লাইব্রেরির GitHub পৃষ্ঠায় সমস্যা ট্যাবের মাধ্যমে সহায়তা প্রদান করে। যদি আপনি অনলাইনে আপনার প্রশ্নের উত্তর খুঁজে না পান, তাহলে অনুগ্রহ করে বইটির জন্য আমাদের ওয়েব পৃষ্ঠাটি দেখুন।

<http://www.deitel.com>

আমাদের ওয়েবসাইটটি একটি বড় আপগ্রেডের মধ্য দিয়ে যাচ্ছে। যদি আপনি আপনার প্রয়োজনীয় কিছু খুঁজে না পান, তাহলে অনুগ্রহ করে আমাদের সরাসরি eitel@deitel.com এ লিখুন।

জুপিটারের সাহায্য পাওয়া

জুপিটার নোটবুকস সাপোর্ট নিষ্পত্তিক্ষেত্রে মাধ্যমে প্রদান করা হয়:

- প্রজেক্ট জুপিটার গুগল ফ্রেম:

<https://groups.google.com/forum/#!forum/jupyter>

- জুপিটার রিয়েল-টাইম চ্যাট রুম:

<https://gitter.im/jupyter/jupyter>

- গিটহাব

<https://github.com/jupyter/help>

- স্ট্যাকওভারফ্লো:

<https://stackoverflow.com/questions/tagged/jupyter>

- জুপিটার ফর এডুকেশন গুগল গ্রুপ (জুপিটারের সাথে শিক্ষকতা করা প্রশিক্ষকদের জন্য):

<https://groups.google.com/forum/#!forum/jupytereducation>

সম্পূরক

উপস্থাপনা থেকে সর্বাধিক সুবিধা পেতে, আপনার প্রতিটি কোড উদাহরণ বইয়ের সংশ্লিষ্ট আলোচনা পড়ার সাথে সাথে সম্পাদন করা উচিত। বইয়ের ওয়েব পৃষ্ঠায়

<http://www.deitel.com>

আমরা প্রদান করি:

- বইয়ের কোড উদাহরণের জন্য ডাউনলোডযোগ্য পাইথন সোর্স কোড (.py ফাইল) এবং জুপিটার নোটবুক (.ipynb ফাইল)।

- গুরুত্বপূর্ণ কোড উদাহরণগুলি কীভাবে ব্যবহার করতে হয় তা দেখায়। আমরা এই সরঞ্জামগুলি অধ্যায় 1.5 তেও পরিচয় করিয়ে দিচ্ছি।

www.EBooksWorld.ir

- ব্লগ পোস্ট এবং বইয়ের আপডেট।

ডাউনলোডের নির্দেশাবলীর জন্য, এই ভূমিকার পরে "শুরু করার আগে" বিভাগটি দেখুন।

লেখকদের সাথে যোগাযোগ রাখা

প্রশ্নের উত্তর পেতে অথবা ত্রুটি জানাতে, আমাদের ইমেল পাঠান

eitel@deitel.com সম্পর্কে

অথবা সোশ্যাল মিডিয়ার মাধ্যমে আমাদের সাথে যোগাযোগ করুন:

- ফেসবুক (<http://www.deitel.com/deitelfan>)
- টুইটার (@deitel)
- লিঙ্কডিন ([®] <https://linkedin.com/company/deitel&associates>)
- ইউটিউব (<http://youtube.com/DeitelTV>)

স্বীকৃতি

এই প্রকল্পের উপর ইন্টারনেট গবেষণার জন্য দীর্ঘ সময় ব্যয় করার জন্য আমরা বারবার ডেইটেলকে ধন্যবাদ জানাতে চাই।
আমরা তাগ্যবান যে পিয়ারসনের প্রকাশনা পেশাদারদের বিবেদিতপ্রাণ দলের সাথে কাজ করতে পেরেছি। আমাদের বক্তৃ এবং
সহকর্মী পিয়ারসন আইটি প্রফেশনাল ফ্রেন্ডের ভাইস প্রেসিডেন্ট মার্ক এল. টাউবের প্রচেষ্টা এবং ২৫ বছরের পরামর্শের জন্য আমরা
কৃতজ্ঞ। মার্ক এবং তার দল সাফারি পরিষেবাতে আমাদের পেশাদার বই, লাইভলেসনের ডিজিট পণ্য এবং শেখার পথ প্রকাশ করে (

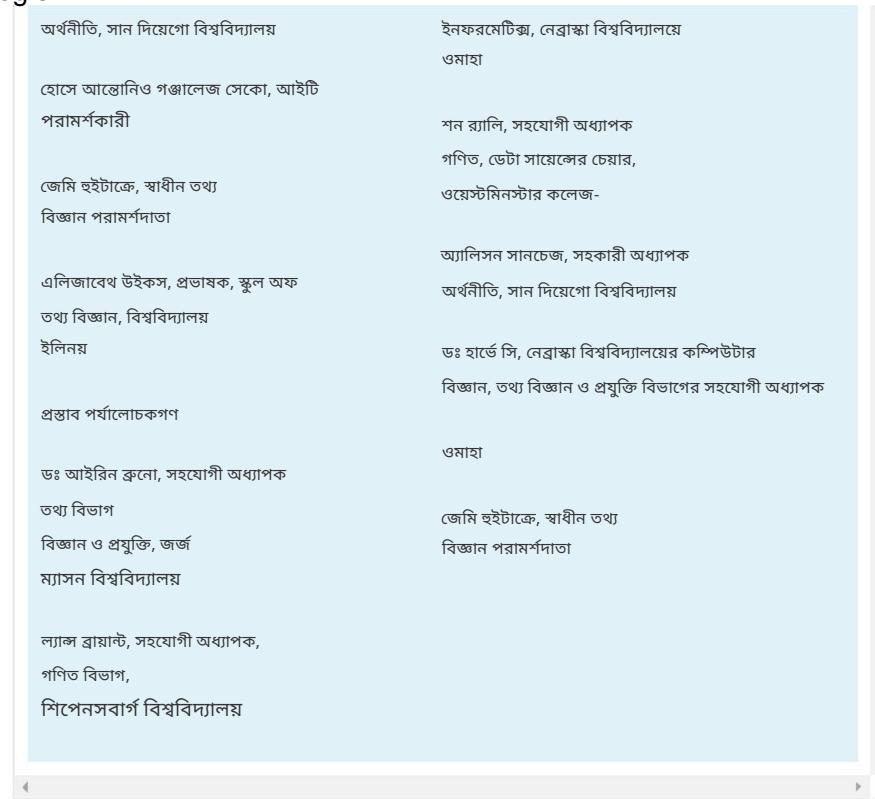
<https://learning.oreilly.com/>)। তারা আমাদের সাফারি লাইভ অনলাইন প্রশিক্ষণেরও পৃষ্ঠপোষকতা করে।

সেমিনার। জুলি নাহিল বাইটির প্রযোজনা পরিচালনা করেছিলেন। আমরা প্রচদ্র শিল্প এবং ছুটি নির্বাচন করেছি।

প্রচদ্রটি ডিজাইন করেছেন প্রসার্টসিথ।

আমরা আমাদের পর্যালোচকদের প্রচেষ্টার প্রশংসা করতে চাই। প্যাট্রিসিয়া বায়রন কিষ্টল এবং মেগান
জ্যাকবি পর্যালোচকদের নিয়ে করেছিলেন এবং পর্যালোচনা প্রক্রিয়া পরিচালনা করেছিলেন। একটি কঠোর সময়সূচী মেনে,
পর্যালোচকরা আমাদের কাজটি পরীক্ষা-নিরীক্ষা করেছিলেন, উপস্থাপনার নির্ভুলতা, সম্পূর্ণতা এবং সময়োপযোগীতা উন্নত
করার জন্য অসংখ্য পরামর্শ প্রদান করেছিলেন।

পর্যালোচকরা	
বই সমালোচক	
ড্যানিয়েল চেন, ডেটা সায়েন্টিস্ট, ল্যাভার বিশ্ববিদ্যালয়	ড্যানিয়েল চেন, ডেটা সায়েন্টিস্ট, ল্যাভার বিশ্ববিদ্যালয়
গ্যারেট ড্যানসিক, সহযোগী অধ্যাপক কম্পিউটার বিজ্ঞান/জৈব তথ্যবিদ্যা, ইন্সটার্ন কানেকটিকাট স্টেট ইউনিভার্সিটি	গ্যারেট ড্যানসিক, সহযোগী অধ্যাপক কম্পিউটার বিজ্ঞান/জৈব তথ্যবিদ্যা, ইন্সটার্ন কানেকটিকাট স্টেট ইউনিভার্সিটি
প্রাণশু গুপ্ত, সহকারী অধ্যাপক, কম্পিউটার বিজ্ঞান, ডিসেলস বিশ্ববিদ্যালয়	প্রাণশু গুপ্ত, সহকারী অধ্যাপক কম্পিউটার বিজ্ঞান, ডিসেলস বিশ্ববিদ্যালয়
ডেভিড কুপ, সহকারী অধ্যাপক, ডেটা বিজ্ঞান প্রোগ্রামের সহ-পরিচালক, ইউমাস ডার্টমাউথ	ডেভিড কুপ, সহকারী অধ্যাপক কম্পিউটার বিজ্ঞান, পূর্ব কানেকটিকাট স্টেট ইউনিভার্সিটি
র্যামন মাতাটোলেতো, অধ্যাপক, কম্পিউটার বিজ্ঞান, জেমস ম্যাডিসন বিশ্ববিদ্যালয়	র্যামন মাতাটোলেতো, অধ্যাপক কম্পিউটার বিজ্ঞান, পূর্ব কানেকটিকাট স্টেট ইউনিভার্সিটি
Shyamal Mitra, Senior Lecturer, কম্পিউটার বিজ্ঞান, টেক্সাস বিশ্ববিদ্যালয় অস্টিন	Shyamal Mitra, Senior Lecturer, কম্পিউটার বিজ্ঞান, টেক্সাস বিশ্ববিদ্যালয় অস্টিন
অ্যালিসন সানচেজ, সহকারী অধ্যাপক	ডঃ মার্ক পাউলি, সিনিয়র রিসার্চ ফেলো, বায়োইনফরমেটিক্স, মার্কিন যুক্তরাষ্ট্রিয়নারি



বইটি পড়ার সময়, আপনার মন্তব্য, সমালোচনা, সংশোধন এবং উন্নতির জন্য পরামর্শের জন্য আমরা কৃতজ্ঞ থাকব

অনুগ্রহ করে সমস্ত চিঠিপত্র পাঠান

eitel@deitel.com সম্পর্কে

আমরা তৎক্ষণিকভাবে সাড়া দেব।
পাইথন প্রোগ্রামিংয়ের উত্তেজনাপূর্ণ ওপেনসোর্স জগতে আবারও স্বাগতম। আমরা আশা করি আপনি পাইথন, আইপিথন, জুপিটার নোটবুকস, ডেটা সায়েন্স, এআই, বিগ ডেটা এবং ক্লাউড সহ শীর্ষস্থানীয় কম্পিউটার অ্যাপ্লিকেশন ডেভেলপমেন্টের এই দৃশ্যটি উপভোগ করবেন।

ডিটেল অ্যান্ড অ্যাসোসিয়েটস, ইনকর্পোরেটেডের সিইও এবং চিফ টেকনিক্যাল অফিসার পল জে. ডিটেল একজন এমআইটি স্নাতক এবং কম্পিউটিংয়ে ৩৮ বছরের অভিজ্ঞতা সম্পন্ন। পল বিশ্বের অন্যতম অভিজ্ঞ প্রোগ্রামিং ল্যাসুয়োজ প্রশিক্ষক, ১৯৯২ সাল থেকে সফটওয়্যার ডেভেলপারদের পেশাদার কোর্স পঢ়াচ্ছেন। তিনি আন্তর্জাতিকভাবে শিখ ক্লায়েন্টদের কাছে শত শত প্রোগ্রামিং কোর্স সরবরাহ করেছেন, যার মধ্যে রয়েছে সিসকো, আইবিএম, সিমেল, সান মাইক্রোসিস্টেমস (বর্তমানে ওরাকল), ডেল ফিল্ডেলিটি, কেমেডি স্পেস সেন্টারে নাসা, ম্যাশনাল সিডিয়ার স্টর্ম ল্যাবরেটরি, হোয়াইট স্যান্ডস মিসাইল রেঞ্জ, রোগ ওয়েভ সফটওয়্যার, বেহিং, নরটেল নেটওয়ার্কিংস, পুমা, আইরোবাট এবং আরও অনেক কিছু। তিনি এবং তার সহ-লেখক, ড. হার্টে এম. পিটেল বিশ্বের প্রাচীনতম তিনিশ্ব।

প্রোফেসর মাসুদ পার্সেক / প্রোফেসর রফি ফিল্ডিংও লেখক

ডিটেল অ্যান্ড অ্যাসোসিয়েটস, ইনকর্পোরেটেডের চেয়ারম্যান এবং প্রধান কৌশল কর্মকর্তা ডঃ এ.এস. হার্ডে এম. ডিটেল, কম্পিউটিংয়ে ৮ে বছরের অভিজ্ঞতা। ডঃ ডিটেল ইলেক্ট্রিক্যালে বিএস এবং এমএস ডিপ্লি অর্জন করেছেন। এমআইটি থেকে ইঞ্জিনিয়ারিং এবং বোস্টন বিশ্ববিদ্যালয় থেকে গণিতে পিইএইচডি করেছেন—তিনি পড়াশোনা করেছেন কম্পিউটার বিজ্ঞান প্রোগ্রামগুলি তৈরি করার আগে এই প্রোগ্রামগুলির প্রতিটিতে কম্পিউটিং। তিনি ১৯৯১ সালে পুর পলের সাথে ডিটেল অ্যান্ড অ্যাসোসিয়েটস, ইনকর্পোরেটেড প্রতিষ্ঠার আগে বোস্টন কলেজে কম্পিউটার বিজ্ঞান বিভাগের চেয়ারম্যান হিসেবে দায়িত্ব পালন এবং মেয়াদ অর্জন সহ কলেজ শিক্ষকতার বিস্তৃত অভিজ্ঞতা রয়েছে। ডিটেলসের প্রকাশনাগুলি আন্তর্জাতিক সীরিজে অর্জন করেছে, জাপানি, জার্মান, রাশিয়ান, স্প্যানিশ, ফরাসি, পোলিশ, ইতালীয়, সল্লীকৃত চীনা, এন্টিয়াবাহী চীনা, কোরিয়ান, পার্শ্বজি, শীরী, উর্দু এবং তুর্কি ভাষার ১০০ টিরও বেশি অনুবাদ প্রকাশিত হয়েছে। ডঃ ডিটেল একাডেমিক কার্যাবাহী সরকারী এবং সামরিক ক্ষেত্রে বিশেষ প্রশংসনীয় কর্মসূল সরবরাত্ব করেছেন।

পল ডেইটেল এবং হার্টে ডেইটেল দ্বারা প্রতিষ্ঠিত eitel & Associates, Inc., একটি আন্তর্জাতিকভাবে

স্বীকৃত লেখক এবং কর্পোরেট প্রশিক্ষণ সংস্থা, কম্পিউটারে বিশেষজ্ঞ

প্রোগ্রামিং ভাষা, অবজেক্ট প্রযুক্তি, মোবাইল অ্যাপ ডেভেলপমেন্ট এবং ইন্টারনেট এবং ওয়েব

সফটওয়্যার প্রযুক্তি। কোম্পানির প্রশিক্ষণ ক্লায়েন্টদের মধ্যে বিশ্বের বৃহত্তম কিছু অন্তর্ভুক্ত রয়েছে

কোম্পানি, সরকারি সংস্থা, সামরিক শাখা এবং শিক্ষা প্রতিষ্ঠান।

কোম্পানি বিশ্বব্যাপী ক্লায়েন্ট সাইটগুলিতে প্রধান বিষয়গুলিতে প্রশিক্ষক-ভিত্তিক প্রশিক্ষণ কোর্স অফার করে

প্রোগ্রামিং ভাষা।

পিয়ারসন/প্রেসিস হলের সাথে ৪৪ বছরের প্রকাশনা অংশীদারিত্বের মাধ্যমে, ডিটেল অ্যান্ড অ্যাসোসিয়েটস, ইনকর্পোরেটেড, প্রিন্ট এবং ই-বুক ফর্ম্যাটে

শীর্ষস্থানীয় প্রোগ্রামিং পাঠ্যপুস্তক এবং পেশাদার বই, লাইভলেসনস ভিডিও কোর্স (<https://www.informit.com> এ কেনার জন্য উপলব্ধ), লার্নিং

পাথস এবং লাইভ অনলাইন প্রশিক্ষণ সেমিনার প্রকাশ করে।

সাফারি পরিবেৰা (<https://learning.oreilly.com>) এবং Revel™ ইন্টারেক্টিভ মাল্টিমিডিয়া

কোর্স।

ডিটেল অ্যান্ড অ্যাসোসিয়েটস, ইনকর্পোরেটেড এবং লেখকদের সাথে যোগাযোগ করতে, অথবা অনসাইট, প্রশিক্ষক প্রশিক্ষণের
প্রস্তাবের জন্য অনুরোধ করতে, লিখুন:

eitel@deitel.com সম্পর্কে

ডিটেল অনসাইট কর্পোরেট প্রশিক্ষণ সম্পর্কে আরও জানতে, ভিজিট করুন

<http://www.deitel.com/training>

ডিটেল বই কিনতে ইচ্ছুক ব্যক্তিরা এখানে তা করতে পারেন

<http://www.amazon.com>

কর্পোরেশন, সরকার, সামরিক বাহিনী এবং শিক্ষা প্রতিষ্ঠানের বাস্ক অর্ডারের ক্ষেত্রে

পিয়ারসনের সাথে সরাসরি নিয়োগ পেতে পারেন। আরও তথ্যের জন্য, দেখুন

<http://www.informit.com/store/sales.aspx>

গুণক করার আগে

এই বিভাগে এমন তথ্য রয়েছে যা এই বইটি ব্যবহারের আগে আপনার পর্যালোচনা করা উচিত। আমরা <http://www.deitel.com> এ অপিকস আপডেট করব।

উপার্জনের পথ

ফন্ট এবং নামকরণ কনভেনশন

অফার্স এবং ডিল

আমরা পাইথন কোড এবং কমান্ড এবং ফাইল এবং ফোল্ডারের নামগুলি একটি sansserif ফন্টে এবং অনঙ্কিন উপাদানগুলি, যেমন মেনু নামগুলি, একটি গাঢ় sansserif ফন্টে দেখাই। highlights

আমরা জোর দেওয়ার জন্য ইটালিক এবং জোর দেওয়ার জন্য মাঝে মাঝে বোল্ড ব্যবহার করি।

এটিং

কোডের উদাহরণ পাওয়া

সমর্থন

আপনি আমাদের প্রোগ্রামারদের জন্য সাইন আউট পাইথন ওয়েব পৃষ্ঠা থেকে বইয়ের উদাহরণ সম্পর্কিত examples.zip ফাইলটি ডাউনলোড করতে পারেন :

<http://www.deitel.com>

আপনার স্থানীয় কম্পিউটারে ফাইলটি সংরক্ষণ করতে ডাউনলোড উদাহরণ লিঙ্কে ক্লিক করুন। বেশিরভাগ ওয়েব ব্রাউজার আপনার ব্যবহারকারী অ্যাকাউন্টের ডাউনলোড ফোল্ডারে ফাইলটি রাখে। ডাউনলোড সম্পূর্ণ হলে, আপনার সিস্টেমে এটি সনাক্ত করুন এবং এর উদাহরণ ফোল্ডারটি আপনার ব্যবহারকারী অ্যাকাউন্টের ডকুমেন্টস ফোল্ডারে এক্সট্রাক্ট করুন:

- উইন্ডোজ: সি:\ব্যবহারকারী\আপনার অ্যাকাউন্ট\ডকুমেন্টস\উদাহরণ
- ম্যাকওএস বা লিনাক্স: ~/ডকুমেন্টস/উদাহরণ

বেশিরভাগ অপারেটিং সিস্টেমে একটি বিল্টইন এক্সট্রাকশন টুল থাকে। আপনি 7Zip (www.7zip.org) অথবা WinZip (www.winzip.com) এর মতো একটি আর্কাইভ টুলও ব্যবহার করতে পারেন।

উদাহরণ ফোল্ডারের গঠন

এই বইটিতে আপনি তিন ধরণের উদাহরণ ব্যবহার করতে পারবেন:

- আইপিথন ইন্টারেক্ষিভ পরিবেশ পৃথক কোড স্লিপেট।
- সম্পূর্ণ অ্যাপ্লিকেশন, যা স্ক্রিপ্ট নামে পরিচিত।
- জুপিটার নোটবুকস—একটি সুবিধাজনক ইন্টারেক্ষিভ, ওয়েব ব্রাউজার ভিত্তিক পরিবেশ যেখানে আপনি কোড লিখতে এবং কার্যকর করতে পারেন এবং কোডটিকে টেক্সট, ছবি এবং ভিডিওর সাথে মিশ্রিত করতে পারেন।

আমরা একশন ১.৫ এর টেস্ট ড্রাইভের প্রতিটি প্রদর্শন করি।

উদাহরণ ফোল্ডারে প্রতি অধ্যায়ে একটি করে সাবফোল্ডার থাকে। এগুলোর নাম ch##, যেখানে ## হল দুই অক্ষের অধ্যায় সংখ্যা 01 থেকে 16—উদাহরণস্বরূপ, ch01।

অধ্যায় ১৩, ৫ এবং ৬, প্রতিটি অধ্যায়ের ফোল্ডারে নিম্নলিখিত আইটেমগুলি রয়েছে:

- snippets_ipynb—অধ্যায়ের Jupyter Notebook ফাইল ধারণকারী একটি ফোল্ডার।
- snippets_py—পাইথন সোর্স কোড ফাইল ধারণকারী একটি ফোল্ডার যেখানে আমরা যে কোড স্লিপেট উপস্থাপন করি তা একটি ফাঁকা লাইন দ্বারা পরেরটি থেকে আলাদা করা হয়। আপনি এই স্লিপেটগুলি IPython অথবা আপনার তৈরি করা নতুন Jupyter Notebooks-এ কপি করে পেস্ট করতে পারেন।
- স্ক্রিপ্ট ফাইল এবং তাদের সহায়ক ফাইল।

হ্যাপটার ১৩ তে একটি অ্যাপ্লিকেশন রয়েছে। হ্যাপটার ১৫ এবং ৬ ব্যাখ্যা করে যে ফাইলগুলি কোথায় পাওয়া যাবে। আপনার যথাক্রমে ch15 এবং ch16 ফোল্ডারে প্রয়োজন।

অ্যানাকোন্ডা ইনস্টল করা হচ্ছে

এই বইয়ের সাথে আমরা easytoinstall Anaconda Python ডিস্ট্রিবিউশন ব্যবহার করি। আমাদের উদাহরণগুলির সাথে কাজ করার জন্য আপনার যা যা প্রয়োজন হবে তার প্রায় সবকিছুই এতে রয়েছে, যার মধ্যে রয়েছে:

- আইপিথন ইন্টারপ্রেটার,
- আমরা যে পাইথন এবং ডেটা সায়েন্স লাইব্রেরি ব্যবহার করি, তার বেশিরভাগই
- একটি স্থানীয় জুপিটার নোটবুক সার্ভার যাতে আপনি আমাদের নোটবুকগুলি লোড এবং কার্যকর করতে পারেন, এবং
- অন্যান্য বিভিন্ন সফ্টওয়্যার প্যাকেজ, যেমন স্পাইডার ইন্টিগ্রেটেড ডেভেলপমেন্ট এনভায়রনমেন্ট (IDE)-এই বইটিতে আমরা শুধুমাত্র IPython এবং Jupyter Notebook ব্যবহার করি।

উইন্ডোজ, ম্যাকওএস অথবা লিনাক্সের জন্য পাইথন ৩.৬-এর অ্যানাকোন্ডা ইনস্টলারটি এখান থেকে ডাউনলোড করুন:

ডাউনলোড সম্পূর্ণ হলে, ইনস্টলারটি চালান এবং অনক্ষিন নির্দেশাবলী অনুসরণ করুন।

Anaconda সঠিকভাবে কাজ করছে কিনা তা নিশ্চিত করার জন্য, এটি ইনস্টল করার পরে এর ফাইলগুলি সরান না।

অ্যানাকোভা আপডেট করা হচ্ছে

এরপর, নিশ্চিত করুন যে Anaconda আপ টু ডেট আছে। আপনার সিস্টেমে নিম্নরূপ একটি কমান্ড লাইন উইন্ডো খুলুন:

- macOS-এ, অ্যাপ্লিকেশন ফোল্ডারের ইউটিলিটিস সাবফোল্ডার থেকে একটি টার্মিনাল খুলুন।
- উইন্ডোজে, স্টার্ট মেনু থেকে Anaconda Prompt খুলুন। Anaconda আপডেট করার জন্য (যেমনটি আপনি এখানে করবেন) অথবা নতুন প্যাকেজ ইনস্টল করার জন্য (কিছুক্ষণের জন্য আলোচনা করা হয়েছে) এটি করার সময়, ডান-ক্লিক করে, তারপর More > Run as administrator নির্বাচন করে Anaconda Prompt অ্যাডমিনিস্ট্রেটর হিসেবে কার্যকর করুন। (যদি আপনি স্টার্ট মেনুতে Anaconda Prompt খুঁজে না পান, তাহলে আপনার স্ক্রিনের নীচে Type here to search ফিল্ডে এটি অনুসন্ধান করুন।)
- লিনাক্সে, আপনার সিস্টেমের টার্মিনাল বা শেল খুলুন (এটি লিনাক্স বিতরণ অনুসারে পরিবর্তিত হয়)।

আপনার সিস্টেমের কমান্ডলাইন উইন্ডোতে, আপডেট করার জন্য নিম্নলিখিত কমান্ডগুলি কার্যকর করুন অ্যানাকোভা ইনস্টল করা প্যাকেজগুলি তাদের সর্বশেষ সংস্করণে:

১. কনডা আপডেট কনডা

২. দ্বিতীয় আপডেট সব

প্যাকেজ ম্যানেজার

উপরে ব্যবহৃত conda কমান্ডটি conda প্যাকেজ ম্যানেজারকে আহ্বান করে— এই বইটিতে আপনি যে দুটি মূল Python প্যাকেজ ম্যানেজার ব্যবহার করবেন তার মধ্যে একটি। অন্যটি হল pip। প্যাকেজগুলিতে একটি নির্দিষ্ট Python লাইব্রেরি বা টুল ইনস্টল করার জন্য প্রয়োজনীয় ফাইল থাকে। পুরো বই জুড়ে, আপনি অতিরিক্ত প্যাকেজ ইনস্টল করার জন্য conda ব্যবহার করবেন, যদি না সেই প্যাকেজগুলি conda এর মাধ্যমে উপলব্ধ না হয়, তবে আপনি pip ব্যবহার করবেন। কিছু লোক একচেটিয়াভাবে pip ব্যবহার করতে পছন্দ করেন কারণ এটি বর্তমানে আরও প্যাকেজ সমর্থন করে। যদি আপনার কখনও conda দিয়ে প্যাকেজ ইনস্টল করতে সমস্যা হয়, তাহলে pip ব্যবহার করে দেখুন।

প্রসপেক্টর স্ট্যাটিক কোড ইনস্টল করা বিশ্লেষণের হাতিয়ার

আপনি প্রসপেক্টর বিশ্লেষণ টুল ব্যবহার করে আপনার পাইথন কোড বিশ্লেষণ করতে চাইতে পারেন, যা আপনার কোডে সাধারণ ত্রুটিগুলি পরীক্ষা করে এবং এটি উন্নত করতে আপনাকে সাহায্য করে। প্রসপেক্টর এবং এটি ব্যবহার করে পাইথন লাইব্রেরি ইনস্টল করতে, কমান্ড লাইনে নিম্নলিখিত কমান্ডটি চালান।

জানালা:

পিপ ইনস্টল প্রসপেক্টর

জুপিটার-ম্যাটপ্লটলিব ইনস্টল করা হচ্ছে

আমরা Matplotlib নামক একটি ভিজুয়ালাইজেশন লাইব্রেরি ব্যবহার করে বেশ কয়েকটি অ্যানিমেশন বাস্তবায়ন করি। Jupyter Notebooks-এ এগুলি ব্যবহার করতে, আপনাকে ipympl নামক একটি টুল ইনস্টল করতে হবে। টার্মিনাল, Anaconda কমান্ড প্রম্পট বা আপনার পূর্বে খোলা শেল-এ, নিম্নলিখিতটি কার্যকর করুন একের পর এক ফ্রমান্ড দেয়:

› <https://github.com/matplotlib/jupytermatplotlib>

conda ইনস্টল c condaforge ipympl

conda ইনস্টল মোডজেস জুপিটার

ল্যাবেরিটেনশন ইনস্টল @jupyterwidgets/jupyterlabmanager jupyter labextension ইনস্টল jupytermatplotlib

অন্যান্য প্যাকেজ ইনস্টল করা

Anaconda আপনার জন্য প্রায় 300 টি জনপ্রিয় Python এবং ডেটা সায়েন্স প্যাকেজ নিয়ে আসে, যেমন NumPy, Matplotlib, pandas, Regex, BeautifulSoup, requests, Bokeh, SciPy, SciKitLearn, Seaborn, Spacy, sqlite, statsmodels এবং আরও অনেক কিছু। বই জুড়ে আপনার ইনস্টল করার জন্য অতিরিক্ত প্যাকেজের সংখ্যা কম হবে এবং প্রয়োজনে আমরা ইনস্টলেশন নির্দেশাবলী প্রদান করব। আপনি যখন নতুন প্যাকেজ আবিষ্কার করবেন, তখন তাদের ডকুমেন্টেশন ব্যাখ্যা করবে কিভাবে সেগুলি ইনস্টল করতে হয়।

একটি টুইটার ডেভেলপার অ্যাকাউন্ট পান

যদি আপনি আমাদের "ডেটা মাইনিং টুইটার" অধ্যায় এবং পরবর্তী অধ্যায়গুলিতে টুইটার-ভিত্তিক কোনও উদাহরণ ব্যবহার করতে চান, তাহলে একটি টুইটার ডেভেলপার অ্যাকাউন্টের জন্য আবেদন করুন। টুইটারকে এখন তাদের API গুলিতে অ্যাক্সেসের জন্য নিবন্ধন করতে হবে। আবেদন করতে, আবেদনপত্র পূরণ করুন এবং জমা দিন

<https://developer.twitter.com/en/applyforaccess>

টুইটার প্রতিটি আবেদন পর্যালোচনা করে। এই লেখার সময়, ব্যক্তিগত ডেভেলপার অ্যাকাউন্টগুলি অবিলম্বে অনুমোদিত হচ্ছিল এবং কোম্পানির অ্যাকাউন্টের আবেদনগুলি

কয়েক দিন থেকে কয়েক সপ্তাহ পর্যন্ত সময় লাগে। অনুমোদনের নিশ্চয়তা নেই।

কিছু ক্ষেত্রে ইন্টারনেট সংযোগ প্রয়োজন অধ্যায়

এই বইটি ব্যবহার করার সময়, বিভিন্ন অতিরিক্ত পাইথন লাইব্রেরি ইনস্টল করার জন্য আপনার একটি ইন্টারনেট সংযোগের প্রয়োজন হবে। কিছু অধ্যায়ে, আপনাকে ক্লাউডভিত্তিক পরিষেবাগুলির সাথে অ্যাকাউন্টগুলির জন্য নিবন্ধন করতে হবে, বেশিরভাগ ক্ষেত্রে তাদের বিনামূল্যের স্তরগুলি ব্যবহার করার জন্য। কিছু পরিষেবার জন্য আপনার পরিচয় যাচাই করার জন্য ক্রেডিট কার্ডের প্রয়োজন হয়। কিছু ক্ষেত্রে, আপনি এমন পরিষেবা ব্যবহার করবেন যা বিনামূল্যে নয়। এই ক্ষেত্রে, আপনি বিক্রেতাদের দ্বারা প্রদত্ত আর্থিক ক্রেডিটের সুবিধা গ্রহণ করবেন যাতে আপনি কোনও চার্জ ছাড়াই তাদের পরিষেবাগুলি চেষ্টা করতে পারেন। সতর্কতা: কিছু ক্লাউডভিত্তিক পরিষেবা সেট আপ করার পরে খরচ হয়। এই ধরনের পরিষেবা ব্যবহার করে আমাদের কেস স্টাডি সম্পূর্ণ করার সময়, আপনার বরাদ্দকৃত সংস্থানগুলি অবিলম্বে মুছে ফেলতে ভুলবেন না।

প্রোগ্রাম আউটপুটগুলিতে সামান্য পার্থক্য

আমাদের উদাহরণগুলি কার্যকর করার সময়, আপনি আমাদের দেখানো ফলাফল এবং আপনার নিজের ফলাফলের মধ্যে কিছু পার্থক্য লক্ষ্য করতে পারেন:

- বিভিন্ন অপারেটিং সিস্টেমে ফ্লোটিংপয়েন্ট সংখ্যা (যেমন -১২৩.৪৫, ৭.৫ অথবা ০.০২৩৬৯৩৭) দিয়ে গণনা করার পদ্ধতিতে পার্থক্যের কারণে, আপনি আউটপুটগুলিতে সামান্য তারতম্য দেখতে পাবেন—বিশেষ করে দশমিক বিন্দুর ডানদিকের সংখ্যাগুলিতে।
- যখন আমরা আলাদা উইন্ডোতে প্রদর্শিত আউটপুটগুলি দেখাই, তখন আমরা উইন্ডোগুলির সীমানা মুছে ফেলার জন্য তাদের ক্রপ করি।

আপনার প্রশ্নের উত্তর পাওয়া

অনলাইন ফোরামগুলি আপনাকে অন্যান্য পাইথন প্রোগ্রামারদের সাথে যোগাযোগ করতে এবং আপনার পাইথন প্রশ্নের উত্তর পেতে সক্ষম করে। জনপ্রিয় পাইথন এবং সাধারণ প্রোগ্রামিং ফোরামগুলির মধ্যে রয়েছে:

- পাইথনফোরাম.আইও
- স্ট্যাকওভারফ্লো.কম
- [চিটিপিএস://www.dreamincode.net/forums/forum/29python/](https://www.dreamincode.net/forums/forum/29python/)

এছাড়াও, অনেক বিক্রেতা তাদের সরঞ্জাম এবং লাইব্রেরির জন্য ফোরাম সরবরাহ করে। এই বইটিতে আপনি যে লাইব্রেরিগুলি ব্যবহার করবেন তার বেশিরভাগই github.com এ পরিচালিত এবং রক্ষণাবেক্ষণ করা হয়। কিছু লাইব্রেরি রক্ষণাবেক্ষণকারী একটি নির্দিষ্ট লাইব্রেরির GitHub পৃষ্ঠায় সমস্যা ট্যাবের মাধ্যমে সহায়তা প্রদান করে।

যদি আপনি অনলাইনে আপনার প্রশ্নের উত্তর খুঁজে না পান, তাহলে অনুগ্রহ করে আমাদের ওয়েব পৃষ্ঠাটি দেখুন

বুক করুন

<http://www.deitel.com>

২

^২ আমাদের ওয়েবসাইটটি একটি বড় ধরণের আপগ্রেডের মধ্য দিয়ে যাচ্ছে। যদি আপনি আপনার প্রয়োজনীয় কিছু খুঁজে না পান, অনুগ্রহ করে সরাসরি আমাদের eitel@deitel.com এ লিখুন।

তুমি এখন প্রোগ্রামারদের জন্য পাইথন পড়া শুরু করতে প্রস্তুত। আশা করি বইটি তোমার ভালো লাগবে!

গল্প ১. কম্পিউটার এবং পাইথনের পরিচিতি

উদ্দেশ্যমূলক মতামত

এই অধ্যায়ে আপনি পাবেন: উপর্যুক্ত পথ

■ কম্পিউটিংয়ের সাম্প্রতিক উত্তেজনাপূর্ণ উন্নয়ন সম্পর্কে জানুন।
অফার্স এবং ডিল

■ অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং এর মূল বিষয়গুলি পর্যালোচনা করুন।

highlights

■ পাইথনের শক্তিগুলি বু�ুন।

এটিৎ

■ এই বইটিতে আপনি যে গুরুত্বপূর্ণ পাইথন এবং ডেটাসায়েন্স লাইব্রেরিগুলি ব্যবহার করবেন তার সাথে পরিচয় করিয়ে দিন।

সমর্থন

■ পাইথন কোড এক্সিকিউট করার জন্য আইপিথন ইন্টারপ্রেটারের ইন্টারেক্ষিভ মোডটি টেস্টড্রাইভ করুন।

সাইন আউট

■ একটি পাইথন স্ক্রিপ্ট চালান যা একটি বার চার্টকে অ্যানিমেট করে।

■ পাইথন কোড এক্সিকিউট করার জন্য একটি ওয়েব ব্রাউজার ভিত্তিক জুপিটার নোটবুক তৈরি এবং টেস্টড্রাইভ করুন।

■ "বিগ ডেটা" কতটা বড় এবং কত দ্রুত এটি আরও বড় হচ্ছে তা জানুন।

■ একটি জনপ্রিয় মোবাইল মেডিগেশন অ্যাপে একটি বিগডেটা কেস স্টাডি পড়ুন।

■ কম্পিউটার বিজ্ঞান এবং তথ্য বিজ্ঞানের সংযোগস্থলে অবস্থিত কৃতিম বুদ্ধিমত্তার সাথে পরিচিত হোন।

কৃপরেখা

.1 ভূমিকা

.2 অবজেক্ট টেকনোলজির মূলনীতির একটি দ্রুত পর্যালোচনা

.3 পাইথন

.8 এটা লাইব্রেরি!

.8.1 পাইথন স্ট্যান্ডার্ড লাইব্রেরি

.8.2 ডেটাসায়েন্স লাইব্রেরি

.৫ টেস্টড্রাইভ: আইপিথন এবং জুপিটার নোটবুক ব্যবহার করা

.৫.১ ক্যালকুলেটর হিসেবে আইপিথন ইন্টারেক্ষিভ মোড ব্যবহার করা

.৫.২ আইপিথন ইন্টারপ্রেটার ব্যবহার করে একটি পাইথন প্রোগ্রাম কার্যকর করা

.৫.৩ জুপিটার নোটবুকে কোড লেখা এবং কার্যকর করা

.৬ ক্লাউড এবং ইন্টারনেট অফ থিংস

.৬.১ মেঘ

.৬.২ ইন্টারনেট অফ থিংস

.৭ বিগ ডেটা কত বড়?

.৭.১ বিগ ডেটা অ্যানালিটিক্স

.৭.২ ডেটা সায়েন্স এবং বিগ ডেটা পার্থক্য আনছে: ব্যবহারের ক্ষেত্রে

.৮ কেস স্টাডি—একটি বিগডেটা মোবাইল অ্যাপ্লিকেশন

.৯ ডেটা সায়েন্সের ভূমিকা: কৃত্রিম বুদ্ধিমত্তা—সিএস এবং ডেটা সায়েন্সের সংযোগস্থলে

.১০ সারসংক্ষেপ

১.১ ভূমিকা

পাইথনে আপনাকে স্বাগতম—বিশ্বের সবচেয়ে বেশি ব্যবহৃত কম্পিউটার প্রোগ্রামিং ভাষাগুলির মধ্যে একটি এবং জনপ্রিয়তা প্রোগ্রামিং ভাষার (PYPL) সূচক অনুসারে, বিশ্বের সবচেয়ে জনপ্রিয়।

১

^১ <https://pypl.github.io/PYPL.html> (জানুয়ারী ২০১৯ অনুযায়ী)।

এখানে, আমরা হ্যাপ্টার্স ২-০ - এ আপনি যে পাইথন প্রোগ্রামিং শিখবেন তার ভিত্তি স্থাপনকারী পরিভাষা এবং ধারণাগুলি এবং হ্যাপ্টার্স ১১-৬ - এ আমরা যে বিগডেটা, কৃত্রিম বুদ্ধিমত্তা এবং ক্লাউড-ভিত্তিক কেস স্টাডি উপস্থাপন করব তার সাথে পরিচয় করিয়ে দেব।

আমরা অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং পরিভাষা এবং ধারণাগুলি পর্যালোচনা করব। আপনি শিখবেন কেন পাইথন এত জনপ্রিয় হয়ে উঠেছে। আমরা পাইথন স্ট্যান্ডার্ড লাইব্রেরি এবং বিভিন্ন ডেটাসায়েন্স লাইব্রেরিগুলির সাথে পরিচয় করিয়ে দেব যা আপনাকে "চাকা পুনর্বীকরণ" এড়াতে সাহায্য করে। আপনি এই লাইব্রেরিগুলি ব্যবহার করে সক্ষওয়্যার অবজেক্ট তৈরি করবেন যার সাথে আপনি বিনয়ীভাবে গুরুত্বপূর্ণ কাজগুলি সম্পাদন করতে ইন্টারঅ্যাক্ট করবেন।
নির্দেশাবলীর সংখ্যা।

এরপর, আপনি তিনটি টেস্টড্রাইভের মাধ্যমে কাজ করবেন যেখানে পাইথন কোড কীভাবে কার্যকর করতে হয় তা দেখানো হবে:

- প্রথমটিতে, আপনি IPython ব্যবহার করে Python নির্দেশাবলী ইন্টারেক্ষিভভাবে কার্যকর করবেন এবং তাৎক্ষণিকভাবে তাদের ফলাফল দেখতে পাবেন।
- দ্বিতীয়টিতে, আপনি একটি বিশাল পাইথন অ্যাপ্লিকেশন কার্যকর করবেন যা একটি অ্যানিমেটেড বার চার্ট প্রদর্শন করবে যা ছয় পার্সিয়ুক্ত ডাইয়ের রোলগুলি সংক্ষেপে সংক্ষেপে প্রদর্শন করবে। আপনি "f Large Numbers" কার্যকর দেখতে পাবেন। বাহ পর্ব 6 এ, আপনি এই অ্যাপ্লিকেশনটি তৈরি করবেন ম্যাটপ্লটলিব ভিজুয়ালাইজেশন লাইব্রেরি।
- শেষ পর্যন্ত, আমরা JupyterLab ব্যবহার করে Jupyter Notebooks পরিচয় করিয়ে দেব—একটি ইন্টারেক্ষিভ, ওয়েব-ব্রাউজার-ভিত্তিক টুল যেখানে আপনি সুবিধাজনকভাবে Python নির্দেশাবলী লিখতে এবং কার্যকর করতে পারবেন। জুপিটার নোটবুক আপনাকে টেক্সট, ছবি, অডিও, ভিডিও, অ্যানিমেশন এবং অন্তর্ভুক্ত করতে সক্ষম করে কোড।

অতীতে, বেশিরভাগ কম্পিউটার অ্যাপ্লিকেশনগুলি স্বতন্ত্র কম্পিউটারে চলত (অর্থাৎ, একসাথে নেটওয়ার্ক করা হত না)। আজকের অ্যাপ্লিকেশনগুলি ইন্টারনেটের মাধ্যমে বিশ্বের কোটি কোটি কম্পিউটারের মধ্যে যোগাযোগের লক্ষ্যে লেখা যেতে পারে। আমরা ইন্টারডেক্স এবং ইন্টারনেট অফ থিংস (IoT) পরিচয় করিয়ে দেব, যা আপনার তৈরি করা সমসাময়িক অ্যাপ্লিকেশনগুলির ভিত্তি স্থাপন করবে।

১১-৬ তারিখের ঘটনা।

"বিগ ডেটা" কতটা বড় এবং কত দ্রুত এটি আরও বড় হচ্ছে তা আপনি জানতে পারবেন। এরপর, আমরা Waze মোবাইল নেভিগেশন অ্যাপের উপর একটি বিগডেটা কেস স্টাডি উপস্থাপন করব, যা অনেক বর্তমান প্রযুক্তি ব্যবহার করে গতিশীল ড্রাইভিং দিকনির্দেশনা প্রদান করে যা আপনাকে যত দ্রুত এবং নিরাপদে সম্ভব আপনার গন্তব্যে পৌঁছে দেয়। এই প্রযুক্তিগুলির মধ্য দিয়ে যাওয়ার সময়, আমরা এই বইটিতে উল্লেখ করব যে আপনি কোথায় সেগুলির অনেকগুলি ব্যবহার করবেন। আমাদের প্রথম ডেটা সায়েন্স ভূমিকা বিভাগের মাধ্যমে অধ্যায়টি শেষ হয় যেখানে আমরা কম্পিউটার বিজ্ঞান এবং ডেটা বিজ্ঞানের মধ্যে একটি গুরুত্বপূর্ণ সংযোগস্থল - কৃত্রিম বুদ্ধিমত্তা - নিয়ে আলোচনা করি।

১.২ বস্তু প্রযুক্তির মূল বিষয়গুলির একটি দ্রুত পর্যালোচনা

নতুন এবং আরও শক্তিশালী সফ্টওয়্যারের চাহিদা ক্রমশ বৃদ্ধি পাচ্ছে, তাই দ্রুত, সঠিকভাবে এবং অর্থনৈতিকভাবে সফ্টওয়্যার তৈরি করা গুরুত্বপূর্ণ। অবজেক্ট, অথবা আরও স্পষ্ট করে বলতে গেলে, অবজেক্টগুলি যে শ্রেণী থেকে আসে, মূলত পুনঃব্যবহারযোগ্য সফ্টওয়্যার উপাদান। তারিখের বস্তু, সময় বস্তু, অডিও বস্তু, ভিডিও বস্তু, অটোমোবাইল বস্তু, মানুষ বস্তু ইত্যাদি রয়েছে। প্রায় যেকোনো বিশেষ্যকে বৈশিষ্ট্য (যেমন, নাম, রঙ এবং আকার) এবং আচরণের (যেমন, গণনা, স্থানান্তর এবং যোগাযোগ) দিক থেকে সফ্টওয়্যার বস্তু হিসাবে পুকুরিসঙ্গতভাবে উপস্থাপন করা যেতে পারে। সফ্টওয়্যার উন্নয়ন গোষ্ঠীগুলি "স্ট্রাকচার্জ প্রোগ্রামিং" এর মতো পূর্ববর্তী জনপ্রিয় কোশলগুলির তুলনায় অনেক বেশি উৎপাদনশীল হওয়ার জন্য একটি মডুলার, বস্তু-ভিত্তিক নকশা এবং বাস্তবায়ন পদ্ধতি ব্যবহার করতে পারে। অবজেক্ট-ভিত্তিক প্রোগ্রামগুলি প্রায়শই বোঝা, সংশোধন এবং পরিবর্তন করা সহজ।

একটি বস্তু হিসেবে অটোমোবাইল

বস্তু এবং তার বিষয়বস্তু বুঝতে সাহায্য করার জন্য, আসুন একটি সহজ উপমা দিয়ে শুরু করি। ধরুন আপনি একটি গাড়ি চালাতে চান এবং এর অ্যাক্সিলারেটর প্যাডেল টিপে এটিকে দ্রুত চালাতে চান। এটি করার আগে আপনার কী ঘটতে হবে? আচ্ছা, আপনি একটি গাড়ি চালানোর আগে, কাউকে এটি ডিজাইন করতে হবে। একটি গাড়ি সাধারণত ইঞ্জিনিয়ারিং অঙ্কন হিসাবে শুরু হয়, যা নীলনকশা বর্ণনা করে

একটি বাড়ির প্রতীক। এই অঙ্গনগুলিতে একটি অ্যাক্সিলারেটর প্যাডেলের নকশা অন্তর্ভুক্ত রয়েছে। প্যাডেলটি ড্রাইভারের কাছ থেকে জাটিল প্রক্রিয়াগুলি লুকিয়ে রাখে যা গাড়িকে দ্রুত চালায়, ঠিক যেমন ব্রেক প্যাডেল গাড়ির গতি কমিয়ে দেয় এমন প্রক্রিয়াগুলিকে "লুকিয়ে রাখে" এবং স্টিয়ারিং হাইল গাড়ি ঘুরিয়ে দেয় এমন প্রক্রিয়াগুলিকে "লুকিয়ে রাখে"। এর ফলে ইঞ্জিন, ব্রেকিং এবং স্টিয়ারিং প্রক্রিয়া কীভাবে কাজ করে সে সম্পর্কে খুব কম বা কোনও জ্ঞান নেই এমন লোকেরা সহজেই গাড়ি চালাতে সক্ষম হয়।

রান্নাঘরের নকশায় যেমন আপনি খাবার রান্না করতে পারবেন না, তেমনি আপনি গাড়ির ইঞ্জিনিয়ারিং অঙ্গনও চালাতে পারবেন না। গাড়ি চালানোর আগে, গাড়িটি তার বর্ণনা অনুযায়ী তৈরি করতে হবে। একটি সম্পূর্ণ গাড়ি দ্রুত গতিতে চালানোর জন্য একটি প্রকৃত অ্যাক্সিলারেটর প্যাডেল থাকে, কিন্তু তাও যথেষ্ট নয়—গাড়িটি নিজে থেকে গতি বাঢ়াবে না (আশা করি!), তাই গাড়িটি দ্রুত গতিতে চালানোর জন্য চালককে প্যাডেল টিপতে হবে।

পদ্ধতি এবং ক্লাস

চলুন আমাদের গাড়ির উদাহরণ ব্যবহার করে কিছু মূল অবজেক্ট ও রিয়েন্টেড প্রোগ্রামিং ধারণা উপস্থাপন করি। একটি প্রোগ্রামে একটি কাজ সম্পাদনের জন্য একটি পদ্ধতির প্রয়োজন হয়। পদ্ধতিটি প্রোগ্রামের বিবৃতিগুলিকে ধারণ করে যা তার কাজগুলি সম্পাদন করে। পদ্ধতিটি তার ব্যবহারকারীর কাছ থেকে এই বিবৃতিগুলিকে লুকিয়ে রাখে, ঠিক যেমন একটি গাড়ির অ্যাক্সিলারেটর প্যাডেল ড্রাইভারের কাছ থেকে গাড়িটিকে দ্রুত চালানোর প্রক্রিয়াগুলি লুকিয়ে রাখে। পাইখনে, একটি প্রোগ্রাম ইউনিট যাকে **ক্লাস** বলা হয়, ক্লাসের কাজগুলি সম্পাদন করার পদ্ধতিগুলির একটি সেট ধারণ করে। উদাহরণস্বরূপ, একটি ব্যাক্স অ্যাকাউন্টের প্রতিনিধিত্বকারী একটি ক্লাস একটি অ্যাকাউন্টে অর্থ জমা করার একটি পদ্ধতি, অন্যটি একটি অ্যাকাউন্ট থেকে অর্থ উত্তোলনের জন্য এবং তৃতীয়টি অ্যাকাউন্টের ব্যালেন্স কী তা জিজ্ঞাসা করার জন্য থাকতে পারে। একটি ক্লাস ধারণার দিক থেকে একটি গাড়ির ইঞ্জিনিয়ারিং অঙ্গনের মতো, যেখানে একটি অ্যাক্সিলারেটর প্যাডেল, স্টিয়ারিং হাইল ইত্যাদির নকশা থাকে।

ইনস্ট্যালিয়েশন

ঠিক যেমন কাউকে গাড়ি চালানোর আগে তার ইঞ্জিনিয়ারিং অঙ্গন থেকে একটি গাড়ি তৈরি করতে হয়, তেমনি একটি প্রোগ্রাম ক্লাসের পদ্ধতি দ্বারা নির্ধারিত কাজগুলি সম্পাদন করার আগে আপনাকে অবশ্যই একটি ক্লাসের একটি অবজেক্ট তৈরি করতে হবে। এটি করার প্রক্রিয়াটিকে ইনস্ট্যালিয়েশন বলা হয়। তারপর একটি অবজেক্টকে তার ক্লাসের একটি **উদাহরণ** হিসাবে উল্লেখ করা হয়।

পুনঃব্যবহার

ঠিক যেমন একটি গাড়ির ইঞ্জিনিয়ারিং অঙ্গন বহুবার ব্যবহার করে অনেক গাড়ি তৈরি করা যায়, তেমনি আপনি একটি ক্লাসকে বহুবার ব্যবহার করে অনেক জিনিস তৈরি করতে পারেন। নতুন ক্লাস এবং প্রোগ্রাম তৈরি করার সময় বিদ্যমান ক্লাসগুলির পুনঃব্যবহার সময় এবং শ্রম সাধারণ করে। পুনঃব্যবহার আপনাকে আরও নির্ভরযোগ্য এবং কার্যকর সিস্টেম তৈরি করতেও সাহায্য করে কারণ বিদ্যমান ক্লাস এবং উপাদানগুলি প্রায়শই ব্যাপক পরিকল্পনা, ডিবাগিং এবং কর্মক্ষমতা টিউনিংয়ের মধ্য দিয়ে গেছে। শিল্প বিপ্লবের জন্য বিনিয়োগ্য যন্ত্রাংশের ধারণা যেমন অত্যন্ত গুরুত্বপূর্ণ ছিল, তেমনি বস্তু প্রযুক্তি দ্বারা উদ্ভৃত সফ্টওয়্যার বিপ্লবের জন্য পুনর্ব্যবহারযোগ্য ক্লাসগুলি অত্যন্ত গুরুত্বপূর্ণ।

পাইখনে, আপনি সাধারণত আপনার প্রোগ্রাম তৈরি করার জন্য একটি বিল্ডিং রুক পদ্ধতি ব্যবহার করবেন। চাকাটি পুনরায় উন্নোবন এড়াতে, যেখানেই সম্ভব বিদ্যমান উচ্চমানের টুকরো ব্যবহার করবেন। এই সফ্টওয়্যার পুনঃব্যবহার অবজেক্ট-ওরিয়েন্টেড প্রোগ্রামিংয়ের একটি মূল সুবিধা।

বার্তা এবং পদ্ধতি কল

যখন আপনি একটি গাড়ি চালান, তখন এর গ্যাস প্যাডেল টিপলে গাড়িতে একটি কাজ সম্পাদনের জন্য একটি বার্তা পাঠানো হয় - অর্থাৎ দ্রুত যেতে। একইভাবে, আপনি একটি বস্তুকে বার্তা পাঠান। প্রতিটি বার্তা একটি পদ্ধতি কল হিসাবে বাস্তবায়িত হয় যা বস্তুর একটি পদ্ধতিকে তার কাজ সম্পাদন করার জন্য বলে। উদাহরণস্বরূপ, একটি প্রোগ্রাম অ্যাকাউন্টের ব্যালেন্স বাড়ানোর জন্য একটি ব্যাংক অ্যাকাউন্ট বস্তুর জমা পদ্ধতিতে কল করতে পারে।

বৈশিষ্ট্য এবং ইনস্ট্যান্স ভেরিয়েবল

একটি গাড়ির কাজ সম্পন্ন করার ক্ষমতা ছাড়াও, এর কিছু বৈশিষ্ট্যও রয়েছে, যেমন এর রঙ, দরজার সংখ্যা, ট্যাঙ্কে গ্যাসের পরিমাণ, বর্তমান গতি এবং মোট মাইল চালিত রেকর্ড (অর্থাৎ, এর ওডোমিটার রিডিং)। এর ক্ষমতার মতো, গাড়ির বৈশিষ্ট্যগুলি তার ইঞ্জিনিয়ারিং ডায়াগ্রামে (যেমন, একটি ওডোমিটার এবং একটি জ্বালানি পরিমাপক যন্ত্র অন্তর্ভুক্ত) এর নকশার অংশ হিসাবে উপস্থাপন করা হয়েছে। আপনি যখন একটি প্রকৃত গাড়ি চালান, তখন এই বৈশিষ্ট্যগুলি গাড়ির সাথে বহন করা হয়। প্রতিটি গাড়ি তার নিজস্ব বৈশিষ্ট্য বজায় রাখে। উদাহরণস্বরূপ, প্রতিটি গাড়ি জানে যে তার নিজস্ব গ্যাস ট্যাঙ্কে কত গ্যাস আছে, কিন্তু অন্য গাড়ির ট্যাঙ্কে কত আছে তা নয়।

একইভাবে, একটি অবজেক্টেরও কিছু বৈশিষ্ট্য থাকে যা এটি একটি প্রোগ্রামে ব্যবহারের সময় বহন করে। এই বৈশিষ্ট্যগুলি অবজেক্টের ক্লাসের অংশ হিসাবে নির্দিষ্ট করা হয়। উদাহরণস্বরূপ, একটি ব্যাংক অ্যাকাউন্ট অবজেক্টের একটি ব্যালেন্স অ্যাট্রিবিউট থাকে যা অ্যাকাউন্টে থাকা অর্থের পরিমাণকে প্রতিনিধিত্ব করে। প্রতিটি ব্যাংক অ্যাকাউন্ট অবজেক্ট যে অ্যাকাউন্টটি প্রতিনিধিত্ব করে তার ব্যালেন্স জানে, কিন্তু ব্যাংকের অন্যান্য অ্যাকাউন্টের ব্যালেন্স জানে না। বৈশিষ্ট্যগুলি ক্লাসের ইনস্ট্যান্স ভেরিয়েবল দ্বারা নির্দিষ্ট করা হয়। একটি ক্লাসের (এবং এর অবজেক্টের) বৈশিষ্ট্য এবং পদ্ধতিগুলি ঘনিষ্ঠভাবে সম্পর্কিত, তাই ক্লাসগুলি তাদের বৈশিষ্ট্য এবং পদ্ধতিগুলিকে একত্রিত করে।

উত্তরাধিকার

উত্তরাধিকারের মাধ্যমে সুবিধাজনকভাবে একটি নতুন শ্রেণীর বস্তু তৈরি করা যেতে পারে — নতুন শ্রেণী (যাকে সাবক্লাস বলা হয়) একটি বিদ্যমান শ্রেণীর (যাকে সুপারক্লাস বলা হয়) বৈশিষ্ট্য দিয়ে শুরু হয়, সম্ভবত সেগুলিকে কাস্টমাইজ করে এবং নিজস্ব অন্য বৈশিষ্ট্য যুক্ত করে। আমাদের গাড়ির উপমায়, "রূপান্তরযোগ্য" শ্রেণীর একটি বস্তু অবশ্যই আরও সাধারণ শ্রেণীর "অটোমোবাইল" এর একটি বস্তু, তবে আরও নির্দিষ্টভাবে বলতে গেলে, ছাদটি উঁচু বা নিচু করা যেতে পারে।

অবজেক্ট-ওরিয়েন্টেড অ্যানালাইসিস অ্যান্ড ডিজাইন (OOAD)

শীষ্টাই তুমি পাইথনে প্রোগ্রাম লিখবে। তুমি কিভাবে তোমার প্রোগ্রামের কোড তৈরি করবে?

হয়তো, অনেক প্রোগ্রামারের মতো, তুমি তোমার কম্পিউটার চালু করে টাইপ করা শুরু করবে। এই পদ্ধতিটি ছোট প্রোগ্রামের জন্য কাজ করতে পারে (যেমন আমরা বইয়ের প্রথম দিকের অধ্যায়গুলিতে উপস্থাপন করেছি), কিন্তু যদি তোমাকে একটি বড় ব্যাংকের জন্য হাজার হাজার স্বয়ংক্রিয় টেলার মেশিন নিয়ন্ত্রণ করার জন্য একটি সফ্টওয়্যার সিস্টেম তৈরি করতে বলা হয়? অথবা ধরো তোমাকে পরবর্তী প্রজয়ের মার্কিন বিমান চলাচল নিয়ন্ত্রণ ব্যবস্থা তৈরির জন্য ১,০০০ সফ্টওয়্যার ডেভেলপারের একটি দলে কাজ করতে বলা হয়? এত বড় এবং জটিল প্রকল্পের জন্য, তোমার কেবল বসে লেখা শুরু করা উচিত নয়।

প্রোগ্রাম।

সর্বোত্তম সমাধান তৈরি করার জন্য, আপনার প্রকল্পের প্রয়োজনীয়তা নির্ধারণের জন্য একটি বিশদ বিশ্লেষণ প্রক্রিয়া অনুসরণ করা উচিত (অর্থাৎ, সিস্টেমটি কী করতে হবে তা নির্ধারণ করা), তারপর এমন একটি **নকশা** তৈরি করা উচিত যা তাদের সম্পৃষ্ট করে (অর্থাৎ, সিস্টেমটি কীভাবে এটি করবে তা নির্দিষ্ট করে)। আদর্শভাবে, আপনি এই প্রক্রিয়াটি অতিক্রম করবেন এবং সাবধানতার সাথে নকশাটি পর্যালোচনা করবেন (এবং আপনার নকশা পর্যালোচনা করাবেন)।

(কোনও কোড লেখার আগে) অন্যান্য সংস্কৃতিগত পেশাদারদের সাথে পরামর্শ করুন। যদি এই প্রক্রিয়ায় বিশেষণ জড়িত থাকে এবং আপনার সিস্টেমকে অবজেক্ট-ওরিয়েন্টেড দৃষ্টিকোণ থেকে ডিজাইন করার সময়, এটিকে অবজেক্ট-ওরিয়েন্টেড অ্যানালাইসিস অ্যান্ড ডিজাইন (OOAD) প্রক্রিয়া বলা হয়। পাইথনের মতো ভাষাগুলি অবজেক্ট-ওরিয়েন্টেড। অবজেক্ট-ওরিয়েন্টেড প্রোগ্রামিং (OOP) নামক এই ভাষায় প্রোগ্রামিং আপনাকে একটি কার্যকরী সিস্টেম হিসাবে অবজেক্ট-ওরিয়েন্টেড ডিজাইন বাস্তবায়ন করতে দেয়।

১.৩ পাইথন

পাইথন একটি অবজেক্ট-ওরিয়েন্টেড স্ক্রিপ্টিং ভাষা যা ১৯৯১ সালে সর্বজনীনভাবে প্রকাশিত হয়েছিল। এটি আমস্টারডামের ন্যাশনাল রিসার্চ ইনসিটিউট ফর ম্যাথমেটিক্যাল অ্যান্ড কম্পিউটার সায়েন্সের গুইডো ভ্যান রসাম দ্বারা তৈরি করা হয়েছিল।

পাইথন দ্রুত বিশ্বের সবচেয়ে জনপ্রিয় প্রোগ্রামিং ভাষাগুলির মধ্যে একটি হয়ে উঠেছে। এটি এখন শিক্ষামূলক এবং বৈজ্ঞানিক কম্পিউটিংয়ের জন্য বিশেষভাবে জনপ্রিয়, এবং সম্প্রতি এটি সবচেয়ে জনপ্রিয় ডেটাসায়েন্স প্রোগ্রামিং ভাষা হিসাবে প্রোগ্রামিং ভাষা R কে ছাড়িয়ে গেছে।

^৮,
^৯,

পাইথন কেন জনপ্রিয় এবং প্রত্যেকেরই এটি শেখার কথা বিবেচনা করা উচিত তার কিছু কারণ এখানে দেওয়া হল:

৮

^২ টিটিপিএস://www.oreilly.com/ideas/5thingstowatchinpythonin2017।

^৩ <https://www.kdnuggets.com/2017/08/pythonovertakesrleader-ন্যালিটিক্সডেটাসায়েন্স.এইচটি.এমএল।>

^৪ <https://www.rbloggers.com/datasciencejobreport2017rpasses-যেমনকি স্ক্রিপ্টঅ্যাজগরপাড়াওতাদেরউভয়পিছনে।>

^৫ টিটিপিএস://www.oreilly.com/ideas/5thingstowatchinpythonin2017।

^৬ <https://dbader.org/blog/whylearnpython.>

^৭ <https://simpleprogrammer.com/2017/01/18/7reasonswhyyoushould-আর্নপাইথন।>

^৮ টিটিপিএস://www.oreilly.com/ideas/5thingstowatchinpythonin2017।

- এটি ওপেন সোর্স, বিনামূলে এবং বিশাল ওপেন সোর্স সম্প্রদায়ের সাথে ব্যাপকভাবে উপলব্ধ।

- C, C++, C# এবং জাভার মতো ভাষাগুলির তুলনায় এটি শেখা সহজ, যা নবীন এবং পেশাদার ডেভেলপারদের দ্রুত গতিতে কাজ করতে সাহায্য করে।

- এটি অন্যান্য অনেক জনপ্রিয় প্রোগ্রামিং ভাষার তুলনায় পড়া সহজ।

- এটি শিক্ষাক্ষেত্রে ব্যাপকভাবে ব্যবহৃত হয়।

৯

^{১০} টোলারভে, এন., শিক্ষায় পাইথন: শেখান, শেখান, প্রোগ্রাম (ও'রিলি মিডিয়া, ইনকর্পোরেটেড, ২০১৫)।

- এটি বিস্তৃত স্ট্যান্ডার্ড লাইব্রেরি এবং থার্ড-পার্টি ওপেনসোর্স লাইব্রেরি সহ ডেভেলপারদের উৎপাদনশীলতা বৃদ্ধি করে, যাতে প্রোগ্রামাররা দ্রুত কোড লিখতে পারে এবং ন্যূনতম কোড ব্যবহার করে জটিল কাজ সম্পাদন করতে পারে। আমরা এই সম্পর্কে আরও বিস্তারিত আলোচনা করব বিভাগ ১.৪-এ।

- প্রচুর সংখ্যক বিনামূল্যের ওপেনসোর্স পাইথন অ্যাপ্লিকেশন রয়েছে।
- এটি ওয়েব ডেভেলপমেন্টে (যেমন, জ্যাপ্সো, ফ্লাস্ক) জনপ্রিয়।
- এটি জনপ্রিয় প্রোগ্রামিং প্যারাডিজমগুলিকে সমর্থন করে—প্রক্রিয়াগত, কার্যকরী শৈলী এবং অবজেক্ট-ওরিয়েন্টেড। আমরা পর্ব ৪^০ এ কার্যকরী শৈলী প্রোগ্রামিং বৈশিষ্ট্যগুলি প্রবর্তন শুরু করব এবং পরবর্তী অধ্যায়গুলিতে সেগুলি ব্যবহার করব।

^০ [https://en.wikipedia.org/wiki/Python_\(programming_language\)](https://en.wikipedia.org/wiki/Python_(programming_language))

- এটি সমকালীন প্রোগ্রামিংকে সহজ করে তোলে—অ্যাসিনসিও এবং অ্যাসিন্ক/অপেক্ষার সাহায্যে, আপনি একক থ্রেডে সমকালীন কোড লিখতে সক্ষম। ^১, এর অন্তর্নিহিত জটিল প্রক্রিয়াগুলিকে ব্যাপকভাবে সরল করে তোলা ^২ কোডটি লেখা, ডিবাগ করা এবং রক্ষণাবেক্ষণ করা। ^৩

^১ <https://docs.python.org/3/library/asyncio.html>.

^২ <https://www.oreilly.com/ideas/5thingstowatchinpythonin-017>।

- পাইথনের কর্মক্ষমতা বৃদ্ধির জন্য প্রচুর ক্ষমতা রয়েছে।
- এটি সহজ স্ক্রিপ্ট থেকে শুরু করে জটিল অ্যাপ পর্যন্ত যেকোনো কিছু তৈরি করতে ব্যবহৃত হয় যার মধ্যে প্রচুর পরিমাণে ব্যবহারকারীদের, যেমন ড্রপবক্স, ইউটিউব, রেডিট, ইনস্টাগ্রাম এবং কোওরা। ^৪

^৩ https://www.hartmannsoftware.com/Blog/Articles_from_Software_Fans/Most_famousSoftwareProgramsWritteninPython.com.

- এটি কৃতিম বুদ্ধিমত্তায় জনপ্রিয়, যা বিস্ফোরক প্রবৃদ্ধি উপভোগ করছে, আংশিকভাবে ডেটা সায়েন্সের সাথে এর বিশেষ সম্পর্কের কারণে।
- এটি আর্থিক সম্পদায়ে ব্যাপকভাবে ব্যবহৃত হয়। ^৫

^৪ কোলানোভিচ, এম. এবং আর. কৃষ্ণমাচারি, বিগ ডেটা এবং এআই কৌশল: মেশিন লার্নিং এবং বিনিয়োগের জন্য বিকল্প ডেটা পদ্ধতি (জেপি মরগান, ২০১৭)।

- বিভিন্ন শাখায়, বিশেষ করে ডেটা সায়েন্স-ভিত্তিক পদে, পাইথন প্রোগ্রামারদের জন্য একটি বিস্তৃত চাকরির বাজার রয়েছে এবং সমস্ত প্রোগ্রামিং চাকরির মধ্যে পাইথনের চাকরিগুলি সর্বোচ্চ বেতনের মধ্যে একটি।

^৫ [৫](#)

^৬ <https://www.infoworld.com/article/3170838/developer/getpaid10-ways-to-earn-money-programming.html>

^৭ <https://medium.com/@ChallengeRocket/top10ofprogramming-www.EBooksWorld.ir>

- পরিসংখ্যানগত অ্যালগোরিদমের সাথে অ্যাঙ্গুয়েজা, ৪৬৮২৫৬ই।
- পাইথন এবং R হল দুটি বহুল ব্যবহৃত ডেটাসায়েন্স ভাষা।

অ্যানাকোন্ডা পাইথন বিতরণ

আমরা Anaconda Python ডিস্ট্রিবিউশন ব্যবহার করি কারণ এটি Windows, macOS এবং Linux-এ ইনস্টল করা সহজ এবং Python-এর সর্বশেষ সংস্করণ, IPython ইন্টারপ্রেটার (version 1.5.1-এ প্রবর্তিত) এবং Jupyter Notebooks (version 1.5.3-এ প্রবর্তিত) সমর্থন করে।

Anaconda-তে আরও রয়েছে

পাইথন প্রোগ্রামিং এবং ডেটা সায়েন্সে সাধারণত ব্যবহৃত অন্যান্য সফ্টওয়্যার প্যাকেজ এবং লাইব্রেরি, যা আপনাকে সফ্টওয়্যার ইনস্টলেশনের সমস্যাগুলির পরিবর্তে পাইথন এবং ডেটা সায়েন্সের উপর মনোযোগ দেওয়ার সুযোগ দেয়। আইপিথন ইন্টারপ্রেটারে এমন বৈশিষ্ট্য রয়েছে যা আপনাকে পাইথন, পাইথন স্ট্যান্ডার্ড লাইব্রেরি এবং বিস্তৃত তৃতীয় পক্ষের লাইব্রেরিগুলি অন্বেষণ, আবিষ্কার এবং পরীক্ষা করতে সহায়তা করে।

^১ <https://ipython.org/>.

পাইথনের জেন

আমরা টিম পিটার্সের 'দ্য জেন অফ পাইথন' বইটি মনে চলি, যেখানে পাইথনের প্রাঞ্চি গুইডো ভ্যান রসামের ভাষার নকশা নীতির সারসংক্ষেপ তুলে ধরা হয়েছে। এই তালিকাটি আইপিথনে 'ইমপোর্ট থিস' কমান্ডের মাধ্যমে দেখা যাবে। পাইথনের জেনকে পাইথন এনহ্যালমেন্ট প্রপোজাল (PEP) 20-এ সংজ্ঞায়িত করা হয়েছে। "একটি PEP হল একটি নকশা নথি যা পাইথন সম্প্রদায়কে তথ্য প্রদান করে, অথবা পাইথন বা এর প্রক্রিয়া বা পরিবেশের জন্য একটি নতুন বৈশিষ্ট্য বর্ণনা করে।"

^২

^৩ <https://chitchat.python.org/dev/peps/pep0001/>

১.৪ এগুলো লাইব্রেরি!

পুরো বই জুড়ে, আমরা বিদ্যমান লাইব্রেরিগুলি ব্যবহার করার উপর জোর দিয়েছি যাতে আপনি "চাকা পুনর্বীকরণ" এড়াতে পারেন, যার ফলে আপনার প্রোগ্রাম ডেভেলপমেন্ট প্রচেষ্টার সুবিধা পাওয়া যায়। প্রায়শই, প্রচুর মূল কোড তৈরি করার পরিবর্তে - একটি ব্যবহৃত এবং সময়সাপেক্ষ প্রক্রিয়া - আপনি কেবল একটি পূর্ব-বিদ্যমান লাইব্রেরি ক্লাসের একটি অবজেক্ট তৈরি করতে পারেন, যার জন্য শুধুমাত্র একটি পাইথন স্টেটমেন্ট লাগে। সুতরাং, লাইব্রেরিগুলি আপনাকে সামান্য পরিমাণে কোড দিয়ে গুরুত্বপূর্ণ কাজ সম্পাদন করতে সহায়তা করবে। এই বইটিতে, আপনি পাইথন স্ট্যান্ডার্ড লাইব্রেরি, ডেটাসায়েন্স লাইব্রেরি এবং তৃতীয় পক্ষের লাইব্রেরির বিস্তৃত পরিসর ব্যবহার করবেন।

১.৪.১ পাইথন স্ট্যান্ডার্ড লাইব্রেরি

পাইথন স্ট্যান্ডার্ড লাইব্রেরি টেক্সট/বাইনারি ডেটা প্রসেসিং, গণিত, ফাংশনাল স্টাইল প্রোগ্রামিং, ফাইল/ডিরেক্টরি অ্যাক্সেস, ডেটা পারসিস্টেন্স, ডেটা ক্ষেপণ/আর্কাইভিং, ক্রিপ্টোগ্রাফি, অপারেটিং সিস্টেম পরিষেবা, সমবর্তী প্রোগ্রামিং, ইন্টারপ্রেস কমিউনিকেশন, নেটওয়ার্কিং প্রোটোকল, JSON/XML/অন্যান্য ইন্টারনেট ডেটা ফর্ম্যাট, মাল্টিমিডিয়া, আন্তর্জাতিক করণ, GUI, ডিবার্গিং, প্রোফাইলিং এবং আরও অনেক কিছুর জন্য সমন্বক্ষণ প্রদান করে। নিম্নলিখিত টেবিলে পাইথন স্ট্যান্ডার্ড লাইব্রেরি মডিউলগুলির কিছু তালিকা দেওয়া হয়েছে যা আমরা উদাহরণ হিসেবে ব্যবহার করি।

বাইটিতে আমরা যে পাইথন স্ট্যান্ডার্ড লাইব্রেরি মডিউলগুলি ব্যবহার করি তার কিছু

সংগ্রহ—অতিরিক্ত তথ্য

তালিকার বাইরের কাঠামো, টিপল,
অভিধান এবং সেট।

CSV—কমা দ্বারা পৃথক করা মান প্রক্রিয়াকরণ
ফাইল।

তারিখ-সময়—তারিখ এবং সময়ের হেরফের।

দশমিক—স্থিরবিন্দু এবং ভাসমানবিন্দু পাটিগণিত, আর্থিক গণনা সহ।

ডক্সেট—ডকস্ট্রিং-এ এমবেড করা বৈধতা পরীক্ষার
মাধ্যমে সহজ ইউনিট পরীক্ষা এবং প্রত্যাশিত ফলাফল।

json—ওয়েব পরিষেবা এবং NoSQL ডকুমেন্ট ডাটাবেসের সাথে
ব্যবহারের জন্য জাভাস্ক্রিপ্ট অবজেক্ট নোটেশন (JSON)
প্রক্রিয়াকরণ।

গণিত—সাধারণ গণিত ফ্রেক এবং
অপারেশন।

OS—অপারেটিং এর সাথে ইন্টারঅ্যাক্স করা

সিস্টেম।

সারি—প্রথম, প্রথম আউট ডেটা
গঠন।

এলোমেলো—ছদ্ম-র্যান্ডম সংখ্যা।

sqlite3—SQLite রিলেশনাল ডাটাবেস
অ্যাক্সেস।

পরিসংখ্যান—গাণিতিক পরিসংখ্যান
গড়, মধ্যমা, মোডের মতো ফাংশন
এবং বৈচিত্র্য।

স্ট্রিং—স্ট্রিং প্রক্রিয়াকরণ।

sys—কমান্ডলাইন আঙ্গুমেন্ট প্রক্রিয়াকরণ;
স্ট্যান্ডার্ড ইনপুট, স্ট্যান্ডার্ড আউটপুট এবং স্ট্যান্ডার্ড এরর
স্ট্রিম।

timeit—কর্মক্ষমতা বিশ্লেষণ।

১.৪.২ ডেটা-সায়েন্স লাইব্রেরি

পাইথনের বিভিন্ন ক্ষেত্রে ওপেনসোর্স ডেভেলপারদের একটি বিশাল এবং দ্রুত বর্ধনশীল সম্প্রদায় রয়েছে। পাইথনের জনপ্রিয়তার অন্যতম বড় কারণ হল এর ওপেনসোর্স সম্প্রদায় দ্বারা তৈরি ওপেন-সোর্স লাইব্রেরির অসাধারণ পরিসর। আমাদের লক্ষ্যগুলির মধ্যে একটি হল উদাহরণ তৈরি করা এবং কেস স্টাডি বাস্তবায়ন করা যা আপনাকে পাইথন প্রোগ্রামিংয়ের একটি আকর্ষণীয়, চ্যালেঞ্জিং এবং বিনোদনমূলক ভূমিকা প্রদান করে, পাশাপাশি আপনাকে হাতে কলমে ডেটা সায়েন্স, মূল ডেটা সায়েন্স লাইব্রেরি এবং আরও অনেক কিছুতে জড়িত করে। মাত্র কয়েকটি লাইন কোডে আপনি যে গুরুত্বপূর্ণ কাজগুলি সম্পন্ন করতে পারেন তা দেখে আপনি অবাক হয়ে যাবেন। নিম্নলিখিত টেবিলে বিভিন্ন জনপ্রিয় ডেটা-সায়েন্স লাইব্রেরির তালিকা রয়েছে। আমাদের ডেটা সায়েন্স উদাহরণগুলি পড়ার সময় আপনি এর অনেকগুলি ব্যবহার করবেন।

ভিজুয়ালাইজেশনের জন্য, আমরা Matplotlib, Seaborn এবং Folium ব্যবহার করব, তবে আরও অনেক কিছু আছে। Python ভিজুয়ালাইজেশন লাইব্রেরির একটি সুন্দর সারাংশের জন্য <http://pyviz.org/> দেখুন।

বৈজ্ঞানিক কম্পিউটিং এবং পরিসংখ্যান

NumPy (সংখ্যাসূচক পাইথন)-পাইথনের কোনও বিল্টইন অ্যারে ডেটা স্ট্রাকচার নেই।

এটি তালিকা ব্যবহার করে, যা সুবিধাজনক কিন্তু তুলনামূলকভাবে ধীর। NumPy তালিকা এবং ম্যাট্রিক্স উপস্থাপনের জন্য উচ্চ-কার্যক্ষমতাসম্পন্ন ndarray ডেটা স্ট্রাকচার প্রদান করে এবং এটি এই ধরনের ডেটা স্ট্রাকচার প্রক্রিয়াকরণের জন্য রুটিনও প্রদান করে।

SciPy (বৈজ্ঞানিক পাইথন)—NumPy-তে নির্মিত, SciPy বৈজ্ঞানিক প্রক্রিয়াকরণের জন্য রুটিন যোগ করে, যেমন ইন্টিগ্রাল, ডিফারেনশিয়াল সমীকরণ, অতিরিক্ত ম্যাট্রিক্স প্রক্রিয়াকরণ এবং আরও অনেক কিছু। scipy.org SciPy এবং NumPy নিয়ন্ত্রণ করে।

StatsModels—পরিসংখ্যানগত মডেল, পরিসংখ্যানগত পরীক্ষা এবং পরিসংখ্যানগত তথ্য অনুসন্ধানের অনুমানের জন্য সহায়তা প্রদান করে।

ডেটা ম্যানিপুলেশন এবং বিশ্লেষণ

পান্ডাস—ডেটা ম্যানিপুলেশনের জন্য একটি অত্যন্ত জনপ্রিয় লাইব্রেরি। পান্ডাস NumPy-এর ndarray-এর প্রচুর ব্যবহার করে। এর দুটি মূল ডেটা স্ট্রাকচার হল সিরিজ (এক মাত্রিক) এবং ডেটাফ্রেম (দ্বি মাত্রিক)।

ডিজ্যুয়ালাইজেশন

ম্যাটপ্লটলিব—একটি অত্যন্ত কাস্টমাইজযোগ্য ডিজ্যুয়ালাইজেশন এবং প্লটিং লাইব্রেরি। সমর্থিত প্লটগুলির মধ্যে রয়েছে রেগুলার, স্ক্যাটার, বার, কনট্যুর, পাই, কোয়েডার, প্রিড, পোলার অক্ষ, 3D এবং টেক্সট।

সির্বর্ন—ম্যাটপ্লটলিবের উপর নির্মিত একটি উচ্চ স্তরের ডিজ্যুয়ালাইজেশন লাইব্রেরি। সির্বর্ন আরও সুল্দর চেহারা এবং অনুভূতি, অতিরিক্ত ডিজ্যুয়ালাইজেশন যোগ করে এবং আপনাকে কম কোড ব্যবহার করে ডিজ্যুয়ালাইজেশন তৈরি করতে সক্ষম করে।

মেশিন লার্নিং, ডিপ লার্নিং এবং রিইনফোর্সমেন্ট লার্নিং

scikitlearn—সেরা মেশিনলার্নিং লাইব্রেরি। মেশিন লার্নিং হলো AI এর একটি উপসেট।

ডিপ লার্নিং হলো মেশিন লার্নিংয়ের একটি উপসেট যা নিউরাল নেটওয়ার্কের উপর দৃষ্টি নিবন্ধন করে।

ensorFlow (গুগল), CNTK (গভীর শিক্ষার জন্য মাইক্রোসফটের জ্ঞানীয় টুলকিট) অথবা
থিয়েনো (মন্ত্রিল বিশ্ববিদ্যালয়।)

টেনসরফ্লো—গুগলের মতে, এটি সর্বাধিক ব্যবহৃত ডিপ লার্নিং লাইব্রেরি।

TensorFlow পারফরম্যান্সের জন্য GPU (গ্রাফিক্স প্রসেসিং ইউনিট) অথবা Google এর কাস্টম TPU (টেলর প্রসেসিং ইউনিট) এর সাথে
কাজ করে। TensorFlow AI এবং বিগ ডেটা অ্যানালিটিক্সে গুরুত্বপূর্ণ—যেখানে প্রক্রিয়াকরণের চাহিদা প্রচুর। আপনি TensorFlow-এর
মধ্যে তৈরি Keras সংস্করণটি ব্যবহার করবেন।

ওপেনএআই জিম—রিইনফোর্সমেন্ট লার্নিং অ্যালগরিদম তৈরি, পরীক্ষা এবং তুলনা করার জন্য একটি লাইব্রেরি এবং
পরিবেশ।

প্রাকৃতিক ভাষা প্রক্রিয়াকরণ (NLP)

NLTK (প্রাকৃতিক ভাষা টুলকিট)- প্রাকৃতিক ভাষা প্রক্রিয়াকরণ (NLP) কাজের জন্য ব্যবহৃত।

টেক্সটেলব—NLTK এবং প্যাটার্ন NLP লাইব্রেরির উপর ভিত্তি করে তৈরি একটি অবজেক্ট-ওরিয়েন্টেড NLP টেক্সট
প্রসেসিং লাইব্রেরি। টেক্সটেলব অনেক NLP কাজকে সহজ করে।

জেনসিম—NLTK-এর অনুরূপ। সাধারণত নথির সংগ্রহের জন্য একটি সূচক তৈরি করতে ব্যবহৃত হয়, তারপর নির্ধারণ
করতে হয় যে অন্য নথিটি প্রতিটির সাথে কতটা মিল
সূচক।

১.৫ টেস্ট-ড্রাইভ: আইপাইথন এবং জুপিটার নোটবুক ব্যবহার

এই বিভাগে, আপনি দুটি মোডে IPython ইন্টারপ্রেটার টেস্টড্রাইভ করবেন:

ঘৰে এই বিভাগটি পড়ার আগে, ইনস্টল করার জন্য "শুরু করার আগে" বিভাগে দেওয়া নির্দেশাবলী অনুসরণ করুন
অ্যানাকোন্ডা পাইথন ডিস্ট্রিবিউশন, যাতে আইপাইথন ইন্টারপ্রেটার রয়েছে।

- ইন্টারেক্শন মোডে, আপনি **স্লিপেটস** নামক পাইথন কোডের ছোট ছোট অংশ প্রবেশ করাবেন এবং অবিলম্বে তাদের
ফলাফল দেখতে পাবেন।
- ক্রিপ্ট মোডে**, আপনি .py এক্সেনশন (সংক্ষিপ্ত) আছে এমন একটি ফাইল থেকে লোড করা কোডটি কার্যকর করবেন।

পাইথনের জন্য)। এই ধরনের ফাইলগুলিকে স্ক্রিপ্ট বা প্রোগ্রাম বলা হয় এবং এগুলি সাধারণত ইন্টারেক্ষিভ মোডে আপনি যে কোড স্লিপেটগুলি ব্যবহার করবেন তার চেয়ে লম্বা হয়।

তারপর, আপনি শিখবেন কিভাবে পাইথন কোড লেখা এবং কার্যকর করার জন্য জুপিটার নোটবুক নামে পরিচিত ব্রাউজার-ভিত্তিক পরিবেশ ব্যবহার করতে হয়।

⁰ জুপিটার অনেক প্রোগ্রামিং ভাষাকে তাদের "কার্নেল" ইনস্টল করে সমর্থন করে। আরও জানতে তথ্য দেখুন <https://github.com/jupyter/jupyter/wiki/Jupyterkernels>.

১.৫.১ ক্যালকুলেটর হিসেবে আইপিথন ইন্টারেক্ষিভ মোড ব্যবহার করা

সহজ গাণিতিক রাশিগুলি মূল্যায়ন করতে আইপিথন ইন্টারেক্ষিভ মোড ব্যবহার করা যাক।

ইন্টারেক্ষিভ মোডে আইপিথন প্রবেশ করানো

প্রথমে, আপনার সিস্টেমে একটি কমান্ড লাইন উইন্ডো খুলুন:

- macOS-এ, অ্যাপ্লিকেশন ফোল্ডারের ইউটিলিটিস সাবফোল্ডার থেকে একটি টার্মিনাল খুলুন।
- উইন্ডোজে, স্টার্ট মেনু থেকে অ্যানাকোন্ডা কমান্ড প্রস্পেক্ট খুলুন।
- লিনাক্সে, আপনার সিস্টেমের টার্মিনাল বা শেল খুলুন (এটি লিনাক্স বিতরণ অনুসারে পরিবর্তিত হয়)।

কমান্ড লাইন উইন্ডোতে, ipython টাইপ করুন, তারপর Enter (অথবা Return) টিপুন। আপনি নিচের মতো টেক্স্ট দেখতে পাবেন, এটি প্ল্যাটফর্ম এবং IPython সংস্করণ অনুসারে পরিবর্তিত হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

পাইথন ৩.৭.০ | কনডাফোর্জ দ্বারা প্র্যাকেজ করা | (ডিফল্ট, ২০ জানুয়ারী ২০১৯, ১৭:২৪:৫২)

আরও তথ্যের জন্য 'কপিরাইট', 'ক্রেডিট' অথবা 'লাইসেন্স' টাইপ করুন IPython 6.5.0 একটি উন্নত ইন্টারেক্ষিভ পাইথন। ?' টাইপ করুন।

সাহায্যের জন্য.

[1] তে:

"[1]: " লেখাটি একটি প্রস্পেক্ট, যা নির্দেশ করে যে আইপিথন আপনার ইনপুটের জন্য অপেক্ষা করছে। আপনি সাহায্যের জন্য ? টাইপ করতে পারেন অথবা স্লিপেট প্রবেশ করা শুরু করতে পারেন, যেমনটি আপনি কিছুক্ষণের জন্য করবেন।

রাশির মূল্যায়ন

ইন্টারেক্ষিভ মোডে, আপনি রাশিগুলি মূল্যায়ন করতে পারেন:

[1] তে: `45 + 72`

আউট[1]: ১১৭

[2] তে:

৪৫ + ৭২ টাইপ করে এন্টার চাপার পর, আইপিথন স্লিপেটটি পড়ে, মূল্যায়ন করে এবং প্রিন্ট করে।

এর ফলাফল Out[1]। তারপর IPython In [2] প্রস্পট প্রদর্শন করে দেখায় যে এটি আপনার দ্বিতীয় স্লিপেট প্রবেশের জন্য অপেক্ষা করছে। প্রতিটি নতুন স্লিপেটের জন্য, IPython বর্গাকার বন্ধনীতে সংখ্যার সাথে 1 যোগ করে। বইয়ের প্রতিটি In [1] প্রস্পট নির্দেশ করে যে আমরা একটি নতুন ইন্টারেক্টিভ সেশন শুরু করেছি। আমরা সাধারণত একটি অধ্যায়ের প্রতিটি নতুন বিভাগের জন্য এটি করি।

পরবর্তী অধ্যায়ে, আপনি দেখতে পাবেন যে কিছু ক্ষেত্রে Out[] প্রদর্শিত হয় না।

আসুন আরও জটিল একটি রাশি মূল্যায়ন করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[2] তে: $5 * (12.7) / 2$

আউট[2]: 21.75

পাইথন গুণের জন্য তারকাচিহ্ন (*) এবং ভাগের জন্য ফরোয়ার্ড ম্যাশ (/) ব্যবহার করে। গণিতের মতো, বন্ধনী মূল্যায়নের ক্রমকে জোর করে, তাই বন্ধনীযুক্ত রাশি (12.7 4) প্রথমে মূল্যায়ন করে, 8.7 দেয়। এরপর, $5 * 8.7$ মূল্যায়ন করে 43.5 দেয়। তারপর, $43.5 / 2$ মূল্যায়ন করে, ফলাফল 21.75 দেয়, যা IPython Out[2] তে প্রদর্শন করে। পূর্ণ সংখ্যা, যেমন 5, 4 এবং 2, কে পূর্ণসংখ্যা বলা হয়। দশমিক বিন্দু বিশিষ্ট সংখ্যা, যেমন 12.7, 43.5 এবং 21.75, কে ফ্লোটিংপয়েন্ট সংখ্যা বলা হয়।

ইন্টারেক্টিভ মোড থেকে বেরিয়ে আসা হচ্ছে

ইন্টারেক্টিভ মোড থেকে বেরিয়ে আসতে, আপনি যা করতে পারেন:

- বর্তমান In [] প্রস্পটে exit কমান্ডটি টাইপ করুন এবং অবিলম্বে প্রস্থান করতে Enter টিপুন।
- কী সিকোয়েল্স <Ctrl> + d (অথবা <control> + d) টাইপ করুন। এটি "আপনি কি সত্যিই ([y]/n) থেকে বেরিয়ে আসতে চান?" প্রস্পটটি প্রদর্শন করে। y এর চারপাশের বর্গাকার বন্ধনীগুলি নির্দেশ করে যে এটি ডিফল্ট প্রতিক্রিয়া - এন্টার টিপলে ডিফল্ট প্রতিক্রিয়া জমা হবে এবং প্রস্থান হবে।
- <Ctrl> + d (অথবা <control> + d) দুইবার টাইপ করুন (শুধুমাত্র macOS এবং Linux)।

১.৫.২ আইপিথন ইন্টারপ্রেটার ব্যবহার করে একটি পাইথন প্রোগ্রাম কার্যকর করা

এই বিভাগে, আপনি RollDieDynamic.py নামে একটি স্ক্রিপ্ট চালাবেন যা আপনি লিখবেন

hapter 6. .py এক্সেনশনটি নির্দেশ করে যে ফাইলটিতে পাইথন সোর্স কোড রয়েছে। স্ক্রিপ্টটি

RollDieDynamic.py একটি ছয় পার্শ্বযুক্ত ডাই রোলিং সিমুলেট করে। এটি একটি রঙিন অ্যানিমেটেড ডিজুয়ালাইজেশন উপস্থাপন করে যা প্রতিটি ডাই ফেসের ফ্রিকোয়েন্সি গতিশীলভাবে গ্রাফ করে।

এই অধ্যায়ের উদাহরণ ফোন্ডারে পরিবর্তন করা হচ্ছে

আপনি বইয়ের ch01 সোর্সকোড ফোন্ডারে স্ক্রিপ্টটি পাবেন। "বিফোর ইউ বিগিন" বিভাগে আপনি উদাহরণ ফোন্ডারটি আপনার ব্যবহারকারী অ্যাকাউন্টের ডকুমেন্টস ফোন্ডারে এক্সট্রাক্ট করেছেন। প্রতিটি অধ্যায় www.EBooksWorld.ir

সেই অধ্যায়ের সোর্স কোড সম্বলিত একটি ফোল্ডার আছে। ফোল্ডারটির নাম ch##, যেখানে ## হল 01 থেকে 17 পর্যন্ত

একটি দুই অঙ্কের অধ্যায় সংখ্যা। প্রথমে, আপনার সিস্টেমের কমান্ড লাইন উইন্ডোটি খুলুন।

এরপর, ch01 ফোল্ডারে পরিবর্তন করতে cd ("directory পরিবর্তন করুন") কমান্ড ব্যবহার করুন:

- macOS/Linux-এ, cd ~/Documents/examples/ch01 টাইপ করুন, তারপর Enter টিপুন।
- উইন্ডোজে, cd C:\Users\YourAccount\Documents\examples\ch01 টাইপ করুন, তারপর Enter টিপুন।

স্ক্রিপ্টটি কার্যকর করা হচ্ছে

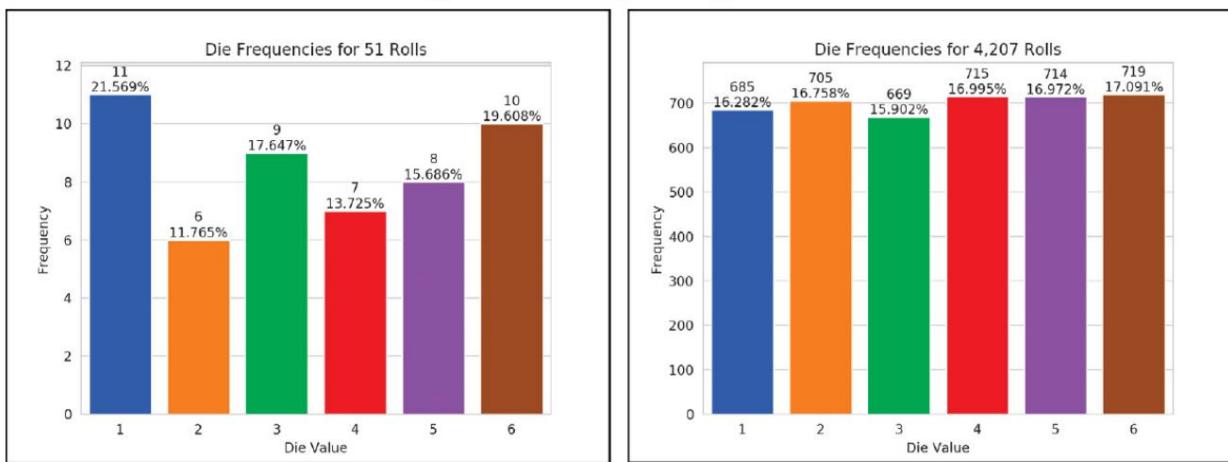
স্ক্রিপ্টটি কার্যকর করতে, কমান্ড লাইনে নিম্নলিখিত কমান্ডটি টাইপ করুন, তারপর এন্টার টিপুন:

```
ipython RollDieDynamic.py 6000 1
```

স্ক্রিপ্টটি একটি উইন্ডো প্রদর্শন করবে, যা ডিজুয়ালাইজেশন দেখাবে। 6000 এবং 1 সংখ্যাটি এই স্ক্রিপ্টটিকে কতবার পাশা ঘুরাতে হবে এবং প্রতিবার কতগুলি পাশা ঘুরাতে হবে তা বলে দেবে। এই ক্ষেত্রে, আমরা একবারে 1টি ডাইয়ের জন্য 6000 বার চার্টটি আপডেট করব।

একটি ছয়পাঞ্চিক ডাইয়ের ক্ষেত্রে, ১ থেকে ৬ মান "সমান সম্ভাবনা" সহকারে হওয়া উচিত - প্রতিটির সম্ভাব্যতা ১/৬ বা প্রায় ১৬.৬৬৭%। যদি আমরা একটি ডাই ৬০০০ বার রোল করি, তাহলে আমরা প্রতিটি ফেসের প্রায় ১০০০টি আশা করব। মুদ্রা ছুঁড়ে মারার মতো, ডাই রোলিং এলোমেলো, তাই কিছু ফেস ১০০০ এর কম, কিছুতে ১০০০ এবং কিছুতে ১০০০ এর বেশি হতে পারে। স্ক্রিপ্টটি কার্যকর করার সময় আমরা নীচের স্ক্রিন ক্যাপচারণালি নিয়েছি। এই স্ক্রিপ্টটি এলোমেলোভাবে তৈরি ডাই মান ব্যবহার করে, তাই আপনার ফলাফল ভিন্ন হবে। মান ১ কে ১০০, ১০০০ এবং ১০০০০ এ পরিবর্তন করে স্ক্রিপ্টটি পরীক্ষা করুন। লক্ষ্য করুন যে ডাই রোলের সংখ্যা যত বেশি হবে, ফিকোয়েলি শূন্য হয়ে ১৬.৬৬৭% হয়ে যাবে। এটি "বৃহৎ সংখ্যার aw" এর একটি ঘটনা।

Roll the dice 6000 times and roll 1 die each time:
ipython RollDieDynamic.py 6000 1



স্ক্রিপ্ট তৈরি করা

সাধারণত, আপনি আপনার পাইথন সোর্স কোডটি এমন একটি এডিটরে তৈরি করেন যা আপনাকে টেক্সট টাইপ করতে সক্ষম করে।

এডিটর ব্যবহার করে, আপনি একটি প্রোগ্রাম টাইপ করেন, প্রয়োজনীয় সংশোধন করেন এবং এটি আপনার কম্পিউটারে সংরক্ষণ করেন।

ইন্টিগ্রেটেড ডেভেলপমেন্ট এনভায়রনমেন্ট (IDEs) এমন সরঞ্জাম সরবরাহ করে যা সমগ্রকে সমর্থন করে

সফটওয়্যার ডেভেলপমেন্ট প্রক্রিয়া, যেমন এডিটর, ডিবাগার, লজিক ক্রিটি সনাত্ত করার জন্য যা প্রোগ্রামগুলিকে ভুলভাবে কার্যকর করে এবং আরও অনেক কিছু। কিছু জনপ্রিয় পাইথন আইডিই-এর মধ্যে রয়েছে স্পাইডার (যা অ্যানাকোন্ডার সাথে আসে), পাইচার্ম এবং ভিজুয়াল স্টুডিও কোড।

মৃত্যুদণ্ড কার্যকর করার সময় যে সমস্যাগুলি দেখা দিতে পারে

প্রোগ্রামগুলি প্রায়শই প্রথম চেষ্টাতেই কাজ করে না। উদাহরণস্বরূপ, একটি এক্সিকিউটিং প্রোগ্রাম শূন্য দিয়ে ভাগ করার চেষ্টা করতে পারে (পাইথনে একটি অবৈধ অপারেশন)। এর ফলে প্রোগ্রামটি একটি ক্রিটি বার্তা প্রদর্শন করবে। যদি এটি একটি স্ক্রিপ্ট ঘটে থাকে, তাহলে আপনাকে সম্পাদকের কাছে ফিরে যেতে হবে, প্রয়োজনীয় সংশোধন করতে হবে এবং সংশোধনগুলি সমস্যা (গুলি) সমাধান করেছে কিনা তা নির্ধারণ করতে স্ক্রিপ্টটি পুনরায় কার্যকর করতে হবে।

প্রোগ্রাম চালানোর সময় শূন্য দিয়ে ভাগ করার মতো ক্রিটি দেখা দেয়, তাই এগুলোকে **রানটাইম ক্রিটি** বলা হয়। মারাত্মক রানটাইম ক্রিটির কারণে প্রোগ্রামগুলি তাদের কাজ সফলভাবে সম্পন্ন না করেই তাৎক্ষণিকভাবে বন্ধ হয়ে যায়। **নন-ফ্যাটাল রানটাইম ক্রিটি** প্রোগ্রামগুলিকে সম্পূর্ণরূপে চালানোর অনুমতি দেয়, প্রায়শই ভুল ফলাফল দেয়।

১.৫.৩ জুপিটার নোটবুকে কোড লেখা এবং কার্যকর করা

"বিফোর ইউ বিগিন" বিভাগে আপনি যে অ্যানাকোন্ডা পাইথন ডিস্ট্রিবিউশনটি ইনস্টল করেছেন তা **জুপিটার নোটবুকের সাথে আসে - একটি ইন্টারেক্টিভ, ব্রাউজার-ভিত্তিক পরিবেশ** যেখানে আপনি কোড লিখতে এবং কার্যকর করতে পারেন এবং কোডটিকে টেক্সট, ছবি এবং ভিডিওর সাথে মিশ্রিত করতে পারেন। জুপিটার নোটবুকগুলি বিশেষ করে ডেটাসায়েন্স কমিউনিটিতে এবং সাধারণভাবে বৃহত্তর বৈজ্ঞানিক কমিউনিটিতে ব্যাপকভাবে ব্যবহৃত হয়। পাইথন-ভিত্তিক ডেটা বিশ্লেষণ অধ্যয়ন এবং পুনরুৎপাদনযোগ্যভাবে তাদের ফলাফল যোগাযোগের জন্য এগুলি পছন্দের মাধ্যম। জুপিটার নোটবুক পরিবেশ ক্রমবর্ধমান সংখ্যক প্রোগ্রামিং ভাষা সমর্থন করে।

আপনার সুবিধার জন্য, বইটির সমস্ত সোর্স কোড Jupyter Notebooks-এ দেওয়া আছে যা আপনি সহজেই লোড এবং এক্সিকিউট করতে পারবেন। এই বিভাগে, আপনি **JupyterLab** ইন্টারেফেস ব্যবহার করবেন, যা আপনাকে আপনার নোটবুক ফাইল এবং আপনার নোটবুক ব্যবহার করে এমন অন্যান্য ফাইল (যেমন ছবি এবং ভিডিও) পরিচালনা করতে সক্ষম করে। আপনি দেখতে পাবেন, JupyterLab কোড লেখা, এক্সিকিউট করা, ফলাফল দেখা, কোড পরিবর্তন করা এবং আবার এক্সিকিউট করাও সুবিধাজনক করে তোলে।

আপনি দেখতে পাবেন যে জুপিটার নোটবুকে কোডিং করা আইপিথনের সাথে কাজ করার মতোই - আসলে, জুপিটার নোটবুকগুলি ডিফল্টরূপে আইপিথন ব্যবহার করে। এই বিভাগে, আপনি একটি নোটবুক তৈরি করবেন, এতে বিভাগ 1.5.1 থেকে কোড যুক্ত করবেন এবং সেই কোডটি কার্যকর করবেন।

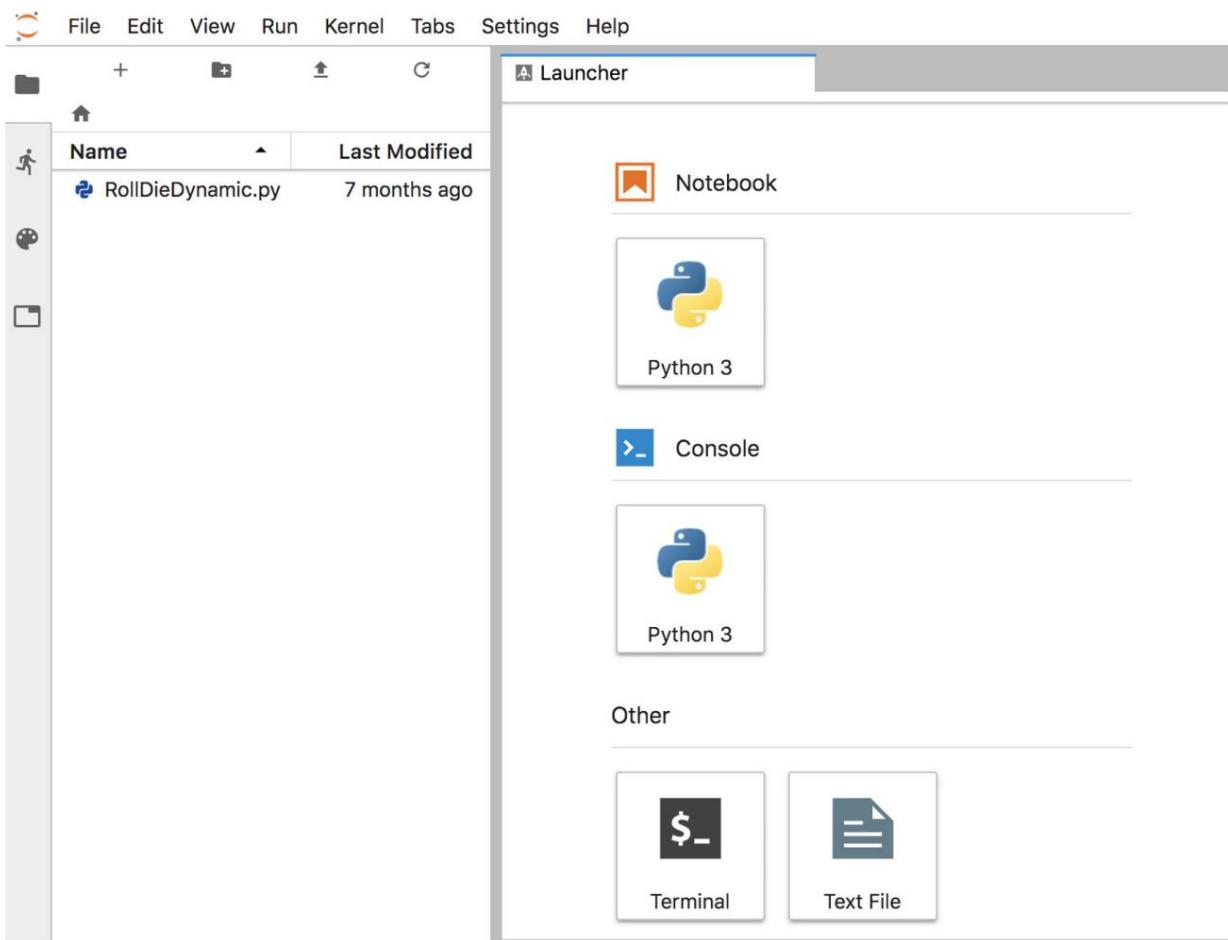
আপনার ব্রাউজারে JupyterLab খুলছে

JupyterLab খুলতে, আপনার টার্মিনাল, শেল বা অ্যানাকোন্ডা কমান্ড প্রস্পেক্ট (যেমন ection 1.5.2 তে) ch01 examples ফোল্ডারে যান, নিম্নলিখিত কমান্ডটি টাইপ করুন, তারপর Enter (অথবা Return) টিপ্পন:

জুপিটার ল্যাব



JupyterLab ইন্টারফেসের বাম দিকে:



Jupyter Notebook সার্ভার আপনাকে আপনার ওয়েব ব্রাউজারে Jupyter Notebooks লোড এবং চালাতে সক্ষম করে।

JupyterLab Files ট্যাব থেকে, আপনি উইন্ডোর ডান দিকে যেখানে লঞ্চার ট্যাবটি বর্তমানে প্রদর্শিত হচ্ছে সেখানে ফাইলগুলি খুলতে ডাবল ক্লিক করতে পারেন। আপনার খোলা প্রতিটি ফাইল উইন্ডোর এই অংশে একটি পৃথক ট্যাব হিসাবে প্রদর্শিত হবে। যদি আপনি ডুলবশত আপনার ব্রাউজারটি বন্ধ করে দেন, তাহলে আপনার ওয়েব ব্রাউজারে নিম্নলিখিত ঠিকানাটি প্রবেশ করে JupyterLab পুনরায় খুলতে পারেন।

<http://লোকালহোস্ট:8888/ল্যাব>

একটি নতুন জুপিটার নোটবুক তৈরি করা হচ্ছে

Notebook এর অধীনে Launcher ট্যাবে, Untitled.ipynb নামে একটি নতুন Jupyter Notebook তৈরি করতে Python 3 বোতামে ক্লিক করুন যেখানে আপনি Python 3 কোড প্রবেশ এবং কার্যকর করতে পারবেন। ফাইল এক্সটেনশন .ipynb হল IPython Notebook এর সংক্ষিপ্ত রূপ—Jupyter এর আসল নাম। নোটবই।

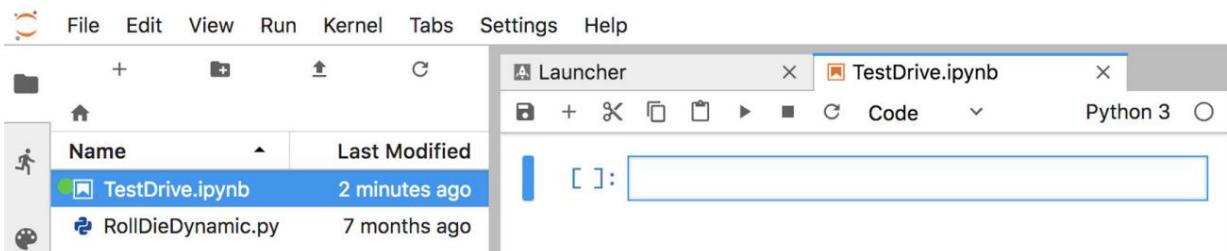
নোটবুকের নাম পরিবর্তন করা হচ্ছে

Untitled.ipynb এর নাম পরিবর্তন করে TestDrive.ipynb করুন:

১. Untitled.ipynb ট্যাবে ডান ক্লিক করুন এবং Rename Notebook নির্বাচন করুন।

২. নামটি TestDrive.ipynb এ পরিবর্তন করুন এবং RENAME এ ক্লিক করুন।

JupyterLab এর উপরের অংশটি এখন নিম্নরূপ প্রদর্শিত হবে:

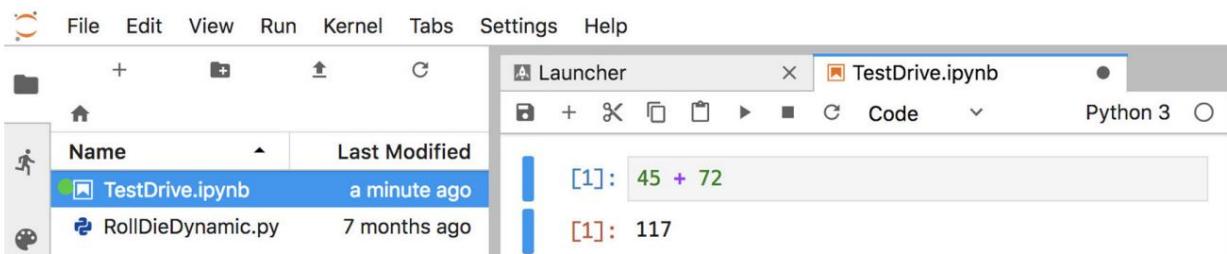


একটি অভিব্যক্তি মূল্যায়ন করা

একটি নোটবুকের কাজের একক হল এমন একটি ঘর যেখানে আপনি কোড স্লিপেট লিখতে পারেন। ডিফল্টেরপে, একটি নতুন নোটবুকে একটি ঘর থাকে—TestDrive.ipynb নোটবুকের আয়তক্ষেত্র—তবে আপনি আরও ঘোগ করতে পারেন। ঘরের বাম দিকে, []: স্বরলিপিটি হল যেখানে জুপিটার নোটবুক আপনি ঘরটি কার্যকর করার পরে ঘরের স্লিপেট নম্বর প্রদর্শন করবে। ঘরে ক্লিক করুন, তারপর এক্সপ্রেশনটি টাইপ করুন।

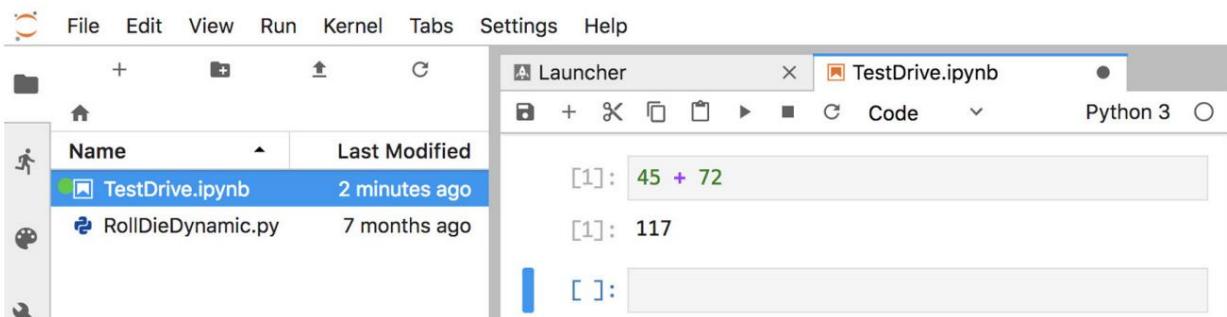
৪৫ + ৭২

বর্তমান সেলের কোডটি কার্যকর করতে, Ctrl + Enter (অথবা control + Enter) টাইপ করুন। JupyterLab আইপিথনে কোডটি কার্যকর করে, তারপর সেলের নীচে ফলাফল প্রদর্শন করে:



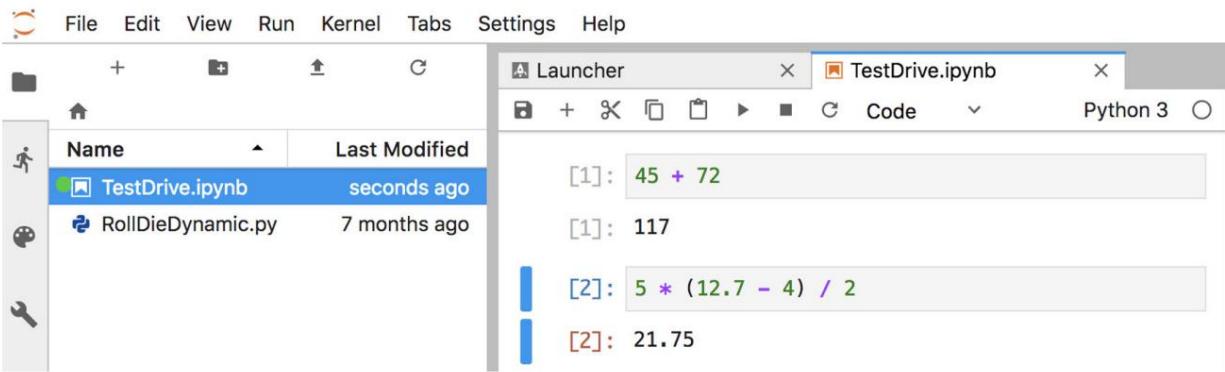
আরেকটি কোষ ঘোগ করা এবং কার্যকর করা

আসুন আরও জটিল একটি রাশির মূল্যায়ন করি। প্রথমে, নোটবুকের প্রথম ঘরের উপরে টুলবারে + বোতামটি ক্লিক করুন—এটি বর্তমান ঘরের নীচে একটি নতুন ঘর যুক্ত করে:



নতুন ঘরে ক্লিক করুন, তারপর এক্সপ্রেশনটি টাইপ করুন

এবং Ctrl + Enter (অথবা control + Enter) টাইপ করে সেলটি এক্সিকিউট করুন:



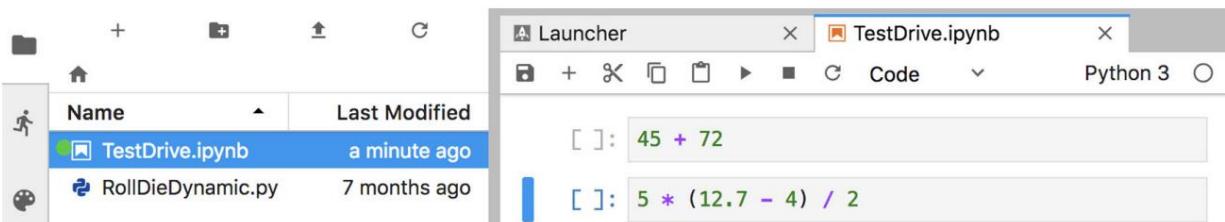
নোটবুক সংরক্ষণ করা হচ্ছে

যদি আপনার নোটবুকে অসংরক্ষিত পরিবর্তন থাকে, তাহলে নোটবুকের ট্যাবে X পরিবর্তন হবে। নোটবুকটি সংরক্ষণ করতে, JupyterLab-এ ফাইল মেনু নির্বাচন করুন (আপনার ব্রাউজারের উইন্ডোর উপরে নয়),
তারপর সেভ নোটবুক নির্বাচন করুন।

প্রতিটি অধ্যায়ের উদাহরণ সহ নোটবুক প্রদান করা হয়েছে

আপনার সুবিধার্থে, প্রতিটি অধ্যায়ের উদাহরণগুলি তাদের আউটপুট ছাড়াই কার্যকর করার জন্য প্রস্তুত নোটবুক হিসাবে সরবরাহ করা হয়েছে।
এটি আপনাকে নিপেট-বাই-নিপেট দিয়ে কাজ করতে এবং প্রতিটি নিপেট কার্যকর করার সময় আউটপুটগুলি প্রদর্শিত হতে দেখতে সক্ষম করে।

একটি বিদ্যমান নোটবুক কীভাবে লোড করতে হয় এবং এর সেলগুলি কীভাবে কার্যকর করতে হয় তা দেখানোর জন্য, আসুন TestDrive.ipynb নোটবুকটি রিসেট করি যাতে এর আউটপুট এবং নিপেট নম্বরগুলি সরানো যায়। এটি পরবর্তী অধ্যায়গুলির উদাহরণগুলির জন্য আমরা যে নোটবুকগুলি প্রদান করি তার মতো অবস্থায় এটি ফিরিয়ে আনবে। কার্নেল মেনু থেকে Restart Kernel এবং Clear All Outputs... নির্বাচন করুন, তারপর RESTART বোতামে ক্লিক করুন। যখনই আপনি একটি নোটবুকের নিপেটগুলি পুনরায় কার্যকর করতে চান তখন পূর্ববর্তী কমান্ডটিও সহায়ক। নোটবুকটি এখন নিম্নরূপ প্রদর্শিত হবে:



ফাইল মেনু থেকে, সেভ নোটবুক নির্বাচন করুন, তারপর TestDrive.ipynb ট্যাবের X-এ ক্লিক করুন।

নোটবুক বন্ধ করার বোতাম।

একটি বিদ্যমান নোটবুক খোলা এবং কার্যকর করা

যখন আপনি একটি নির্দিষ্ট অধ্যায়ের উদাহরণ ফোল্ডার থেকে JupyterLab চালু করবেন, তখন আপনি সেই ফোল্ডার বা এর যেকোনো সাবফোল্ডার থেকে নোটবুক খুলতে সক্ষম হবেন। একবার আপনি একটি নির্দিষ্ট নোটবুক খুঁজে পেলে, এটি খুলতে ডাবল ক্লিক করুন। এখন আবার TestDrive.ipynb নোটবুকটি খুলুন। একবার একটি নোটবুক খোলা হয়ে গেলে, আপনি প্রতিটি সেল পৃথকভাবে কার্যকর করতে পারেন,
যেমনটি আপনি এই বিভাগে আগে করেছিলেন, অথবা আপনি একবারে সম্পূর্ণ নোটবুকটি কার্যকর করতে পারেন। এটি করার জন্য, Run মেনু থেকে Run All নির্বাচন করুন।

কোষ। নেটুরুকটি কোষগুলিকে ক্রমানুসারে কার্যকর করবে, প্রতিটি কোষের আউটপুট তার নীচে প্রদর্শন করবে কোষ।

জুপিটারল্যাব বন্ধ করা হচ্ছে

JupyterLab এর কাজ শেষ হলে, আপনি এর ব্রাউজার ট্যাবটি বন্ধ করতে পারেন, তারপর Terminal, shell অথবা Anaconda Command Prompt যেখান থেকে আপনি JupyterLab চালু করেছেন, সেখানে Ctrl + c (অথবা control + c) দ্বারা টাইপ করুন।

জুপিটারল্যাব টিপস

JupyterLab-এ কাজ করার সময়, আপনি এই টিপসগুলি সহায়ক বলে মনে করতে পারেন:

- যদি আপনার অনেক স্লিপেট প্রবেশ করানো এবং চালানোর প্রয়োজন হয়, তাহলে আপনি বর্তমান সেলটি কার্যকর করতে পারেন এবং Ctrl + Enter (অথবা control + Enter) এর পরিবর্তে Shift + Enter টাইপ করে তার নীচে একটি নতুন সেল ঘূর্ণ করতে পারেন।
- পরবর্তী অধ্যায়গুলিতে প্রবেশ করার সাথে সাথে, Jupyter Notebooks-এ আপনি যে কিছু স্লিপেট লিখবেন তাতে অনেক লাইন কোড থাকবে। প্রতিটি কক্ষের মধ্যে লাইন নম্বর প্রদর্শন করতে, JupyterLab-এর View মেনু থেকে Show line numbers নির্বাচন করুন।

JupyterLab এর সাথে কাজ করার বিষয়ে আরও তথ্য

JupyterLab-এর আরও অনেক বৈশিষ্ট্য রয়েছে যা আপনার জন্য সহায়ক হবে। আমরা আপনাকে Jupyter টিমের JupyterLab-এর ভূমিকাটি এখানে পড়ার পরামর্শ দিচ্ছি:

<https://jupyterlab.readthedocs.io/en/stable/index.html>

দ্রুত সারসংক্ষেপের জন্য, GETTING STARTED-এর অধীনে "Overview"-এ ক্লিক করুন। এছাড়াও, USER GUIDE-এর অধীনে "The JupyterLab Interface", "Working with Files", "Text Editor" এবং "Notebooks"-এর ভূমিকা পড়ুন এবং আরও অনেক অতিরিক্ত বৈশিষ্ট্য পড়ুন।

১.৬ বন্ধুর মেঘ এবং ইন্টারনেট

১.৬.১ মেঘ

আজকাল আরও বেশি সংখ্যক কম্পিউটিং "ক্লাউডে" করা হচ্ছে—অর্থাৎ, বিশ্বব্যাপী ইন্টারনেট জুড়ে বিতরণ করা হচ্ছে। আপনি প্রতিদিন যে অনেক অ্যাপ ব্যবহার করেন তা ক্লাউডভিত্তিক পরিষেবার উপর নির্ভরশীল যা বিশাল কম্পিউটিং রিসোর্স (কম্পিউটার, প্রসেসর, মেমরি, ডিস্ক ড্রাইভ ইত্যাদি) এবং ডাটাবেস ব্যবহার করে যা ইন্টারনেটের মাধ্যমে একে অপরের সাথে এবং আপনার ব্যবহৃত অ্যাপগুলির সাথে যোগাযোগ করে।

যে পরিষেবা ইন্টারনেটের মাধ্যমে নিজের অ্যাক্তোস প্রদান করে তাকে ওয়েব পরিষেবা বলা হয়। আপনি দেখতে পাবেন, পাইথনে ক্লাউডভিত্তিক পরিষেবা ব্যবহার করা প্রায়শই একটি সফ্টওয়্যার অবজেক্ট তৈরি করা এবং এর সাথে ইন্টারঅ্যাক্ট করার মতোই সহজ। সেই অবজেক্টটি তখন এমন ওয়েব পরিষেবা ব্যবহার করে যা আপনার পক্ষ থেকে ক্লাউডের সাথে সংযুক্ত হয়।

পর্ব ১১-৬ এর উদাহরণ জুড়ে, আপনি অনেক ক্লাউডভিত্তিক পরিষেবার সাথে কাজ করবেন:

- ১২ এবং ৬ নম্বর পর্বে, আপনি নির্দিষ্ট টুইটার ব্যবহারকারীদের সম্পর্কে তথ্য পেতে, গত সাত দিনের টুইটগুলি অনুসন্ধান করতে এবং টুইটগুলির স্ট্রিমগুলি প্রাপ্ত করতে টুইটারের ওয়েব পরিষেবাগুলি (পাইথন লাইব্রেরি টুইপির মাধ্যমে)
 - ব্যবহার করবেন - অর্থাৎ, রিয়েল টাইমে।
-
- অধ্যায় ১১ এবং ২-এ, আপনি ভাষার মধ্যে টেক্সট অনুবাদ করার জন্য পাইথন লাইব্রেরি টেক্সটের ব্যবহার করবেন। পর্দার আড়ালে, টেক্সটের সেই অনুবাদগুলি সম্পাদন করার জন্য গুগল ট্রান্সলেট ওয়েব পরিষেবা ব্যবহার করে।
-
- হ্যাপ্টার ১৩-এ, আপনি আইবিএম ওয়াটসনের টেক্সট টু স্পিচ, স্পিচ টু টেক্সট এবং ট্রান্সলেট পরিষেবা ব্যবহার করবেন। আপনি একটি ভ্রমণকারী সহকারী অনুবাদ অ্যাপ বাস্তবায়ন করবেন যা আপনাকে ইংরেজিতে একটি প্রশ্ন বলতে, বক্তৃতাকে টেক্সটে ট্রান্সক্রাইব করতে, টেক্সটকে স্প্যানিশ ভাষায় অনুবাদ করতে এবং স্প্যানিশ টেক্সট বলতে সক্ষম করে। এরপর অ্যাপটি আপনাকে স্প্যানিশ উভের বলতে দেয় (যদি আপনি স্প্যানিশ না বলতে পারেন, আমরা একটি অডিও ফাইল সরবরাহ করি যা আপনি ব্যবহার করতে পারেন), বক্তৃতাকে টেক্সটে ট্রান্সক্রাইব করে, টেক্সটকে ইংরেজিতে অনুবাদ করে এবং ইংরেজি প্রতিক্রিয়া বলতে। আইবিএম ওয়াটসন ডেমোর মাধ্যমে, আপনি হ্যাপ্টার ১৩-তে ওয়াটসনের আরও অনেক ক্লাউডভিত্তিক পরিষেবা নিয়ে পরীক্ষা-নিরীক্ষা করতে পারবেন।
-
- পর্ব ১৬-তে, আপনি Apache Hadoop এবং Spark ব্যবহার করে বিগডেটা অ্যাপ্লিকেশন বাস্তবায়নের সময় Microsoft Azure-এর HDInsight পরিষেবা এবং অন্যান্য Azure ওয়েব পরিষেবাগুলির সাথে কাজ করবেন। Azure হল Microsoft-এর ক্লাউড-ভিত্তিক পরিষেবাগুলির একটি সেট।
-
- ১৬তম পর্বে, আপনি Dweet.io ওয়েব পরিষেবা ব্যবহার করে একটি ইন্টারনেট-সংযুক্ত থার্মোস্ট্যাট তৈরি করবেন যা অনলাইনে তাপমাত্রার রিডিং প্রকাশ করে। আপনি একটি ওয়েব-ভিত্তিক পরিষেবাও ব্যবহার করে একটি "ড্যাশবোর্ড" তৈরি করবেন যা সময়ের সাথে সাথে তাপমাত্রার রিডিং কল্পনা করে এবং তাপমাত্রা খুব কম বা খুব বেশি হলে আপনাকে সতর্ক করে।
-
- পর্ব ১৬-তে, আপনি PubNub ওয়েব পরিষেবা থেকে লাইভ সেলর ডেটার একটি সিমুলেটেড স্ট্রিম কল্পনা করার জন্য একটি ওয়েবভিত্তিক ড্যাশবোর্ড ব্যবহার করবেন। আপনি একটি পাইথন অ্যাপও তৈরি করবেন যা লাইভ স্টক মূল্য পরিবর্তনের একটি PubNub সিমুলেটেড স্ট্রিম কল্পনা করবে।

বেশিরভাগ ক্ষেত্রে, আপনি পাইথন অবজেক্ট তৈরি করবেন যা আপনার পক্ষ থেকে ওয়েব পরিষেবাগুলির সাথে ইন্টারঅ্যাক্ট করবে, ইন্টারনেটের মাধ্যমে এই পরিষেবাগুলি কীভাবে অ্যাক্সেস করবেন তার বিশদ গোপন করবে।

ম্যাশআপস

ম্যাশআপের অ্যাপ্লিকেশন ডেভেলপমেন্ট পদ্ধতি আপনাকে (প্রায়শই বিনামূল্যে) পরিপূরক ওয়েব পরিষেবা এবং অন্যান্য ধরণের তথ্য ফিড একত্রিত করে দ্রুত শক্তিশালী সফ্টওয়্যার অ্যাপ্লিকেশন তৈরি করতে সক্ষম করে—যেমনটি আপনি আমাদের IBM ওয়াটসন ট্রান্সলার্স অ্যাসিস্ট্যান্ট ট্রান্সলেশন অ্যাপে করবেন। প্রথম ম্যাশআপগুলির মধ্যে একটি হল রিয়েল এস্টেট তালিকা একত্রিত করা।

<http://www.craigslist.org> গুগল ম্যাপের ম্যাপিং ক্ষমতা সহ একটি নির্দিষ্ট এলাকায় বিক্রয় বা ভাড়ার জন্য বাড়ির অবস্থান দেখানো মানচিত্র।

প্রোগ্রামেবলওয়েব (<http://www.programmableweb.com/>) ২০,৭৫০টিরও বেশি ওয়েব পরিষেবা এবং প্রায় ৮,০০০ ম্যাশআপের একটি ডিরেক্টরি প্রদান করে। তারা ওয়েব পরিষেবাগুলির সাথে কাজ করার এবং আপনার নিজস্ব ম্যাশআপ তৈরি করার জন্য কীভাবে নির্দেশিকা এবং নমুনা কোডও প্রদান করে। তাদের ওয়েবসাইট অনুসারে, সর্বাধিক ব্যবহৃত কিছু ওয়েব পরিষেবা হল ফেসবুক, গুগল ম্যাপস, টুইটার এবং

১.৬.২ ইন্টারনেট অফ থিংস

ইন্টারনেট এখন আর কেবল কম্পিউটারের একটি নেটওয়ার্ক নয় - এটি একটি ইন্টারনেট অফ থিংস (IoT)। জিনিস হল এমন যেকোনো বস্তু যার একটি IP ঠিকানা রয়েছে এবং ইন্টারনেটের মাধ্যমে স্বয়ংক্রিয়ভাবে ডেটা প্রেরণ এবং কিছু ক্ষেত্রে গ্রহণ করার ক্ষমতা রয়েছে। এই ধরণের জিনিসগুলির মধ্যে রয়েছে:

- টোল দেওয়ার জন্য ট্রান্সপন্ডার সহ একটি গাড়ি,
- গ্যারেজে পার্কিং স্পেসের প্রাপ্যতার জন্য মনিটর,
- একজন মানুষের শরীরে বসানো হার্ট মনিটর,
- জলের গুণমান মনিটর,
- একটি স্মার্ট মিটার যা শক্তির ব্যবহার রিপোর্ট করে,
- বিকিরণ সনাক্তকারী যন্ত্র,
- গুদামে আইটেম ট্র্যাকার,
- আপনার গতিবিধি এবং অবস্থান ট্র্যাক করতে পারে এমন মোবাইল অ্যাপ,
- স্মার্ট থার্মোস্ট্যাট যা আবহাওয়ার পূর্বাভাস এবং বাড়ির কার্যকলাপের উপর ভিত্তি করে ঘরের তাপমাত্রা সামঞ্জস্য করে, এবং
- বুদ্ধিমান গৃহস্থালী যন্ত্রপাতি।

statista.com এর তথ্য অনুযায়ী, বর্তমানে ২৩ বিলিয়নেরও বেশি IoT ডিভাইস ব্যবহার করা হচ্ছে এবং ২০২৫ সালে ৭৫ বিলিয়নেরও বেশি IoT ডিভাইস ব্যবহার করা হতে পারে।

২

[২ https://www.statista.com/statistics/471264/iotnumberofconnected-devices-worldwide/](https://www.statista.com/statistics/471264/iotnumberofconnected-devices-worldwide/)

১.৭ বড় তথ্য কত বড়?

কম্পিউটার বিজ্ঞানী এবং ডেটা বিজ্ঞানীদের কাছে, ডেটা এখন প্রোগ্রাম লেখার মতোই গুরুত্বপূর্ণ।

আইবিএমের মতে, প্রতিদিন প্রায় ২.৫ কুইন্টিলিয়ন বাইট (২.৫ এক্সাবাইট) ডেটা তৈরি হয় এবং বিশ্বের ৯০% ডেটা গত দুই বছরে তৈরি হয়েছে। আইডিসির মতে, ২০২৫^৩ সালের মধ্যে বিশ্বব্যাপী ডেটা সরবরাহ বার্ষিক ১৭৫ জেটাবাইট (১৭৫ ট্রিলিয়ন গিগাবাইট বা ১৭৫ বিলিয়ন টেরাবাইট) পৌঁছাবে। বিভিন্ন জনপ্রিয় ডেটার নিম্নলিখিত উদাহরণগুলি বিবেচনা করুন।

৪

ব্যবস্থা।

[৩ https://www.ibm.com/blogs/watson/2016/06/welcometotheworldofdata/](https://www.ibm.com/blogs/watson/2016/06/welcometotheworldofdata/)

৪.

^৮ <https://public.dhe.ibm.com/common/ssi/ecm/wr/en/wrl12345usen/watson-r12345usen20170719.pdf>.

গ্রাহকসংশ্লিষ্টতাওয়ার্টসনবিপণনকাগজপত্রএবংপ্রতিবেদন-

১২৫ সালের মধ্যে ৭৫জেটাবাইটস অফডাটাওয়ার্ডওয়াইড.html।।

ইগাবাইট (এমবি)

এক মেগাবাইট প্রায় দশ লক্ষ (আসলে ২) বাইট। আমরা প্রতিদিন ^{২০} ফাইলগুলি ব্যবহার করি তার অনেকের জন্য এক বা একাধিক এমবি স্টোরেজ প্রয়োজন। কিছু উদাহরণের মধ্যে রয়েছে:

- MP3 অডিও ফাইল—উচ্চমানের MP3 গুলি প্রতি মিনিটে ১ থেকে ২.৪ মেগাবাইট পর্যন্ত হতে পারে।

^৯ <https://www.audiomountain.com/tech/audiofilesize.html>.

- ছবি—ডিজিটাল ক্যামেরায তোলা JPEG ফর্ম্যাটের ছবিগুলির জন্য প্রতি ছবিতে প্রায় ৮ থেকে ১০ মেগাবাইট প্রয়োজন হতে পারে।

- ভিডিও—স্মার্টফোনের ক্যামেরা বিভিন্ন রেজোলিউশনে ভিডিও রেকর্ড করতে পারে। প্রতি মিনিটের ভিডিওর জন্য অনেক মেগাবাইট স্টোরেজ প্রয়োজন হতে পারে। উদাহরণস্বরূপ, আমাদের আইফোনের ক্যামেরা সেটিংস অ্যাপ জানিয়েছে যে ৩০ ফ্রেম প্রতি সেকেন্ডে (FPS) ১০৮০p ভিডিওর জন্য ১৩০ মেগাবাইট/মিনিট প্রয়োজন এবং ৩০ FPS এ ৪K ভিডিওর জন্য ৩৫০ মেগাবাইট/মিনিট প্রয়োজন।

গিগাবাইট (জিবি)

এক গিগাবাইট প্রায় ১০০০ মেগাবাইট (আসলে ২ বাইট)। একটি ডুয়াললেয়ের ডিভিডি ৮.৫ গিগাবাইট পর্যন্ত স্টোর করতে পারে।

^{১০}, যার অনুবাদ হল:

^{১১} <https://en.wikipedia.org/wiki/DVD>

- ১৪১ ঘন্টার MP3 অডিও,
- একটি ১৬ মেগাপিক্সেল ক্যামেরা থেকে প্রায় ১০০০টি ছবি,
- ৩০ FPS এ আনুমানিক ৭.৭ মিনিটের ১০৮০p ভিডিও, অথবা
- ৩০ FPS এ প্রায় ২.৮৫ মিনিটের ৪K ভিডিও।

বর্তমান সর্বোচ্চ ক্ষমতাসম্পর্ক আল্ট্রা এইচডি বুরে ডিস্ক ১০০ গিগাবাইট পর্যন্ত ভিডিও সংরক্ষণ করতে পারে।
একটি 4K মুভি স্ট্রিমিং প্রতি ঘন্টায় 7 থেকে 10 GB পর্যন্ত ব্যবহার করতে পারে (অত্যন্ত সংকুচিত)।

^{১২} https://en.wikipedia.org/wiki/Ultra_HD_Blu-ray

টেরাবাইট (টিবি)

এক টেরাবাইট প্রায় ১০০০ গিগাবাইট (আসলে ২ বাইট)। ডেস্কটপের ^{১০} জন্য সাম্প্রতিক ডিস্ক ড্রাইভ

www.EBooksWorld.ir

কম্পিউটারগুলি ১৫ টির পর্যন্ত আকারে আসে, যাঁ এর সমতুল্য:

^১ <https://www.zdnet.com/article/worldsbiggestharddrivemeet->

এস্টারডিজিটালস ১৫টি বিমনস্টার।

- প্রায় ২৮ বছরের MP3 অডিও,
- ১৬ মেগাপিক্সেল ক্যামেরা থেকে প্রায় ১.৬৮ মিলিয়ন ছবি,
- ৩০ FPS এ প্রায় ২২৬ ঘন্টার ১০৮০p ভিডিও এবং
- ৩০ FPS এ প্রায় ৮৪ ঘন্টার ৪K ভিডিও।

নিষ্পাস ডেটার এখন সবচেয়ে বড় সলিডস্টেট ড্রাইভ (SSD) রয়েছে যার ক্ষমতা ১০০ টেরাবাইট, যা ৬.৬৭ টেরাবাইট ধারণ করতে পারে।

উপরে তালিকাভুক্ত অডিও, ছবি এবং ভিডিওর ১৫ টেরাবাইট উদাহরণের চেয়ে গুণ বেশি।

^০ <https://www.cinema5d.com/nimbusdata100tbssdworldslargestssd/>.

পেটাবাইট, এক্সাবাইট এবং জেটাবাইট

অনলাইনে প্রায় চার বিলিয়ন মানুষ প্রতিদিন প্রায় ২.৫ কুইন্টিলিয়ন বাইট ডেটা তৈরি করে - অর্থাৎ ২৫০০ পেটাবাইট (প্রতি পেটাবাইট প্রায় ১০০০ টেরাবাইট) অথবা ২.৫ এক্সাবাইট (প্রতি এক্সাবাইট প্রায় ১০০০ পেটাবাইট)। মার্চ ২০১৬ অ্যানালিটিক্স উইকের একটি নিরবন্ধ অনুসারে, পাঁচ বছরের মধ্যে ৫০ বিলিয়নেরও বেশি ডিভাইস ইন্টারনেটের সাথে সংযুক্ত থাকবে (যার বেশিরভাগই ইন্টারনেট অফ থিংসের মাধ্যমে, যা আমরা ১.৬.২ এবং ৬.৮ অনুচ্ছেদে আলোচনা করেছি) এবং ২০২০ সালের মধ্যে আমরা গ্রহের প্রতিটি ব্যক্তির জন্য প্রতি সেকেন্ডে ১.৭ মেগাবাইট নতুন ডেটা তৈরি করব।

আজকের সংখ্যা অনুসারে (প্রায় ৭.৭ বিলিয়ন মানুষ), অর্থাৎ প্রায়

^১ <https://public.dhe.ibm.com/common/ssi/ecm/wr/en/wrl12345usen/watson->

গ্রাহকসংশ্লিষ্টাওয়ার্টসনবিপণনকাগজপত্র এবং প্রতিবেদন-

r12345usen20170719.pdf.

^২ <https://analyticsweek.com/content/bigdatafacts/>.

^৩ https://en.wikipedia.org/wiki/World_population.

- প্রতি সেকেন্ডে ১৩ পেটাবাইট নতুন ডেটা,
- প্রতি মিনিটে ৭৮০ পেটাবাইট,
- প্রতি ঘন্টায় ৪৬,৮০০ পেটাবাইট (৪৬.৮ এক্সাবাইট) এবং
- প্রতিদিন ১,১২৩ এক্সাবাইট—যা প্রতিদিন ১.১২৩ জেটাবাইট (ZB) (প্রতিটি জেটাবাইট প্রায় ১০০০ এক্সাবাইট)।

অতিরিক্ত বিগ-ডেটা পরিসংখ্যান

বিগ ডেটার একটি বিনোদনমূলক রিয়েলটাইম ধারণার জন্য, দেখুন

<https://www.internetlivestats.com>, বিভিন্ন পরিসংখ্যান সহ, সংখ্যা সহ

আজ পর্যন্ত

- গুগল সার্চ।
- টুইট।
- ইউটিউবে দেখা ভিডিও।
- ছবিগুলো ইনস্টাগ্রামে আপলোড করা হয়েছে।

আরও তথ্যের জন্য আপনি প্রতিটি পরিসংখ্যানে ক্লিক করতে পারেন। উদাহরণস্বরূপ, তারা বলেছে যে ২০১৮ সালে ২৫০ বিলিয়নেরও বেশি টুইট পাঠানো হয়েছিল।

বিগডেটা সম্পর্কিত আরও কিছু আকর্ষণীয় তথ্য:

- প্রতি ঘন্টায়, ইউটিউব ব্যবহারকারীরা ২৪,০০০ ঘন্টা ভিডিও আপলোড করে, এবং প্রায় ১ বিলিয়ন ঘন্টা ইউটিউবে প্রতিদিন কত ভিডিও দেখা হয়।^৮

^৮ <https://www.brandwatch.com/blog/youtubestats/>

- প্রতি সেকেন্ডে, ৫১,৭৭৩ জিবি (অথবা ৫১.৭৭৩ টিবি) ইন্টারনেট ট্র্যাফিক, ৭৮৯৪টি টুইট পাঠানো, ৬৪,৩৩২টি গুগল অনুসন্ধান এবং ৭২,০২৯টি ইউটিউব ভিডিও দেখা হয়।^৯

^৯ <https://www.internetlivestats.com/onesecond/>.

- ফেসবুকে প্রতিদিন ৮০ কোটি "লাইক" হয়, ৬ কোটি ইমোজি পাঠানো হয় এবং সাইটটি শুরু হওয়ার পর থেকে ২.৫^{১০} ট্রিলিয়নেরও বেশি ফেসবুক পোস্টের মধ্যে দুই বিলিয়নেরও বেশি অনুসন্ধান করা হয়েছে।^{১১}

^{১০} <https://newsroom.fb.com/news/2017/06/twobillionpeoplecoming-on-globe/>

^{১১} <https://mashable.com/2017/07/17/facebookworldemojiday/>.

^{১২} <https://techcrunch.com/2016/07/27/facebookwillmakeyoustalk/>.

- ২০১৭ সালের জুন মাসে, প্ল্যানেটের সিইও উইল মার্শাল বলেছিলেন যে তাদের কোম্পানির ১৪২টি উপগ্রহ রয়েছে যা প্রতিদিন একবার পুরো গ্রহের স্থলভাগের ছবি তোলে। তারা প্রতিদিন দশ লক্ষ ছবি এবং সাত টেরাবাইটের নতুন তথ্য যোগ করে। তাদের অংশীদারদের সাথে একসাথে, তারা ফসলের উৎপাদন উন্নত করতে, একটি নির্দিষ্ট বন্দরে কতগুলি জাহাজ আছে তা দেখতে এবং ট্র্যাক করতে সেই তথ্যের উপর মেশিন লার্নিং ব্যবহার করছে।

ই-বনায়ন। আমাজনের বন উজাড়ের প্রসঙ্গে তিনি বলেন: "আগে আমরা কয়েক বছর পর জেগে উঠতাম এবং আমাজনে একটি বড় গর্ত তৈরি হত। এখন আমরা আক্ষরিক অথেই প্রতিদিন গ্রহের প্রতিটি গাছ গণনা করতে পারি।"

৯

১ <https://www.bloomberg.com/news/videos/20170630/learningfrom-lanetsshoeboxedsizedsatellitesvideo>, ৩০ জুন, ২০১৭।

ডেমো, ইনকর্পোরেটেডের "ডেটা নেভার স্লিপস ৬.০" নামে একটি সুন্দর ইনফোগ্রাফিক রয়েছে যা দেখায় যে প্রতি মিনিটে কত ডেটা তৈরি হয়, যার মধ্যে রয়েছে:

০ চিটিপিএস://www.domo.com/learn/dataneversleeps6.

- ৪,৭৩,৮০০টি টুইট পাঠানো হয়েছে।
- ২০,৮৩,৩৩৩টি স্ব্যাপচ্যাট ছবি শেয়ার করা হয়েছে।
- নেটফ্লিক্সের ৯৭,২২২ ঘন্টা ভিডিও দেখা হয়েছে।
- ১,২৯,৮৬,১১১ মিলিয়ন টেক্সট মেসেজ পাঠানো হয়েছে।
- ৪৯,৩৮০টি ইনস্টাগ্রাম পোস্ট।
- ১,৭৬,২২০টি স্কাইপ কল।
- ৭,৫০,০০০ স্পটিফাই গান স্ট্রিম করা হয়েছে।
- ৩৮,৭৭,১৪০টি গুগল সার্চ।
- ৪,৩৩৩,৫৬০টি ইউটিউব ভিডিও দেখা হয়েছে।

বছরের পর বছর ধরে কম্পিউটিং শক্তি

ডেটা ক্রমশ বিশাল হয়ে উঠছে এবং এর প্রক্রিয়াকরণের জন্য কম্পিউটিং শক্তি বৃদ্ধি পাচ্ছে। আজকের প্রসেসরের কর্মক্ষমতা প্রায়শই **FLOPS** (প্রতি সেকেন্ডে ডাসমান বিল্ডু অপারেশন) এর মাধ্যমে পরিমাপ করা হয়। ১৯৯০-এর দশকের গোড়ার দিকে থেকে মাঝামাঝি সময়ে, দ্রুততম সুপার কম্পিউটারের গতি গিগাফ্লপ (১০ FLOPS) এ পরিমাপ করা হত। ১৯৯০-এর দশকের শেষের দিকে, ইন্টেল প্রথম টেরাফ্লপ (১০ FLOPS) সুপার কম্পিউটার তৈরি করে। ২০০০-এর দশকের ১২ মাঝামাঝি সময়ে, গতি শত শত টেরাফ্লপে পৌঁছেছিল, তারপর ২০০৮ সালে, IBM প্রথম পেটাফ্লপ (১০ FLOPS) সুপার কম্পিউটার প্রকাশ করে। বর্তমানে, দ্রুততম সুপার কম্পিউটার - IBM সামিট, যা জ্বালানি বিভাগের (DOE) ওক-এ অবস্থিত।

রিজ ন্যাশনাল ল্যাবরেটরি (ORNL)-এর ক্ষমতা ১২২.৩ পেটাফ্লপ।

১ <https://en.wikipedia.org/wiki/FLOPS>

ডিস্ট্রিবিউটেড কম্পিউটিং ইন্টারনেটের মাধ্যমে হাজার হাজার ব্যক্তিগত কম্পিউটারকে সংযুক্ত করতে পারে যাতে আরও বেশি FLOPS তৈরি হয়। ২০১৬ সালের শেষের দিকে, Folding@home নেটওয়ার্ক - একটি ডিস্ট্রিবিউটেড নেটওয়ার্ক যেখানে লোকেরা রোগ গবেষণা এবং ওষুধের জন্য তাদের ব্যক্তিগত কম্পিউটারের সম্পদ স্বেচ্ছাসেবক হিসেবে ব্যবহার করে।

esign → ১০০টিরও বেশি পেটাফ্লপ চালাতে সক্ষম ছিল। IBM-এর মতো কোম্পানিগুলি এখন এক্সাফ্লপ চালাতে সক্ষম সুপার কম্পিউটার তৈরির জন্য কাজ করছে (১০টি FLOPS)।

^২ <https://en.wikipedia.org/wiki/Folding@home>.

^৩ <https://en.wikipedia.org/wiki/FLOPS>

^৪ <https://www.ibm.com/blogs/research/2017/06/supercomputingweather->

ওডেলেক্সাক্সেল/।

বর্তমানে যে কোয়ান্টাম কম্পিউটারগুলি তৈরি হচ্ছে, তা তাত্ত্বিকভাবে আজকের "প্রচলিত কম্পিউটার"-এর ১৮,০০০,০০০,০০০,০০০,০০০ গুণ গতিতে কাজ করতে পারে! এই সংখ্যাটি এতটাই অসাধারণ যে, এক সেকেন্ডে, একটি কোয়ান্টাম কম্পিউটার তাত্ত্বিকভাবে বিশ্বের প্রথম^৫ কম্পিউটার আবির্ভাবের পর থেকে সমস্ত কম্পিউটারের দ্বারা করা মোট গণনার চেয়েও আশ্চর্যজনকভাবে বেশি গণনা করতে পারে। এই প্রায় অকল্পনীয় কম্পিউটিং শক্তি বিটকয়েনের মতো ব্লকচেইন-তিতিক ক্রিপ্টোকারেঙ্গির সাথে বিপর্যয় দেকে আনতে পারে। কম্পিউটিং শক্তির এত বিশাল বৃদ্ধির জন্য প্রস্তুত হওয়ার জন্য ইঞ্জিনিয়াররা ইতিমধ্যেই ব্লকচেইন নিয়ে পুনর্বিবেচনা করছেন।

৬

^৫ <https://medium.com/@n.biedrzycki/onlygodcancountthatfastthe->

কোয়ান্টাম কম্পিউটিং 406a0a91fcf4 এর orldof.

^৬ <https://singularityhub.com/2017/11/05/isquantumcomputingan->

ব্লকচেইনের জন্য ঝুঁকিপূর্ণ হমকিপ্রযুক্তি/।

সুপারকম্পিউটিং পাওয়ারের ইতিহাস হল এটি অবশেষে গবেষণাগার থেকে বেরিয়ে আসে, যেখানে এই কর্মক্ষমতা অর্জনের জন্য অসাধারণ পরিমাণ অর্থ ব্যয় করা হয়েছিল, "যুক্তিসংজ্ঞত মূল্যের" বাণিজ্যিক কম্পিউটার সিস্টেম এমনকি ডেস্কটপ কম্পিউটার, ল্যাপটপ, ট্যাবলেট এবং স্মার্টফোনেও।

কম্পিউটিং পাওয়ারের খরচ ক্রমাগত কমছে, বিশেষ করে ব্লাউড কম্পিউটিংয়ের ক্ষেত্রে। মানুষ আগে এই প্রশ্ন জিজ্ঞাসা করত, "আমার সর্বোচ্চ প্রক্রিয়াকরণের চাহিদা পূরণের জন্য আমার সিস্টেমে কতটা কম্পিউটিং পাওয়ার প্রয়োজন?" আজ, সেই চিন্তাভাবনা "আমার সবচেয়ে কঠিন কম্পিউটিং কাজের জন্য আমি কি অস্থায়ীভাবে ব্লাউডে যা প্রয়োজন তা দ্রুত তৈরি করতে পারি?" -এ পরিবর্তিত হয়েছে। আপনি কেবল একটি নির্দিষ্ট কাজ সম্পর্ক করার জন্য যা ব্যবহার করেন তার জন্যই অর্থ প্রদান করেন।

বিশ্বের তথ্য প্রক্রিয়াকরণের জন্য প্রচুর বিদ্যুতের প্রয়োজন হয়

বিশ্বের ইন্টারনেট-সংযুক্ত ডিভাইসগুলি থেকে তথ্য বিস্ফোরিত হচ্ছে, এবং সেই তথ্য প্রক্রিয়াকরণের জন্য প্রচুর পরিমাণে শক্তির প্রয়োজন হয়। সাম্প্রতিক একটি প্রবন্ধ অনুসারে, ২০১৫ সালে তথ্য প্রক্রিয়াকরণের জন্য শক্তির ব্যবহার প্রতি বছর ২০% হারে বৃদ্ধি পাচ্ছিল এবং বিশ্বের প্রায় তিনি থেকে পাঁচ শতাংশ শক্তি ব্যবহার করছিল। প্রবন্ধে বলা হয়েছে যে মোট তথ্য প্রক্রিয়াকরণ শক্তি

২০২৫ সালের মধ্যে ব্যবহার ২০% এ পৌঁছাতে পারে।

৭

^৭ <https://www.theguardian.com/environment/2017/dec/11/tsunamiof->

২০২৫ সালের মধ্যে বিশ্বব্যাপী বিদ্যুৎ খরচ কমাতে পারবে।

ব্লকচেইন-ভিত্তিক ক্রিপ্টোকারেন্সি বিটকয়েন হল সবচেয়ে বড় বিদ্যুৎ গ্রাহক।

মাত্র একটি বিটকয়েন লেনদেন প্রক্রিয়াকরণে প্রায় একই পরিমাণ শক্তি খরচ হয় যা গড় আমেরিকান বাড়িতে এক সপ্তাহ ধরে বিদ্যুৎ সরবরাহ করতে ব্যবহৃত হয়! লেনদেনের তথ্য বৈধ কিনা তা প্রমাণ করার জন্য বিটকয়েন "মাইনার্স" যে প্রক্রিয়া ব্যবহার করে তা থেকে শক্তির ব্যবহার আসে।

^৮ https://motherboard.vice.com/en_us/article/ywbbpm/bitcoinmining-

বৈদ্যুতিক ব্যবহার এবং থার্মেনার্জি জলবায়ু পরিবর্তন।

কিছু অনুমান অনুসারে, বিটকয়েন লেনদেনের এক বছরে অনেক দেশের চেয়ে বেশি শক্তি খরচ হয়। একসাথে, বিটকয়েন এবং ইথেরিয়াম (আরেকটি জনপ্রিয় ব্লকচেইন-ভিত্তিক প্ল্যাটফর্ম এবং ক্রিপ্টোকারেন্সি) প্রতি বছর ইসরায়েলের চেয়ে বেশি শক্তি খরচ করে এবং প্রায় একই পরিমাণ

গ্রীস হিসেবে।^৯

^{১০} <https://digiconomist.net/bitcoineenergyconsumption.>

^{১১} <https://digiconomist.net/ethereumenergyconsumption.>

২০১৮ সালে মরগান স্ট্যানলি ভবিষ্যদ্বাণী করেছিলেন যে, "এই বছর ক্রিপ্টোকারেন্সি তৈরির জন্য প্রয়োজনীয় বিদ্যুৎ খরচ আসলে ফার্মের বিপ্লব্যাপী বৈদ্যুতিক গাড়ির চাহিদাকে ছাড়িয়ে যেতে পারে - ২০২৫ সালে।" এই পরিস্থিতি টেকসই নয়, বিশেষ করে ব্লকচেইন-ভিত্তিক অ্যাপ্লিকেশনগুলিতে বিপুল আগ্রহের কারণে, এমনকি ক্রিপ্টোকারেন্সি বিস্ফোরণের বাইরেও। ব্লকচেইন সম্প্রদায় সমাধানের জন্য কাজ করছে।

^{১২, ১৩}

^{১৪} <https://www.morganstanley.com/ideas/cryptocurrenciesglobal-challenges.html>

^{১৫} <https://www.technologyreview.com/s/609480/bitcoinusesmassive-energy/>

শক্তির মাউন্ট কিন্তু সেখানে উত্তিদের ফিরু।

^{১৬} <http://mashable.com/2017/12/01/bitcoinenergy/>

বিগ-ডেটা সুযোগ

বিগডেটা বিস্ফোরণ আগামী বছরগুলিতেও তীব্র গতিতে চলতে পারে। ৫০ বিলিয়ন কম্পিউটিং ডিভাইসের আগমনের সাথে সাথে, আগামী কয়েক দশকে আরও কতগুলি ডিভাইস থাকবে তা আমরা কেবল কল্পনা করতে পারি। ব্যবসা, সরকার, সামরিক বাহিনী এমনকি ব্যক্তিদের জন্যও এই সমস্ত ডেটার উপর নিয়ন্ত্রণ রাখা অত্যন্ত গুরুত্বপূর্ণ।

এটা মজার যে বিগ ডেটা, ডেটা সায়েন্স, কৃত্রিম বুদ্ধিমত্তা এবং আরও অনেক কিছু সম্পর্কে সেরা কিছু লেখা জেপির মতো বিশিষ্ট ব্যবসায়িক সংস্থা থেকে প্রকাশিত হচ্ছে।

মরগান, ম্যাককিনসে এবং আরও অনেকে। দ্রুত ক্রমবর্ধমান সাফল্যের কারণে বৃহৎ ব্যবসার কাছে বৃহৎ তথ্যের আবেদন অনস্বীকার্য।

অনেক কোম্পানি এই বিহু উল্লেখিত প্রযুক্তির মাধ্যমে উল্লেখযোগ্য বিনিয়োগ করছে এবং মূল্যবান ফলাফল পাচ্ছে, যেমন বৃহৎ তথ্য, মেশিন লার্নিং, গভীর শিক্ষা এবং প্রাকৃতিক ভাষা প্রক্রিয়াকরণ। এটি প্রতিযোগীদের বাধ্য করছে, যার ফলে ডেটাসায়েন্স এবং কম্পিউটার বিজ্ঞানের অভিজ্ঞতা সম্পর্ক কম্পিউটিং পেশাদারদের চাহিদা দ্রুত বৃদ্ধি পাচ্ছে। এই বৃদ্ধি অনেক বছর ধরে অব্যাহত থাকার সম্ভাবনা রয়েছে।

.৭.১ বিগ ডেটা অ্যানালিটিক্স

ডেটা অ্যানালিটিক্স একটি পরিপন্থ এবং উন্নত একাডেমিক এবং পেশাদার শৃঙ্খলা। "ডেটা অ্যানালিটিক্স" শব্দটি ১৯৬২ সালে তৈরি হয়েছিল, যদিও প্রাচীন মিশরীয়দের সময় থেকে হাঁজার হাজার বছর ধরে মানুষ পরিসংখ্যান ব্যবহার করে ডেটা বিশ্লেষণ করে আসছে। বিগ ডেটা অ্যানালিটিক্স একটি সাম্প্রতিক ঘটনা - "বিগ ডেটা" শব্দটি ২০০০ খ্রিস্টাব্দের দিকে তৈরি হয়েছিল।

৬

^৮ <https://www.forbes.com/sites/gilpress/2013/05/28/averyshort-history-of-big-data/>

তথ্যবিজ্ঞানের ইতিহাস।

^৯ <https://www.flydata.com/blog/abriefhistoryofdataanalysis/>

^{১০} <https://bits.blogs.nytimes.com/2013/02/01/theorigins-of-big-data-metamorphosis/>

বিগ ডেটার চারটি V বিবেচনা করুন:

১, ৮

^{১১} <https://www.ibmbigdatahub.com/infographic/four-vs-big-data/>

এই তালিকায় আরও অনেক Vwords যুক্ত করেছে এমন অনেক নিবন্ধ এবং গবেষণাপত্র রয়েছে।

১. আয়তন—বিশ্ব যে পরিমাণ তথ্য উৎপাদন করছে তা দ্রুতগতিতে বৃদ্ধি পাচ্ছে।

২. বেগ—যে গতিতে তথ্য তৈরি হচ্ছে, যে গতিতে এটি চলাচল করে

এবং যে গতিতে ডেটা পরিবর্তন দ্রুত বৃদ্ধি পাচ্ছে।

০, ১

^{১২} <https://www.zdnet.com/article/volume-velocity-and-variety-the-three-v-of-big-data/>

^{১৩} <https://whatis.techtarget.com/definition/3Vs>

^{১৪} <https://www.forbes.com/sites/brentdykes/2017/06/28/big-data-what-is-it-and-why-is-it-so-important/>

অথবা আয়তন এবং বৈচিত্র্য বেগের উপর ফোকাস করুন।

৩. বৈচিত্র্য—উপাত্ত আগে আলফানিউমেরিক ছিল (অর্থাৎ, বর্ণানুক্রমিক অক্ষর, সংখ্যা, বিরামচিহ্ন এবং কিছু বিশেষ অক্ষর সমষ্টিত) —আজ এতে আমাদের বাড়ি, ব্যবসা, যানবাহন, শহর এবং আরও অনেক কিছুতে ইন্টারনেট অফ থিংস সেল্সের বিস্ফোরিত সংখ্যক ছবি, অডিও, ভিডিও এবং ডেটাও অন্তর্ভুক্ত রয়েছে।

৪. সত্যতা—তথ্যের বৈধতা—এটি কি সম্পূর্ণ এবং নির্ভুল? গুরুত্বপূর্ণ সিদ্ধান্ত নেওয়ার সময় আমরা কি সেই তথ্য বিশ্বাস করতে পারি? এটি কি বাস্তব?

বেশিরভাগ তথ্য এখন ডিজিটালভাবে বিভিন্ন ধরণের, অসাধারণ পরিমাণে এবং আশ্চর্যজনক গতিতে তৈরি হচ্ছে। মুরের সূত্র এবং সম্পর্কিত পর্যবেক্ষণগুলি আমাদেরকে অর্থনৈতিকভাবে ডেটা সংরক্ষণ করতে এবং দ্রুত প্রক্রিয়াজাতকরণ এবং স্থানান্তর করতে সক্ষম করেছে - এবং সময়ের সাথে সাথে তাংপর্যপূর্ণভাবে বৃদ্ধির হারে। ডিজিটাল ডেটা স্টোরেজ ক্ষমতার দিক থেকে এত বিশাল, সন্তা এবং ছোট হয়ে উঠেছে

আমরা এখন আমাদের তৈরি করা সমস্ত ডিজিটাল ডেটা সুবিধাজনকভাবে এবং অর্থনৈতিকভাবে ধরে রাখতে পারি।

ওটা তো বিশাল তথ্য।

^২ <http://www.lesk.com/mlesk/ksg97/ksg.html>. [পরবর্তী প্রবন্ধটি আমাদের নির্দেশ করেছে এই মাইকেল লেস্কের লেখাটি:

<https://www.forbes.com/sites/gilpress/2013/05/28/averyshort->

তথ্যবিজ্ঞানের ইতিহাস/.]

নিম্নলিখিত রিচার্ড ডল্লিউ. হ্যামিংয়ের উক্তি - যদিও ১৯৬২ সালের - বাকিদের জন্য সুর নির্ধারণ করে এই বই:

"কম্পিউটিংয়ের উদ্দেশ্য অন্তর্দৃষ্টি, সংখ্যা নয়!"^৩

^৩ হ্যামিং, আরডল্লিউ, সায়েন্টিস্টস অ্যান্ড ইঞ্জিনিয়ারদের জন্য সংখ্যাসূচক পদ্ধতি (নিউ ইয়র্ক, এনওয়াই, ম্যাকগ্রা হিল, ১৯৬২)।

[পরবর্তী প্রবন্ধটি আমাদের হ্যামিংসের বই এবং তার উন্নতিটির দিকে নির্দেশ করেছে যা আমরা উন্নত করেছি: <https://www.forbes.com/sites/gilpress/2013/05/28/avery-horthistoryofdatascience/.>]

ডেটা সায়েন্স অসাধারণ গতিতে নতুন, গভীর, সুস্থ এবং আরও মূল্যবান অন্তর্দৃষ্টি তৈরি করছে। এটি সত্যিই একটি পরিবর্তন আনছে। বিগ ডেটা অ্যানালিটিক্স এই উত্তরের একটি অবিচ্ছেদ্য অংশ। আমরা পর্ব ১৬- তে NoSQL ডাটাবেস, Hadoop MapReduce প্রোগ্রামিং, Spark, রিয়েলটাইম ইন্টারনেট অফ থিংস (IoT) স্ট্রিম প্রোগ্রামিং এবং আরও অনেক কিছুর উপর হাতে-কলমে কেস স্টাডির মাধ্যমে বিগ ডেটা অবকাঠামোর উপর আলোকপাত করি।

শিল্প, সরকার এবং শিক্ষাক্ষেত্রে বৃহৎ তথ্যের পরিধি সম্পর্কে ধারণা পেতে, উচ্চ-রেজোলিউশনের গ্রাফিকটি দেখুন। সহজে পঠনযোগ্যতার জন্য আপনি জুম করতে ক্লিক করতে^৪ পারেন:

^৪ টার্ক, এম., এবং জে. হাও, প্রেট পাওয়ার, প্রেট রেসপন্সিভিলিটি: দ্য ২০১৮ বিগ ডেটা এবং এআই ল্যান্ডস্কেপ, <http://mattturck.com/bigdata2018/>

http://mattturck.com/wpcontent/uploads/2018/07/Matt_Turck_FirstMark_Big_Data_L_এনডিস্কেপ



৭.২ ডেটা সায়েন্স এবং বিগ ডেটা পার্থক্য আনছে: ব্যবহারের ক্ষেত্রে

ডেটাসায়েন্স ক্ষেত্রটি দ্রুত বৃদ্ধি পাচ্ছে কারণ এটি উল্লেখযোগ্য ফলাফল তৈরি করছে যা একটি পার্থক্য তৈরি করছে। আমরা নিম্নলিখিত সারণীতে ডেটাসায়েন্স এবং বৃহৎ ডেটা ব্যবহারের ঘটনাগুলি তালিকাভুক্ত করেছি।

আমরা আশা করি যে বইটিতে থাকা ব্যবহারের উদাহরণ এবং আমাদের উদাহরণগুলি আপনাকে আপনার ক্যারিয়ারে নতুন ব্যবহারের ক্ষেত্রে অনুপ্রাণিত করবে। বিগডেটা অ্যানালিটিক্সের ফলে লাভ বেড়েছে, গ্রাহক সম্পর্ক উন্নত হয়েছে, এমনকি ক্রীড়া দলগুলি খেলোয়াড়দের উপর কম খরচ করে আরও বেশি গেম এবং চ্যাম্পিয়নশিপ জিতেছে।

৬ টেক্স

^৫ সাওচিক, টি., বিগ ডেটা বেসবল: গণিত, অলৌকিক ঘটনা এবং ২০ বছরের হারানোর ধারার সমাপ্তি (নিউ ইয়র্ক, ফ্ল্যাট আয়রন বুকস, ২০১৫)।

আইরেস, আই., সুপার ক্রাঞ্চার্স (ব্যাট্টাম বুকস, ২০০৭), পৃষ্ঠা ৭১০।

^৭ লুইস, এম., মানিবল: দ্য আর্ট অফ উইনিং অ্যান আনফেয়ার গেম (ডিলিউডলিউ নটন অ্যান্ড কোম্পানি, ২০০৪)।

বিজ্ঞান ব্যবহার মামলা	অসঙ্গতি সনাত্তকরণ প্রতিবন্ধী ব্যক্তিদের সহায়তা করা স্বয়ংক্রিয় বীমা ঝুঁকি ভবিষ্যদ্বাণী স্বয়ংক্রিয়ভাবে বন্ধ ক্যাপশনিং স্বয়ংক্রিয় ছবির ক্যাপশন	আবহাওয়া-সংবেদনশীল পণ্য বিক্রয়ের পূর্বাভাস দেওয়া ভবিষ্যদ্বাণীমূলক বিশ্লেষণ প্রতিরোধমূলক ওষধ রোগ প্রতিরোধ প্রাদুর্ভাব সাংকেতিক ভাষা পড়া রিয়েল এস্টেট মূল্যায়ন সুপারিশ সিস্টেম অতিরিক্ত বুকিং কমানো রাইড শেয়ারিং বুঁকি হ্রাসকরণ রোবো আর্থিক উপদেষ্টা নিরাপত্তা বুদ্ধি
স্বয়ংক্রিয় বিনিয়োগ	মুখের স্বীকৃতি	ভবিষ্যদ্বাণীমূলক বিশ্লেষণ
স্বায়ত্ত্বাসিত জাহাজ	ফিটনেস ট্র্যাকিং	প্রতিরোধমূলক ওষধ
মন্তিক্ষের মানচিত্র তৈরি করা	জালিয়াতি সনাত্তকরণ	রোগ প্রতিরোধ
কলার শনাক্তকরণ	খেলা খেলা	প্রাদুর্ভাব
ক্যান্সার	জিনোমিক্স এবং স্বাস্থ্যসেবা	সাংকেতিক ভাষা পড়া
রোগ নির্ণয়/চিকিৎসা	ভৌগোলিক তথ্য ব্যবস্থা	রিয়েল এস্টেট মূল্যায়ন
কার্বন নির্গমন	(জিআইএস)	সুপারিশ
হ্রাস	জিপিএস সিস্টেম	সিস্টেম
হাতের লেখার	হাসপাতালে ভর্তি হ্রাস	অতিরিক্ত বুকিং কমানো
শ্রেণীবিভাগ	স্বাস্থ্যের উন্নতি	রাইড শেয়ারিং
কম্পিউটার দৃষ্টি	মানব জিনোম সিকোয়েলিং	বুঁকি হ্রাসকরণ
ক্রেডিট স্লেটারিং	পরিচয় চুরি প্রতিরোধ	রোবো আর্থিক উপদেষ্টা
অপরাধ: অবস্থান ভবিষ্যদ্বাণী করা	পরিচয় চুরি প্রতিরোধ	নিরাপত্তা বুদ্ধি

ରାଇମ: ପୁନରାବୃତ୍ତିର ପୂର୍ବାଭାସ	ଇମିଉନୋଥେରାପି	ସ୍ଵଚାଲିତ ଗାଡ଼ି
ଦେଓଯା	ବୀମା ମୂଲ୍ୟ ନିର୍ଧାରଣ	ଅନୁଭୂତି ବିଶ୍ଳେଷଣ
ଅପରାଧ: ଭବିଷ୍ୟତ୍ୱାଣୀମୂଲକ ପୁଲିଶିଂ	ବୁଦ୍ଧିମାନ ସହକାରୀ	ଭାଗାଭାଗୀ ଅର୍ଥନୀତି
ଅପରାଧ: ପ୍ରତିରୋଧ	ଇନ୍ଟାରନେଟ ଅଫ ଥିଂସ (ଆଇଓଟି) ଏବଂ ମେଡିକେଲ ଡିଭାଇସ ପର୍ଯ୍ୟବେକ୍ଷଣ	ସାଦୃଶ୍ୟ ସନାତକରଣ
CRISPR ଜିନ	ଇନ୍ଟାରନେଟ ଅଫ ଥିଂସ ଏବଂ ଆବହାଓଯାର ପୂର୍ବାଭାସ	ସ୍ମାର୍ଟ ଶହର
ଫସଲ	ମଜୁଦ ନିୟମନ	ସ୍ମାର୍ଟ ହୋମସ
ଉତ୍ପାଦନେର ଉନ୍ନତି	ଭାଷା ଅନୁବାଦ	ସ୍ମାର୍ଟ ମିଟାର
ଗ୍ରାହକ ମଞ୍ଚନ	ଅବସ୍ଥାନ ଭିତ୍ତିକ ପରିବେବା	ସ୍ମାର୍ଟ ଟ୍ରାଫିକ ନିୟମନ
ଅଭିଜ୍ଞତା	ଆନୁଗତ୍ୟ ପ୍ରୋଗ୍ରାମ	ସାମାଜିକ ବିଶ୍ଳେଷଣ
ଗ୍ରାହକ ଧରେ ରାଖା	ମ୍ୟାଲୋଯାର ସନାତକରଣ	ସାମାଜିକ ଗ୍ରାଫ ବିଶ୍ଳେଷଣ
ଗ୍ରାହକ ସନ୍ତ୍ରିଷ୍ଟି	ମ୍ୟାପିଂ	ସ୍ପ୍ଯାମ ସନାତକରଣ
ଗ୍ରାହକ ସେବା	ବିପନ୍ନ	ସ୍ଵାନିକ ତଥ୍ୟ ବିଶ୍ଳେଷଣ
ଗ୍ରାହକ ସେବା	ମାର୍କେଟିଂ ବିଶ୍ଳେଷଣ	କ୍ରୀଡ଼ା ନିୟୋଗ ଏବଂ କୋଟିଂ
ଏଜେନ୍ଟ	ସଙ୍ଗୀତ ପ୍ରଜନ୍ମ	
କାସ୍ଟମାଇଜଡ ଡାଯେଟ	ପ୍ରାକୃତିକ ଭାଷା ଅନୁବାଦ	ଶେୟାର ବାଜାରେର ପୂର୍ବାଭାସ
ସାଇବାର ନିରାପତ୍ତା	ନତୁନ ଓସ୍ଥି	ଛାତ୍ରଦେର ପାରଫର୍ମେଞ୍ଜ
ଡେଟା ମାଇନିଂ	ଓପିଓୟେଡ ଅପବ୍ୟବହାର ପ୍ରତିରୋଧ	ମୂଲ୍ୟାଯନ
ଡେଟା ଭିଜ୍ୟୁଯାଲାଇଜେଶନ	ବ୍ୟକ୍ତିଗତ ସହକାରୀ	ପାଠ୍ୟର ସାରସଂକ୍ଷେପ
ନତୁନ ଭାଇରାସ ସନାତକରଣ	ବ୍ୟକ୍ତିଗତକୃତ ଓସ୍ଥି	ଟେଲିମେଡିସିନ
ନତୁନ ଭାଇରାସ ସନାତକରଣ	ବ୍ୟକ୍ତିଗତକୃତ କେନାକାଟା	ସନ୍ତ୍ରାସୀ ହାମଲା
ସ୍ତନ ରୋଗ ନିର୍ଣ୍ୟ		ପ୍ରତିରୋଧ
କ୍ୟାଳ୍ୟାର	ଫିଶିଂ ନିର୍ମୂଳ	ଚୁରି ପ୍ରତିରୋଧ
ହଦରୋଗ ନିର୍ଣ୍ୟ	ଦୂସଣ ହ୍ରାସ	ଭରଣେର ପରାମର୍ଶ
ରୋଗ ନିର୍ଣ୍ୟରେ ଓସ୍ଥି	ନିର୍ଭୁଲ ଚିକିତ୍ସା	ଟ୍ରେନ୍ଡ ସ୍ପଟିଂ
		ଭିଜ୍ୟୁଯାଲ ପଣ୍ୟ ଅନୁସନ୍ଧାନ

দুর্যোগের শিকার	রোগের প্রাদুর্ভাবের পূর্বাভাস দেওয়া	ভয়েস স্বীকৃতি
সনাত্তকরণ	স্বাস্থ্যের ফলাফলের পূর্বাভাস দেওয়া	ভয়েস সার্চ
ড্রোন	শিক্ষার্থী ভর্তির পূর্বাভাস দেওয়া	আবহাওয়ার পূর্বাভাস
গতিশীল ড্রাইভিং		
রুট		
গতিশীল মূল্য নির্ধারণ		
ইলেকট্রনিক স্বাস্থ্য		
রেকর্ড		
আবেগ সনাত্তকরণ		
শক্তি খরচ		
হ্রাস		

১.৮ কেস স্টাডি—একটি বিগ-ডেটা মোবাইল অ্যাপ্লিকেশন

গুগলের ওয়েজ জিপিএস নেভিগেশন অ্যাপ, যার ৯০ মিলিয়ন মাসিক সক্রিয় ব্যবহারকারী রয়েছে, এটি সবচেয়ে সফল বিগডেটা অ্যাপগুলির মধ্যে একটি। প্রাথমিক জিপিএস নেভিগেশন ডিভাইস এবং অ্যাপগুলি আপনার গন্তব্যে পৌঁছানোর সেরা রুট নির্ধারণের জন্য স্ট্যাটিক ম্যাপ এবং জিপিএস স্থানাঙ্কের উপর নির্ভর করত। তারা পরিবর্তনশীল ট্র্যাফিক পরিস্থিতির সাথে গতিশীলভাবে খাপ খাইয়ে নিতে পারত না।

^৮ <https://www.waze.com/brands/drivers/>.

Waze বিপুল পরিমাণে **ক্রাউডসোর্সড** ডেটা প্রক্রিয়া করে—অর্থাৎ, বিশ্বব্যাপী তাদের ব্যবহারকারীদের এবং তাদের ব্যবহারকারীদের ডিভাইসগুলি থেকে ক্রমাগত সরবরাহ করা ডেটা। তারা এই ডেটা আসার সাথে সাথে বিশ্লেষণ করে আপনাকে সবচেয়ে কম সময়ের মধ্যে আপনার গন্তব্যে পৌঁছানোর জন্য সেরা রুট নির্ধারণ করে। এটি সম্পূর্ণ করার জন্য, Waze আপনার স্মার্টফোনের ইন্টারনেট সংযোগের উপর নির্ভর করে।

অ্যাপটি স্বয়ংক্রিয়ভাবে তাদের সার্ভারে অবস্থান আপডেট পাঠায় (ধরে নিচ্ছি আপনি এটির অনুমতি দিয়েছেন)।

বর্তমান ট্র্যাফিক পরিস্থিতির উপর ভিত্তি করে তারা আপনাকে গতিশীলভাবে পুনরায় রুট করতে এবং তাদের মানচিত্র সুর করতে সেই ডেটা ব্যবহার করে। ব্যবহারকারীরা অন্যান্য তথ্য যেমন রাস্তাঘাট, নির্মাণ, বাধা, ভাঙা লেনে যানবাহন, পুলিশের অবস্থান, গ্যাসের দাম এবং আরও অনেক কিছু রিপোর্ট করে। এরপর Waze অন্যান্যদের সতর্ক করে এসব স্থানে ড্রাইভারদের।

Waze তার পরিষেবা প্রদানের জন্য অনেক প্রযুক্তি ব্যবহার করে। Waze কীভাবে বাস্তবায়িত হয় সে সম্পর্কে আমরা অবগত নই, তবে তারা সম্ভবত যে প্রযুক্তিগুলি ব্যবহার করে তার একটি তালিকা আমরা নীচে উপস্থাপন করছি। আপনি পর্ব ১১-৬ -এ এর মধ্যে অনেকগুলি ব্যবহার করবেন। উদাহরণস্বরূপ,

- আজকাল তৈরি বেশিরভাগ অ্যাপই অন্তত কিছু ওপেনসোর্স সফ্টওয়্যার ব্যবহার করে। এই বইটিতে আপনি অনেক ওপেনসোর্স লাইব্রেরি এবং সরঞ্জামের সুবিধা পাবেন।
- ওয়েজ তাদের সার্ভার এবং ব্যবহারকারীদের মধ্যে ইন্টারনেটের মাধ্যমে তথ্য আদান-প্রদান করে। মোবাইল ডিভাইস। আজকাল, এই ধরনের ডেটা প্রায়শই JSON (জাভাস্ক্রিপ্ট অবজেক্ট নোটেশন) ফর্ম্যাটে প্রেরণ করা হয়, যা আমরা পর্ব ৯ এ উপস্থাপন করব এবং পরবর্তী অধ্যায়গুলিতে ব্যবহার করব। JSON ডেটা সাধারণত আপনার ব্যবহৃত লাইব্রেরি দ্বারা আপনার কাছ থেকে লুকানো থাকে।
- Waze আপনাকে নির্দেশিকা এবং সতর্কতা জানানোর জন্য স্পিচ সিনথেসিস ব্যবহার করে এবং আপনার কথ্য আদেশগুলি বোঝার জন্য স্পিচ রিকগনিশন ব্যবহার করে। আমরা হ্যাপটার 13- এ IBM ওয়াটসনের স্পিচ-সিনথেসিস এবং স্পিচ রিকগনিশন ক্ষমতা ব্যবহার করি।
- একবার Waze একটি কথ্য naturallanguage কমান্ডকে টেক্সটে রূপান্তরিত করলে, এটিকে সঠিক ক্রিয়াটি নির্ধারণ করতে হবে, যার জন্য প্রাকৃতিক ভাষা প্রক্রিয়াকরণ (NLP) প্রয়োজন। আমরা অধ্যায় 11- এ NLP উপস্থাপন করি এবং পরবর্তী কয়েকটি অধ্যায়ে এটি ব্যবহার করি।
- Waze অ্যালার্ট এবং ম্যাপের মতো গতিশীলভাবে আপডেট করা ভিজুয়ালাইজেশন প্রদর্শন করে। Waze আপনাকে মানচিত্রগুলি সরানোর মাধ্যমে বা জুম ইন এবং আউট করে তাদের সাথে ইন্টারঅ্যাক্ট করতে সক্ষম করে। আমরা পুরো বই জুড়ে Matplotlib এবং Seaborn দিয়ে গতিশীল ভিজুয়ালাইজেশন তৈরি করি এবং আমরা অধ্যায় 12 এবং 12-এ Folium দিয়ে ইন্টারেক্টিভ মানচিত্র প্রদর্শন করি।
- Waze আপনার ফোনটিকে স্ট্রিমিং ইন্টারনেট অফ থিংস (IoT) ডিভাইস হিসেবে ব্যবহার করে। প্রতিটি ফোন একটি GPS সেন্সর যা ইন্টারনেটের মাধ্যমে Waze-তে ক্রমাগত ডেটা স্ট্রিম করে। অধ্যায় 16-এ, আমরা IoT চালু করি এবং সিমুলেটেড IoT স্ট্রিমিং সেন্সরগুলির সাথে কাজ করি।
- Waze একসাথে লক্ষ লক্ষ ফোন থেকে IoT স্ট্রিম গ্রহণ করে। আপনার ডিভাইসের মানচিত্র আপডেট করতে, প্রাসঙ্গিক সতর্কতা প্রদর্শন করতে এবং বলতে এবং সম্ভবত আপনার ড্রাইভিং দিকনির্দেশনা আপডেট করতে এটিকে অবিলম্বে সেই ডেটা প্রক্রিয়াকরণ, সংরক্ষণ এবং বিশ্লেষণ করতে হবে। এর জন্য ক্লাউডে কম্পিউটারের ক্লাস্টারগুলির সাথে ব্যাপকভাবে সম্মতরাল প্রক্রিয়াকরণ ক্ষমতা প্রয়োজন। অধ্যায় 6-এ, আমরা স্ট্রিমিং ডেটা গ্রহণের জন্য, উপর্যুক্ত ডাটাবেসে সেই বড় ডেটা সংরক্ষণ করার জন্য এবং সফ্টওয়্যার এবং হার্ডওয়্যার দিয়ে ডেটা প্রক্রিয়াকরণের জন্য বিভিন্ন বিগডেটা অবকাঠামো প্রযুক্তি চালু করব যা ব্যাপকভাবে সম্মতরাল প্রক্রিয়াকরণ ক্ষমতা প্রদান করে।
- ওয়েজ কৃত্রিম বুদ্ধিমত্তার ক্ষমতা ব্যবহার করে ডেটা বিশ্লেষণের কাজ সম্পাদন করে যা এটি প্রাপ্ত তথ্যের উপর ভিত্তি করে সেরা রুটগুলি পূর্বাভাস দিতে সক্ষম করে। অধ্যায় 18- এ এবং বিপুল পরিমাণ ডেটা বিশ্লেষণ করতে এবং সেই তথ্যের উপর ভিত্তি করে ভবিষ্যত্বান্বী করতে যথাক্রমে মেশিন লার্নিং এবং ডিপ লার্নিং ব্যবহার করে।
- Waze সম্ভবত তার রাউটিং তথ্য একটি গ্রাফ ডাটাবেসে সংরক্ষণ করে। এই ধরনের ডাটাবেস দক্ষতার সাথে সংক্ষিপ্তম রুট গণনা করতে পারে। আমরা Neo4J এর মতো গ্রাফ ডাটাবেস চালু করি অধ্যায় ১৬।
- অনেক গাড়ি এখন এমন ডিভাইস দিয়ে সজ্জিত যা তাদের গাড়ি এবং তাদের চারপাশের বাধাগুলি "দেখতে" সক্ষম করে। উদাহরণস্বরূপ, এগুলি স্বয়ংক্রিয় ব্রেকিং সিস্টেম বাস্তবায়নে সহায়তা করার জন্য ব্যবহৃত হয় এবং স্ব-চালিত গাড়ি প্রযুক্তির একটি গুরুত্বপূর্ণ অংশ। রিপোর্ট করার জন্য ব্যবহারকারীদের উপর নির্ভর করার পরিবর্তে

রাস্তার পাশে আটকে থাকা গাড়ি এবং বিপর্যস্ত গাড়ির ক্ষেত্রে, মেডিগেশন অ্যাপগুলি ক্যামেরা এবং অন্যান্য সেলরের সুবিধা নিতে পারে ডিপলার্নিং কম্পিউটারভিশন কোশল ব্যবহার করে "অন দ্য ফ্লাই" ছবি বিশ্লেষণ করতে এবং স্বয়ংক্রিয়ভাবে সেই জিনিসগুলি রিপোর্ট করতে। আমরা হ্যাপ্টার 15-এ কম্পিউটার ভিশনের জন্য ডিপ লার্নিং চালু করেছি।

১.৯ তথ্য বিজ্ঞানের ভূমিকা: কৃত্রিম বুদ্ধিমত্তা—সিএস এবং তথ্য বিজ্ঞানের হেদস্তলে

যখন একটি শিশু প্রথম চোখ খোলে, তখন কি সে তার বাবা-মায়ের মুখ "দেখে"? সে কি মুখ কী—এমনকি একটি সাধারণ আকৃতি কী—সে সম্পর্কে কোনও ধারণা বোবে? শিশুদের অবশ্যই তাদের চারপাশের জগৎ "শিখতে" হবে। আজকাল কৃত্রিম বুদ্ধিমত্তা (AI) এটাই করছে। এটি প্রচুর পরিমাণে তথ্য দেখছে এবং তা থেকে শিখছে। AI গেম খেলতে, বিস্তৃত পরিসরে কম্পিউটার ভিশন অ্যাপ্লিকেশন বাস্তবায়ন করতে, স্ব-চালিত গাড়ি সক্ষম করতে, রোবটদের নতুন কাজ করতে শেখাতে, চিকিৎসার অবস্থা নির্ণয় করতে, প্রায় বাস্তব সময়ে অন্যান্য ভাষায় কথা অনুবাদ করতে, জ্ঞানের বিশাল ডাটাবেস ব্যবহার করে ইচ্ছামত প্রশ্নের উত্তর দিতে পারে এমন চ্যাটবট তৈরি করতে এবং আরও অনেক কিছু করতে ব্যবহৃত হচ্ছে। কয়েক বছর আগে কে অনুমান করেছিল যে কৃত্রিম বুদ্ধিমান স্ব-চালিত গাড়ি আমাদের রাস্তায় চলতে দেওয়া হবে—অথবা এমনকি সাধারণ হয়ে উঠবে? তবুও, এটি এখন একটি অত্যন্ত প্রতিযোগিতামূলক ক্ষেত্র। এই সমস্ত শেখার চূড়ান্ত লক্ষ্য হল কৃত্রিম সাধারণ বুদ্ধিমত্তা—একটি কৃত্রিম বুদ্ধিমত্তা যা মানুষের পাশাপাশি বুদ্ধিমত্তার কাজও করতে পারে। এটি অনেকের কাছে একটি ভৌতিক চিন্তা।

কৃত্রিম বুদ্ধিমত্তার মাইলফলক

বিশেষ করে, কৃত্রিম বুদ্ধিমত্তার বেশ কিছু মাইলফলক মানুষের মনোযোগ এবং কল্পনাকে আকর্ষণ করেছিল, সাধারণ জনগণকে ভাবতে বাধ্য করেছিল যে কৃত্রিম বুদ্ধিমত্তা বাস্তব এবং ব্যবসাগুলিকে কৃত্রিম বুদ্ধিমত্তার বাণিজ্যিকীকরণ সম্পর্কে ভাবতে বাধ্য করেছিল:

- ১৯৯৭ সালে আইবিএমের ডিপলু^১ কম্পিউটার সিস্টেম এবং দাবা গ্র্যান্ডমাস্টারের মধ্যে একটি ম্যাচে গ্যারি কাসপারভ, ডিপলু প্রথম কম্পিউটার হিসেবে দাবায়ের রাজস্বকালকে হারিয়েছেন টুর্নামেন্টের পরিস্থিতিতে চ্যাম্পিয়ন। আইবিএম ডিপলুকে লক্ষ লাঙ্ক গ্র্যান্ডমাস্টার দাবা গেম দিয়ে লোড করেছিল। ডিপলু প্রতি সেকেন্ডে ২০০ মিলিয়ন চাল পর্যন্ত নির্ভুল শক্তি ব্যবহার করতে সক্ষম ছিল! এটিই বিশাল তথ্য। আইবিএম কানেক্ট মেলন বিশ্ববিদ্যালয় ফ্রেডকিন পুরস্কার পেয়েছিল, যা ১৯৮০ সালে বিশ্ব দাবা চ্যাম্পিয়নকে পরাজিত করে প্রথম কম্পিউটারের নির্মাতাদের জন্য ১০০,০০০ ডলার অফার করেছিল।

২

^১ https://en.wikipedia.org/wiki/Deep_Blue_versus_Garry_Kasparov

^০ [https://en.wikipedia.org/wiki/Deep_Blue_\(computer_chess\)](https://en.wikipedia.org/wiki/Deep_Blue_(computer_chess))

^১ [https://en.wikipedia.org/wiki/Deep_Blue_\(computer_chess\)](https://en.wikipedia.org/wiki/Deep_Blue_(computer_chess))

^২ <https://articles.latimes.com/1997/jul/30/news/mn176961>

- ২০১১ সালে, আইবিএমের ওয়াটসন ১ মিলিয়ন ডলারের ম্যাচে দুই সেরা মানব জেওপার্ডি! খেলোয়াড়কে পরাজিত করেছিলেন। ওয়াটসন একই সাথে শত শত ভাষা বিশ্লেষণ কোশল ব্যবহার করে সনাত্ত করেছিলেন

২০ কোটি পৃষ্ঠার কন্টেন্ট (সমস্ত উইকিপিডিয়া সহ) সঠিক উভর তৈরির জন্য চার টেরাবাইট স্টোরেজ প্রয়োজন।

ওয়াটসনকে মেশিন লার্নিং এবং রিইনফোর্সমেন্ট লার্নিং কৌশল সম্পর্কে প্রশিক্ষণ দেওয়া হয়েছিল। হ্যাপটার

১৩ আইবিএম ওয়াটসন নিয়ে আলোচনা করে এবং হ্যাপটার ১৪ মেশিন লার্নিং নিয়ে আলোচনা করে।

^৭ <https://www.techrepublic.com/article/ibmwatsontheinside/>

সুপার কম্পিউটারের বিপদ জয়ের গল্প জন্মগ্রহণ করেছে-
হাতিটান্টস্টোডোনেক্সট/.

^৮ [https://en.wikipedia.org/wiki/Watson_\(computer\)](https://en.wikipedia.org/wiki/Watson_(computer))

^৯ <https://www.aaai.org/Magazine/Watson/watson.php>, এআই ম্যাগাজিন, শরৎ
২০১০।

- গো—হাজার হাজার বছর আগে চীনে তৈরি একটি বোর্ড গেম—এটিকে ব্যাপকভাবে বিবেচনা করা হয় ১০টি সন্তান্য বোর্ড কনফিগারেশন সহ উন্নতিপূর্ণ সবচেয়ে জটিল গেমগুলির মধ্যে একটি।

এটি কত বড় সংখ্যা তা বোঝার জন্য, এটি বিশ্বাস করা হয় যে পরিচিত মহাবিশ্বে (মাত্র) 78 ৮, ৯ টি 10 থেকে 10 পরমাণুর মধ্যে রয়েছে!

৮৭

২০১৫ সালে, গুগলের ডিপমাইন্ড গ্রুপ

দ্বারা তৈরি আলফাগো ইউরোপীয় গো চ্যাম্পিয়ন ফ্যান ছাড়িকে পরাজিত করার জন্য দুটি নিউরাল নেটওয়ার্কের সাহায্যে গভীর শিক্ষা ব্যবহার করে। গোকে দাবার চেয়ে অনেক বেশি জটিল খেলা খেলে মনে করা হয়। হ্যাপটার ১৫ নিউরাল নেটওয়ার্ক এবং গভীর শিক্ষা নিয়ে আলোচনা করে।

^{১০} <http://www.usgo.org/briefhistorygo>.

^{১১} <https://www.pbs.org/newshour/science/googleartificial-brain-champs-world-chess-champs/>-
বুদ্ধিমত্তাবিটচ্যাম্পিয়ন এবিশ্বসবচেয়ে জটিলবোর্ড-
আমে।

^{১২} <https://www.universetoday.com/36302/atomsintheuniverse/>

^{১৩} https://en.wikipedia.org/wiki/Observable_universe#Matter_content

- সম্পত্তি, গুগল তার AlphaGo AI কে সাধারণীকরণ করে **AlphaZero** তৈরি করেছে—একটি গেমপ্লেয়িং AI যা নিজেকে অন্যান্য গেম খেলতে শেখায়। ২০১৭ সালের ডিসেম্বরে, AlphaZero রিইনফোর্সমেন্ট লার্নিং ব্যবহার করে চার ঘন্টারও কম সময়ে দাবা খেলার নিয়ম শিখেছে এবং নিজেকে দাবা খেলতে শিখিয়েছে। এরপর এটি ১০০ গেমের ম্যাচে বিশ্ব চ্যাম্পিয়ন দাবা প্রোগ্রাম, Stockfish 8 কে হারিয়েছে—প্রতিটি খেলা জিতেছে অথবা ড্র করেছে। মাত্র আট ঘন্টা ধরে Go তে প্রশিক্ষণ নেওয়ার পর, AlphaZero সক্ষম হয়েছিল গো বনাম আলফাগো খেলতে, ১০০টি খেলার মধ্যে ৬০টি জিতেছে।

^{১৪} [https://www.theguardian.com/technology/2017/dec/07/alphazero-](https://www.theguardian.com/technology/2017/dec/07/alphazero-goes-to-millions-of-games)

গুগলডিপমিন্ডাইবিটসচ্যাম্পিয়নপ্রোগ্রামনিজেকে খেলতে শেখানো-
আমাদের ঘন্টা।

একটি ব্যক্তিগত উপাখ্যান

যখন লেখকদের একজন, হার্ডে ডিটেল, ১৯৩০ সালের মাঝামাঝি সময়ে এমআইটিতে স্নাতকোত্তর ছাত্র ছিলেন www.EBooksWorld.ir

৯৬০-এর দশকে, তিনি কৃতিম বুদ্ধিমত্তার (এআই) প্রতিষ্ঠাতাদের একজন মারভিন মিনস্কির (যাকে এই বইটি উৎসর্গ করা হয়েছে) সাথে স্নাতক স্তরের

কৃতিম বুদ্ধিমত্তা কোর্স করেন। হার্ডে:

প্রফেসর মিনস্কির একটি বড় টার্ম প্রজেক্টের প্রয়োজন ছিল। তিনি আমাদের বুদ্ধিমত্তা কী তা নিয়ে ভাবতে এবং একটি কম্পিউটারকে বুদ্ধিমত্তার সাথে কিছু করতে শেখাতে বলেছিলেন। কোর্সে আমাদের গ্রেড প্রায় সম্পূর্ণরূপে প্রকল্পের উপর নির্ভরশীল হবে। কোনও চাপ নেই!

আমি স্কুলগুলি তাদের শিক্ষার্থীদের বুদ্ধিমত্তার দক্ষতা মূল্যায়নে সহায়তা করার জন্য পরিচালিত স্ট্যান্ডার্ডাইজড আইকিউ পরীক্ষাগুলি নিয়ে গবেষণা করেছি। একজন গণিতবিদ হিসেবে, আমি পরবর্তী সংখ্যাটি ইচ্ছামত দৈর্ঘ্য এবং জটিলতার সংখ্যার ক্রমানুসারে ভবিষ্যদ্বাণী করার জনপ্রিয় আইকিউটেস্ট সমস্যাটি মোকাবেলা করার সিদ্ধান্ত নিয়েছি। আমি একটি প্রাথমিক ডিজিটাল ইকুইপমেন্ট কর্পোরেশন PDP1-এ ইন্টারেক্টিভ লিস্প রানিং ব্যবহার করেছি এবং আমার সিকোয়েল প্রেডিস্টেরকে বেশ জটিল কিছু স্টাফের উপর রান করতে সক্ষম হয়েছি, যা আইকিউ পরীক্ষায় যা দেখেছি তার চেয়েও বেশি চ্যালেঞ্জ মোকাবেলা করেছে।

প্রকল্পের প্রয়োজনীয়তা পূরণের জন্য লিস্পের দীর্ঘ তালিকাগুলিকে ইচ্ছামত পুনরাবৃত্তভাবে পরিচালনা করার ক্ষমতা ঠিক এটাই ছিল যা আমার প্রয়োজন ছিল। পাইথন অফ ers recursion এবং generalized list processing (অধ্যায় 5)।

আমি আমার অনেক MIT সহপাঠীর উপর সিকোয়েল প্রেডিস্টের ব্যবহার করে দেখেছি। তারা সংখ্যার সিকোয়েল তৈরি করত এবং আমার প্রেডিস্টের টাইপ করত। PDP1 কিছুক্ষণের জন্য "চিন্তা" করত—প্রায়শই দীর্ঘ সময় ধরে—এবং প্রায় সবসময় সঠিক উত্তরটিই পেত।

তারপর আমার একটা সমস্যা হলো। আমার এক সহপাঠী ১৪, ২৩, ৩৪ এবং ৪২ নম্বর ক্রম টাইপ করলো। আমার ভবিষ্যদ্বাণীকারী এটি নিয়ে কাজ করতে লাগলো, এবং PDP1 দীর্ঘ সময় ধরে কাজ বন্ধ করে দিল, পরবর্তী সংখ্যাটি ভবিষ্যদ্বাণী করতে ব্যর্থ হল। আমি বুঝতে পারলাম না। আমার সহপাঠী আমাকে রাতারাতি এটি নিয়ে ভাবতে বললো, এবং সে পরের দিন উত্তরটি প্রকাশ করবে, দাবি করলো যে এটি একটি সহজ ক্রম। আমার প্রচেষ্টা কোন কাজে আসেনি।

পরের দিন সে আমাকে বললো পরের সংখ্যাটা ৫৭, কিন্তু কেন আমি বুঝতে পারলাম না। তাই সে আমাকে আবার রাতারাতি এটা নিয়ে ভাবতে বললো, আর পরের দিন বললো পরের সংখ্যাটা ১২৫। তাতেও কোন লাভ হলো না—আমি হতবাক হয়ে গেলাম। সে বললো যে ক্রমটা ম্যানহাটনের দুই-পাশ রাস্তার সংখ্যা। আমি চিক্কার করে বললাম, "অপমানজনক", কিন্তু সে বললো যে এটি সংখ্যাসূচক ক্রম অনুসারে পরবর্তী সংখ্যাটি ভবিষ্যদ্বাণী করার আমার মানদণ্ড পূরণ করে। আমার বিশ্বদর্শন ছিল গণিত—তার দৃষ্টিভঙ্গি ছিল বিস্তৃত।

বছরের পর বছর ধরে, আমি বন্ধুবান্ধব, আঞ্চলিক সম্পর্কে এবং পেশাদার সহকর্মীদের উপর এই সিকোয়েলটি চেষ্টা করেছি। ম্যানহাটনে সময় কাটিয়েছেন এমন কয়েকজনই এটি সঠিকভাবে করেছেন। আমার সিকোয়েল প্রেডিস্টের এই ধরণের সমস্যাগুলি পরিচালনা করার জন্য কেবল গণিতিক জ্ঞানের চেয়ে অনেক বেশি প্রয়োজন ছিল, (সম্ভবত বিশাল) বিশ্ব জ্ঞানের প্রয়োজন ছিল।

ওয়াটসন এবং বিগ ডেটা নতুন সম্ভাবনার সূচনা করে যখন আমি এবং পল এই পাইথন বইটি লেখা শুরু করি, তখনই আমরা আইবিএমের ওয়াটসনের প্রতি আকৃষ্ট হই যেখানে তারা প্রাকৃতিক ভাষা প্রক্রিয়াকরণ (NLP) এবং মেশিন লার্নিংয়ের মতো বিগ ডেটা এবং কৃতিম বুদ্ধিমত্তা কৌশল ব্যবহার করে বিশ্বের সেরা দুই মানব জেওপার্টি! খেলোয়াড়কে পরাজিত করে। আমরা বুঝতে পেরেছিলাম যে ওয়াটসন সম্ভবত সিকোয়েল প্রেডিস্টের মতো সমস্যাগুলি পরিচালনা করতে পারবেন কারণ এটি বিশ্বের রাস্তার মানচিত্র এবং আরও অনেক কিছু দিয়ে পূর্ণ ছিল।

আজকের কৃত্রিম বুদ্ধিমত্তা এবং বিগ ডেটা সম্পর্কে গভীরভাবে অনুসন্ধান করার জন্য আমাদের আগ্রহ
প্রযুক্তি, এবং এই বইয়ের ১১-৬ নম্বর অধ্যায় গঠনে সাহায্য করেছে।

এটি উল্লেখযোগ্য যে, পর্ব ১১-৬ - এর সমস্ত ডেটাসায়েল বাস্তবায়ন কেস স্টাডি হয় কৃত্রিম বুদ্ধিমত্তা প্রযুক্তির উপর ভিত্তি করে তৈরি, অথবা
বিগ ডেটা হার্ডওয়্যার এবং সফ্টওয়্যার অবকাঠামো নিয়ে আলোচনা করা হয়েছে যা কম্পিউটার বিজ্ঞানী এবং ডেটা বিজ্ঞানীদের কার্যকরভাবে
উন্নত AI-ভিত্তিক সমাধান বাস্তবায়ন করতে সক্ষম করে।

কৃত্রিম বুদ্ধিমত্তা: সমস্যাযুক্ত কিন্তু সমাধানহীন একটি ক্ষেত্র

বহু দশক ধরে, AI কে এমন একটি ক্ষেত্র হিসেবে দেখা হয়ে আসছে যেখানে সমস্যা আছে কিন্তু কোন সমাধান নেই। কারণ একবার
কোন নির্দিষ্ট সমস্যা সমাধান হয়ে গেলে মানুষ বলে, "আচ্ছা, এটা বুদ্ধিমত্তা নয়, এটা কেবল একটি কম্পিউটার প্রোগ্রাম যা কম্পিউটারকে
ঠিক কী করতে হবে তা বলে দেয়।" যাইহোক, মেশিন লার্নিং (অধ্যায় 14) এবং ডিপ লার্নিং (অধ্যায় 15) এর মাধ্যমে আমরা নির্দিষ্ট
সমস্যার সমাধান প্রি-প্রোগ্রাম করছি না। পরিবর্তে, আমরা আমাদের কম্পিউটারগুলিকে ডেটা থেকে শেখার মাধ্যমে সমস্যার সমাধান করতে
দিচ্ছি - এবং, সাধারণত, এর অনেকগুলি।

গভীর শিক্ষার মাধ্যমে অনেক আকর্ষণীয় এবং চ্যালেঞ্জিং সমস্যা সমাধান করা হচ্ছে। গুগলের একাই হাজার হাজার গভীর
শিক্ষার প্রকল্প চলছে এবং এই সংখ্যা দ্রুত বৃদ্ধি পাচ্ছে। এই বইটি পড়ার সাথে সাথে, আমরা আপনাকে অনেক আধুনিক কৃত্রিম বুদ্ধিমত্তা, বিগ
ডেটা এবং ক্লাউড প্রযুক্তির ^{১,২} সাথে পরিচয় করিয়ে দেব।

^১ টিটিপি://theweek.com/speedreads/654463/googlemorethan1000

সরকারি গোয়েন্দা প্রকল্পের কাজ।

^২ [https://www.zdnet.com/article/googlesaysexponentialgrowthofai-](https://www.zdnet.com/article/googlesaysexponentialgrowthofai-generating-power-faster-than-ever/)

গণনার প্রকৃতি পরিবর্তন করা।

১.১০ র্যাপ-আপ

এই অধ্যায়ে, আমরা এমন পরিভাষা এবং ধারণাগুলি উপস্থাপন করেছি যা হ্যাপ্টার্স 2-০- এ আপনি যে পাইথন প্রোগ্রামিং শিখবেন
এবং হ্যাপ্টার্স 11-৬- এ আমরা যে বিগডেটা, কৃত্রিম বুদ্ধিমত্তা এবং ক্লাউডভিত্তিক কেস স্টাডি উপস্থাপন করব তার ভিত্তি স্থাপন করে।

আমরা অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ধারণাগুলি পর্যালোচনা করেছি এবং আলোচনা করেছি কেন পাইথন এত জনপ্রিয় হয়ে উঠেছে। আমরা পাইথন
স্ট্যান্ডার্ড লাইব্রেরি এবং বিভিন্ন ডেটাসায়েল লাইব্রেরি চালু করেছি যা আপনাকে "চাকা পুনর্বীকরণ" এড়াতে সাহায্য করে। পরবর্তী অধ্যায়গুলিতে,
আপনি এই লাইব্রেরি ব্যবহার করে সফ্টওয়্যার অবজেক্ট তৈরি করবেন যার সাথে আপনি অল্প সংখ্যক নির্দেশাবলীর সাথে গুরুত্বপূর্ণ কাজ সম্পাদন
করতে ইন্টারঅ্যাক্ট করবেন।

তুমি তিনটি টেস্টড্রাইভের মাধ্যমে কাজ করেছ যেখানে দেখানো হয়েছে কিভাবে IPython ইন্টারপ্রেটার এবং Jupyter
Notebooks ব্যবহার করে Python কোড এক্সিকিউট করতে হয়। আমরা ক্লাউড এবং ইন্টারনেট অফ থিংস (IoT) চালু
করেছি, যা তোমার জন্য সমসাময়িক অ্যাপ্লিকেশনগুলির ভিত্তি স্থাপন করেছে যা তুমি পরবর্তীতে তৈরি করবে।

Waze মোবাইল নেভিগেশন অ্যাপে একটি বিগড়েটা কেস স্টাডির প্রতি বিরক্তি প্রকাশ করছি, যা অনেক আধুনিক প্রযুক্তি ব্যবহার করে গতিশীল ড্রাইভিং দিকনির্দেশনা প্রদান করে যা আপনাকে যত তাড়াতাড়ি এবং যতটা সম্ভব নিরাপদে আপনার গন্তব্যে পৌঁছে দেয়। আমরা এই বইয়ের কোথায় আপনি এই প্রযুক্তিগুলির অনেকগুলি ব্যবহার করবেন তা উল্লেখ করেছি। আমাদের প্রথম ডেটা সায়েন্স ভূমিকা বিভাগের মাধ্যমে অধ্যায়টি শেষ হয়েছে যেখানে আমরা কম্পিউটার বিজ্ঞান এবং ডেটা বিজ্ঞানের মধ্যে একটি গুরুত্বপূর্ণ সংযোগস্থল - কৃত্রিম বুদ্ধিমত্তা - নিয়ে আলোচনা করেছি।

প্লেলিস্ট

গুরু. পাইথন প্রোগ্রামিং এর ভূমিকা

উদ্দেশ্যমূলক মতামত

এই অধ্যায়ে, আপনি পাবেন:

- কোড স্বিপেট প্রবেশ করাতে এবং তাদের ফলাফল দেখতে IPython ইন্টারেক্ষিভ মোড ব্যবহার চালিয়ে যান।
অফার্স এবং ডিল
অবিলম্বে।

Highlights সহজ পাইথন স্টেটমেন্ট এবং স্ট্রিপ্ট লিখুন।

etting শর্করাতী ব্যবহারের জন্য ডেটা সংরক্ষণের জন্য ডেরিয়েবল তৈরি করুন।

সমস্থ বিল্টইন ডেটা টাইপের সাথে পরিচিত হন।

- গাণিতিক অপারেটর এবং তুলনামূলক অপারেটর ব্যবহার করুন এবং তাদের সাইন আউট অগ্রাধিকার বুনুন।
- একক, দ্বিগুণ এবং তিনগুণ উদ্বৃত্ত স্ট্রিং ব্যবহার করুন।
- টেক্সট প্রদর্শনের জন্য বিল্টইন ফাংশন print ব্যবহার করুন।
- ব্যবহারকারীকে কীবোর্ডে ডেটা প্রবেশ করাতে এবং প্রোগ্রামে ব্যবহারের জন্য সেই ডেটা পেতে অনুরোধ করার জন্য বিল্টইন ফাংশন ইনপুট ব্যবহার করুন।
- বিল্টইন ফাংশন int ব্যবহার করে টেক্সটকে পূর্ণসংখ্যার মানে রূপান্তর করুন।
- একটি কার্যকর করাতে হবে কিনা তা নির্ধারণ করাতে তুলনা অপারেটর এবং if স্টেটমেন্ট ব্যবহার করুন বিবৃতি বা বিবৃতির দল।
- অবজেক্ট এবং পাইথনের ডাইনামিক টাইপিং সম্পর্কে জানুন।
- একটি বস্তুর ধরণ পেতে বিল্ট-ইন ফাংশন টাইপ ব্যবহার করুন

রূপরেখা

.১ ভূমিকা

.২ চলক এবং অ্যাসাইনমেন্ট বিবৃতি

.৩ পাটিগণিত

.৪ ফাংশন প্রিন্ট এবং সিঙ্গেল এবং ডাবল কোটেড স্ট্রিং-এর একটি ভূমিকা

.৫ ট্রিপল কোটেড স্ট্রিং

.৬ ব্যবহারকারীর কাছ থেকে ইনপুট নেওয়া

.৭ সিদ্ধান্ত গ্রহণ: if বিবৃতি এবং তুলনা অপারেটর

.৮ অবজেক্ট এবং ডায়নামিক টাইপিং

.৯ তথ্য বিজ্ঞানের ভূমিকা: মৌলিক বর্ণনামূলক পরিসংখ্যান

.১০ সারসংক্ষেপ

২.১ ভূমিকা

এই অধ্যায়ে, আমরা পাইথন প্রোগ্রামিং পরিচয় করিয়ে দেব এবং মূল ভাষার বৈশিষ্ট্যগুলি ব্যাখ্যা করার উদাহরণ উপস্থাপন করব। আমরা ধরে নিচ্ছি আপনি পর্ব ১-এ IPython TestDrive পড়েছেন, যেখানে IPython ইন্টারপ্রেটার চালু করা হয়েছিল এবং সহজ গাণিতিক রাশিগুলি মূল্যায়নের জন্য এটি ব্যবহার করা হয়েছিল।

২.২ পরিবর্তনশীল এবং নিয়োগ বিবৃতি

আপনি আইপিথনের ইন্টারেক্টিভ মোডকে ক্যালকুলেটর হিসেবে ব্যবহার করেছেন, যেমন এক্সপ্রেশন সহ

[1] তে: `45 + 72`

আউট[1]: ১১৭

চলুন x নামের একটি ভ্যারিয়েবল তৈরি করি যা পূর্ণসংখ্যা 7 সংরক্ষণ করে:

[2] তে: `x = 7`

স্বিপেট [2] হল একটি বিবৃতি। প্রতিটি বিবৃতি একটি কার্য সম্পাদনের জন্য নির্দিষ্ট করে। পূর্ববর্তী বিবৃতিটি x তৈরি করে এবং xa মান দেওয়ার জন্য অ্যাসাইনমেন্ট প্রতীক (=) ব্যবহার করে। বেশিরভাগ

স্টেটমেন্টগুলো লাইনের শেষে থেমে যায়, যদিও স্টেটমেন্টগুলো একাধিক লাইনে বিস্তৃত হতে পারে। নিম্নলিখিত স্টেটমেন্টটি *y* ভ্যারিয়েবল তৈরি করে এবং এতে মান নির্ধারণ করে

তো:

[3] তো: $y = 3$

আপনি এখন এক্সপ্রেশনে *x* এবং *y* এর মান ব্যবহার করতে পারেন:

[4] তো: $x + y$

আউট[4]: ১০

অ্যাসাইনমেন্ট স্টেটমেন্ট গণনা

নিম্নলিখিত বিবৃতিটি *x* এবং *y* ভেরিয়েবলের মান যোগ করে এবং ফলাফলটিকে ভেরিয়েবলের মোট সংখ্যায় নির্ধারণ করে, যা আমরা তারপর প্রদর্শন করব:

[5] তো: মোট = $x + y$

[6] তো: মোট

আউট[6]: ১০

= প্রতীকটি কোন অপারেটর নয়। = প্রতীকের ডান দিকটি সর্বদা প্রথমে কার্যকর হয়, তারপর ফলাফলটি প্রতীকের বাম দিকের ভেরিয়েবলে নির্ধারিত হয়।

পাইথন স্টাইল

পাইথন কোডের স্টাইল গাইড আপনাকে পাইথনের^১ কোডিং কনভেনশনের সাথে সঙ্গতিপূর্ণ কোড লিখতে সাহায্য করে।

স্টাইল গাইডটি প্রোগ্রামগুলিকে আরও পঠনযোগ্য করে তুলতে অ্যাসাইনমেন্ট প্রতীকের প্রতিটি পাশে একটি করে স্পেস = এবং বাইনারি অপারেটরগুলি + এর মতো সন্নিবেশ করার পরামর্শ দেয়।

^১ টিটিপিএস://www.python.org/dev/peps/pep0008/।

পরিবর্তনশীল নাম

একটি চলক নাম, যেমন *x*, একটি শনাক্তকারী। প্রতিটি শনাক্তকারী অক্ষর, সংখ্যা এবং আন্ডারস্কোর (_) দিয়ে তৈরি হতে পারে কিন্তু একটি অক্ষ দিয়ে শুরু নাও হতে পারে। পাইথন কেস সংবেদনশীল, তাই সংখ্যা এবং সংখ্যা ভিন্ন শনাক্তকারী কারণ একটি ছোট হাতের অক্ষর দিয়ে শুরু হয় এবং অন্যটি বড় হাতের অক্ষর দিয়ে শুরু হয়।

পাইথনে `ach` মানের একটি **ধরণ** থাকে যা মানটি যে ধরণের ডেটা উপস্থাপন করে তা নির্দেশ করে।

পাইথনের বিল্টইন **টাইপ** ফাংশনের সাহায্যে আপনি একটি মানের ধরণ দেখতে পারেন, যেমন:

[7] তে: টাইপ(x)

আউট[7]: int

[8] তে: **টাইপ(10.5)**

আউট[8]: ভাসমান

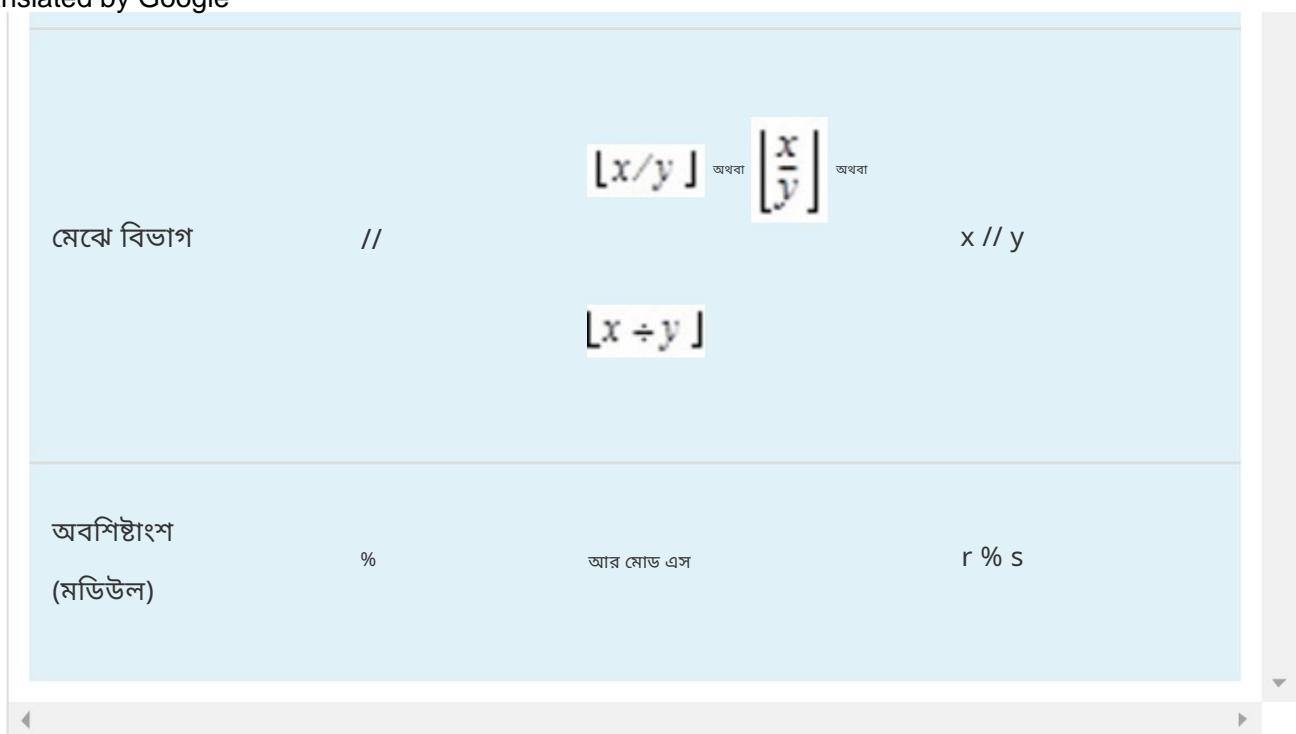
\times ভেরিয়েবলটিতে পূর্ণসংখ্যার মান 7 থাকে (মিপেট [2] থেকে), তাই পাইথন int প্রদর্শন করে

(পূর্ণসংখ্যার সংক্ষিপ্ত রূপ)। মান 10.5 একটি ফ্লোটিংপয়েন্ট সংখ্যা, তাই পাইথন প্রদর্শন করে ভাসমান।

২.৩ পাটিগণিত

নিম্নলিখিত সারণিতে **গাণিতিক অপারেটরগুলির সারসংক্ষেপ** দেওয়া হয়েছে, যার মধ্যে কিছু অন্তর্ভুক্ত রয়েছে বীজগণিতে ব্যবহৃত হয় না এমন প্রতীক।

পাইথন	পাটিগণিত	বীজগণিতীয়	পাইথন
অপারেশন	অপারেটর	অভিব্যক্তি	অভিব্যক্তি
সংযোজন	+	চ + ৭	চ + ৭
বিয়োগ	-	পু - গ	পি সি
গুণ	*	খ * মি	খ * মি
সূচকীকরণ	**	এক্স এক্স	এক্স এক্স
প্রকৃত বিভাজন	/	x/y অথবা $\frac{x}{y}$	$x \div y$ x / y



୭୭ (*)

পাইথন তারকাচিহ্ন (*) গুণন অপারেটর ব্যবহার করে:

[1] $\text{t} \leftarrow 7 * 4$

আউট[1]: ২৮

সূচকীকরণ (**)

সূচকীকরণ (**) অপারেটর একটি মানকে অন্যটির ঘাতে উন্নীত করে:

[2]: 2^{**10}

আউট[2]: ১০২৮

বর্গমূল গণনা করতে, আপনি সূচক $1/2$ (অর্থাৎ, 0.5) ব্যবহার করতে পারেন:

[3] തേ: 9 ** (1 / 2)

আউট[3]: 3.0

ট্রু ডিভিশন (/) বনাম ফ্লোর ডিভিশন (//)

সত্য ভাগ (/) একটি লবকে একটি হর দ্বারা ভাগ করে এবং দশমিক বিন্দু সহ একটি ভাসমান বিন্দু সংখ্যা উৎপন্ন করে,

[4] തേ: 7 / 4

আউট[4]: 1.75

তল ভাগ (//) একটি লবকে একটি হর দ্বারা ভাগ করে, যার ফলে সর্বোচ্চ পূর্ণসংখ্যা পাওয়া যায় যা ফলাফলের

চেয়ে বড় নয়। পাইথন ভগ্নাংশকে ছাঁটাই (বাদ) করে

অংশ:

[5]: `7 // 4` এ

আউট[5]: 1

[6]: `3 // 5` এ

আউট[6]: 0

[7]: `14 // 7` এ

আউট[7]: 2

প্রকৃত ভাগে, ১৩ কে ৪ দিয়ে ভাগ করলে ৩.২৫ পাওয়া যাবে:

[8] তে: `13 / 4`

আউট[8]: 3.25

তল বিভাজন 3.25 এর চেয়ে বড় নয় এমন নিকটতম পূর্ণসংখ্যা দেয়—যা 4:

[9]: `13 // 4` এ

আউট[9]: 4

ব্যতিক্রম এবং ট্রেসব্যাক

/ অথবা // দিয়ে শূন্য দিয়ে ভাগ করা অনুমোদিত নয় এবং এর ফলে একটি ব্যতিক্রম দেখা দেয়— একটি সমস্যা ঘটেছে এমন একটি চিহ্ন:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

[10] তে: `123 / 0`

জিরোভিডিশনএরর

```
ipython10cd759d3fcf39> <module>() এ
> ১ ১২৩ / ০
```

ট্রেসব্যাক (সর্বশেষ কলাটি শেষ

জিরোভিডিশনএরর: শূন্য দিয়ে ভাগ

পাইথন একটি ট্রেসব্যাক সহ একটি ব্যতিক্রম রিপোর্ট করে। এই ট্রেসব্যাকটি নির্দেশ করে যে ZeroDivisionError ধরণের একটি ব্যতিক্রম ঘটেছে—বেশিরভাগ ব্যতিক্রমের নাম ক্রটি দিয়ে শেষ হয়। ইন্টারেক্টিভ মোডে, ব্যতিক্রমের কারণ স্লিপেট নম্বর নির্দিষ্ট করা হয়।

```
<ipythonininput10cd759d3fcf39> <module>() এ
```

> দিয়ে শুরু হওয়া লাইনটি সেই কোডটি দেখায় যা ব্যতিক্রমটি তৈরি করেছে। কখনও কখনও স্লিপেটে একাধিক লাইন কোড থাকে
- > এর ডানদিকের 1 নির্দেশ করে যে স্লিপেটের মধ্যে থাকা 1 নম্বর লাইনটি ব্যতিক্রমটি তৈরি করেছে। শেষ লাইনটি ঘটে যাওয়া
ব্যতিক্রমটি দেখায়, তারপরে একটি কোলন (:) এবং ব্যতিক্রম সম্পর্কে আরও তথ্য সহ একটি ক্রটি বার্তা:

জিরোডিভিশনএর: শুন্য দিয়ে ভাগ

"ফাইল এবং ব্যতিক্রম" অধ্যায়ে ব্যতিক্রমগুলি বিস্তারিতভাবে আলোচনা করা হয়েছে।

আপনি যদি এমন একটি ভেরিয়েবল ব্যবহার করার চেষ্টা করেন যা আপনি এখনও তৈরি করেননি, তাহলে একটি ব্যতিক্রমও ঘটে।
নিম্নলিখিত স্লিপেটটি অনির্ধারিত ভেরিয়েবল z-এ 7 যোগ করার চেষ্টা করে, যার ফলে একটি NameError তৈরি হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

এবং [11]: + 7 সহ

নাম ক্রটি

ট্রেসব্যাক (সর্বশেষ কলটি শেষ

<module>() এ ipythonininput11f2cdbc4fe75d>

> ১ z + ৭

নাম ক্রটি: 'z' নামটি সংজ্ঞায়িত করা হয়নি

অবশিষ্ট অপারেটর

বাম অপারেন্টকে ডান অপারেন্ট দ্বারা ভাগ করার পর পাইথনের **অবশিষ্টাংশ অপারেটর (%)** অবশিষ্টাংশ প্রদান করে:

[12] তে: ১৭ % ৫

আউট[12]: ২

এই ক্ষেত্রে, ১৭ কে ৫ দিয়ে ভাগ করলে ৩ এর ভাগফল এবং ২ এর অবশিষ্টাংশ পাওয়া যায়। এই অপারেটরটি সাধারণত পূর্ণসংখ্যার
সাথে ব্যবহৃত হয়, তবে অন্যান্য সংখ্যাসূচক ধরণের সাথেও ব্যবহার করা যেতে পারে:

[13] তে: 7.5 % 3.5

আউট[13]: 0.5

সরলরেখা ফর্ম

বীজগণিতীয় স্বরলিপি যেমন

a
b

সাধারণত কম্পাইলার বা ইন্টারপ্রেটারদের কাছে গ্রহণযোগ্য নয়। এই কারণে, পাইথনের অপারেটর ব্যবহার করে বীজগণিতীয় রাশিগুলিকে **সরলরেখায়** টাইপ করতে হবে। উপরের রাশিটি a / b (অথবা তল বিভাগের জন্য $a // b$) হিসাবে লিখতে হবে যাতে সমস্ত অপারেটর এবং অপারেন্ট একটি অনুভূমিক সরলরেখায় উপস্থিত হয়।

বন্ধনীর সাহায্যে রাশিগুলিকে গোষ্ঠীবদ্ধ করা

বন্ধনীগুলি পাইথন এক্সপ্রেশনগুলিকে গোষ্ঠীভুক্ত করে, যেমনটি বীজগণিতীয় এক্সপ্রেশনগুলিতে করা হয়।
উদাহরণস্বরূপ, নিম্নলিখিত কোডটি $5 + 3$ পরিমাণের 10 গুণ করে:

[14] তে: $10 * (5 + 3)$

আউট[14]: 80

এই বন্ধনী ছাড়া, ফলাফল ভিন্ন:

[15] তে: $10 * 5 + 3$

আউট[15]: 53

বন্ধনীগুলি **অপ্রয়োজনীয়** (অপ্রয়োজনীয়) যদি সেগুলি অপসারণ করলে একই ফলাফল পাওয়া যায় ফলাফল।

অপারেটর অগ্রাধিকার নিয়ম

পাইথন অপারেটর অগ্রাধিকারের নিম্নলিখিত নিয়ম অনুসারে গাণিতিক রাশিতে অপারেটর প্রয়োগ করে। এগুলি সাধারণত বীজগণিতের মতোই:

1. বন্ধনীর রাশিগুলি প্রথমে মূল্যায়ন করে, তাই বন্ধনীগুলি ক্রমকে জোর করে নির্দেশ করতে পারে আপনার ইচ্ছামত যেকোনো ক্রমানুসারে মূল্যায়ন করা হবে। বন্ধনীর সর্বোচ্চ স্তরের অগ্রাধিকার রয়েছে। **নেক্স্টেড বন্ধনীযুক্ত রাশিগুলিতে**, যেমন $(a / (b c))$, সবচেয়ে ভেতরের বন্ধনীর (অর্থাৎ, $b c$) রাশিটি প্রথমে মূল্যায়ন করা হয়।
2. সূচকীকরণের ক্রিয়াকলাপগুলি পরবর্তী মূল্যায়ন করে। যদি একটি রাশিতে একাধিক সূচকীকরণের ক্রিয়াকলাপ থাকে, তাহলে পাইথন সেগুলি ডান থেকে বামে প্রয়োগ করে।
3. গুণ, ভাগ এবং মডুলাস ক্রিয়াকলাপ পরবর্তী মূল্যায়ন করে। যদি একটি রাশি

পাইথনে বেশ কয়েকটি গুণ, সত্য ভাগ, তল ভাগ এবং মডুলাস অপারেশন রয়েছে, পাইথন এগুলি
বাম থেকে ডানে প্রয়োগ করে। গুণ, ভাগ এবং মডুলাস "একই স্তরের অগ্রাধিকারে"।

৪. যোগ এবং বিয়োগের ক্রিয়াকলাপগুলি শেষ মূল্যায়ন করে। যদি একটি রাশিতে বেশ কয়েকটি যোগ এবং বিয়োগের
ক্রিয়াকলাপ থাকে, তাহলে পাইথন সেগুলি বাম থেকে ডানে প্রয়োগ করে।
যোগ এবং বিয়োগেরও একই স্তরের অগ্রাধিকার রয়েছে।

অপারেটরদের সম্পূর্ণ তালিকা এবং তাদের অগ্রাধিকারের জন্য (সর্বনিম্ন থেকে সর্বোচ্চ ক্রমে), দেখুন

<https://docs.python.org/3/reference/expressions.html#operatorprecedence>

অপারেটর গ্রুপিং

যখন আমরা বলি যে পাইথন বাম থেকে ডানে নির্দিষ্ট অপারেটর প্রয়োগ করে, তখন আমরা অপারেটরদের গ্রুপিংকে
উল্লেখ করছি। উদাহরণস্বরূপ, এক্সপ্রেশনে

ক + খ + গ

যোগ অপারেটর (+) গ্রুপ বাম থেকে ডানে, যেন আমরা এক্সপ্রেশনটিকে $(a + b) + c$ হিসাবে বন্ধনীবদ্ধ করেছি। একই
অগ্রাধিকার গ্রুপের সমস্ত পাইথন অপারেটর lefttoright, exponentiation অপারেটর (**) ব্যতীত, যা
righttoleft গ্রুপ করে।

অপ্রয়োজনীয় বন্ধনী

আপনি অপ্রয়োজনীয় বন্ধনী ব্যবহার করে উপ-অভিযন্তাগুলিকে গোষ্ঠীভুক্ত করতে পারেন যাতে রাশিটি স্পষ্ট হয়।
উদাহরণস্বরূপ, দ্বিতীয়-ডিগ্রি বহুপদী

$y = a^* + b^{**} 2 + c *$

স্পষ্টতার জন্য, বন্ধনীতে রাখা যেতে পারে, যেমন

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

$y = (a * (x ** 2)) + (b * x) + c$

জটিল রাশিগুলিকে ছোট, সরল রাশি দিয়ে বাকের ক্রমানুসারে বিভক্ত করলেও স্পষ্টতা বৃদ্ধি পেতে পারে।

অপারেন্ডের ধরণ

প্রতিটি গাণিতিক অপারেটর পূর্ণসংখ্যা এবং ফ্লোটিংপয়েন্ট সংখ্যার সাথে ব্যবহার করা যেতে পারে। যদি উভয় অপারেন্টই পূর্ণসংখ্যা হয়, তাহলে ফলাফলটি একটি পূর্ণসংখ্যা হবে—সত্যবিভাগ (/) অপারেটর ব্যতীত, যা সর্বদা একটি ফ্লোটিংপয়েন্ট সংখ্যা প্রদান করে। যদি উভয় অপারেন্টই ফ্লোটিংপয়েন্ট সংখ্যা প্রদান করে, তাহলে ফলাফলটি একটি ফ্লোটিংপয়েন্ট সংখ্যা প্রদান করে। একটি পূর্ণসংখ্যা এবং একটি ফ্লোটিংপয়েন্ট সংখ্যা ধারণকারী রাশিগুলি মিশ্র ধরণের রাশি - এগুলি সর্বদা ফ্লোটিং-পয়েন্ট ফলাফল প্রদান করে।

২.৪ ফাংশন প্রিন্ট এবং একক এবং দ্বিগুণ উদ্ভৃত স্ট্রিংগুলির ভূমিকা

বিল্টইন **প্রিন্ট** ফাংশনটি তার আর্গুমেন্ট(গুলি) টেক্সটের একটি লাইন হিসেবে প্রদর্শন করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: print("পাইথনে স্বাগতম!")
```

পাইথনে স্বাগতম!

এই ক্ষেত্রে, 'পাইথনে স্বাগতম!' আর্গুমেন্টটি একটি স্ট্রিং - একক উদ্ভৃতি ('') দিয়ে আবদ্ধ অক্ষরের একটি ক্রম। ইন্টারেক্টিভ মোডে এক্সেপশন মূল্যায়ন করার সময়, এখানে প্রিন্ট করা টেক্সটটি Out[1] দ্বারা পূর্বে প্রদর্শিত হয় না। এছাড়াও, প্রিন্ট কোনও স্ট্রিংয়ের উদ্ভৃতি প্রদর্শন করে না, যদিও আমরা শীঘ্রই দেখাব কিভাবে স্ট্রিংগুলিতে উদ্ভৃতি প্রদর্শন করতে হয়।

আপনি একটি স্ট্রিংকে ডাবল কোটেশন ("") দিয়েও সংযুক্ত করতে পারেন, যেমন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[2] তে: print("পাইথনে স্বাগতম!")
```

পাইথনে স্বাগতম!

পাইথন প্রোগ্রামাররা সাধারণত একক উদ্ভৃতি পছন্দ করে। প্রিন্ট যখন তার কাজ সম্পন্ন করে, তখন এটি পরবর্তী লাইনের শুরুতে স্ক্রিন কার্সারটি স্থাপন করে।

কমা দ্বারা পৃথক করা আইটেমের তালিকা মুদ্রণ করা

প্রিন্ট ফাংশনটি কমা দ্বারা পৃথক করা আর্গুমেন্টের একটি তালিকা পেতে পারে, যেমন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[3] তে: print('welcome', 'to', 'Python!')
```

পাইথনে স্বাগতম!

t প্রতিটি আর্গমেন্টকে একটি স্পেস দিয়ে আলাদা করে প্রদর্শন করে, যা পূর্ববর্তী দুটি স্লিপেটের মতো একই আউটপুট তৈরি করে। এখানে আমরা কমা দ্বারা পৃথক করা স্ট্রিংগুলির একটি তালিকা দেখিয়েছি, তবে মানগুলি যেকোনো ধরণের হতে পারে। আমরা পরবর্তী অধ্যায়ে দেখাব কিভাবে মানগুলির মধ্যে স্বয়ংক্রিয় ব্যবধান রোধ করা যায় অথবা স্পেসের চেয়ে ভিন্ন বিভাজক ব্যবহার করা যায়।

একটি বিবৃতি দিয়ে অনেক লাইন লেখা মুদ্রণ করা

যখন একটি স্ট্রিং-এ ব্যাকস্যাশ (\) প্রদর্শিত হয়, তখন এটিকে এঙ্কেপ ক্যারেক্টার বলা হয়। ব্যাকস্যাশ এবং তার সাথে সাথেই অক্ষরটি একটি এঙ্কেপ সিকোয়েল তৈরি করে। উদাহরণস্বরূপ, \n নিউলাইন ক্যারেক্টার এঙ্কেপ সিকোয়েল উপস্থাপন করে, যা প্রিন্টকে আউটপুট কার্সারটিকে পরবর্তী লাইনে সরাতে বলে। নিম্নলিখিত স্লিপেটে তিনটি নিউলাইন ক্যারেক্টার ব্যবহার করে আউটপুটের বেশ কয়েকটি লাইন তৈরি করা হয়েছে: i

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[4] তে: print('Welcome\n to\n\nPython!')
```

স্বাগতম

থেকে

পাইথন!

অন্যান্য পালানোর ক্রম

নিম্নলিখিত টেবিলে কিছু সাধারণ পালানোর ক্রম দেখানো হয়েছে।

পালানো
ক্রম

বিবরণ

\n
একটি স্ট্রিং-এ একটি নতুন লাইনের অক্ষর সন্নিবেশ করান। যখন স্ট্রিংটি প্রদর্শিত হবে, প্রতিটি নতুন লাইনের জন্য, স্ক্রিন কার্সারটিকে পরবর্তী লাইনের শুরুতে নিয়ে যান।

\t
একটি অনুভূমিক ট্যাব ঢোকান। স্ট্রিংটি প্রদর্শিত হলে, প্রতিটি ট্যাবের জন্য, স্ক্রিন কার্সারটিকে পরবর্তী ট্যাব স্টেপে নিয়ে যান।

\\br
একটি স্ট্রিং-এ একটি ব্যাকস্যাশ অক্ষর সন্নিবেশ করান।

```
\"
    একটি স্ট্রিং-এ একটি ডবল কোট অক্ষর সন্নিবেশ করান।
```

```
\'
    একটি স্ট্রিং-এ একটি একক উদ্ধৃতি অক্ষর সন্নিবেশ করান।
```

লম্বা স্ট্রিংয়ে লাইন ব্রেক উপেক্ষা করা

আপনি লাইন ব্রেক উপেক্ষা করার জন্য \ continuation অক্ষরটিকে একটি লাইনের শেষ অক্ষর হিসেবে ব্যবহার করে একটি দীর্ঘ স্ট্রিং (অথবা একটি দীর্ঘ বিবৃতি) কে কয়েকটি লাইনে বিভক্ত করতে পারেন :

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[5] তে: print('এটি দীর্ঘ স্ট্রিং, তাই আমরা \
...: এটিকে দুটি লাইনে ভাগ করুন')
এটি একটি দীর্ঘ স্ট্রিং, তাই আমরা এটিকে দুটি লাইনে ভাগ করেছি।
```

ইন্টারপ্রেটার স্ট্রিং এর অংশগুলিকে কোন লাইন ব্রেক ছাড়াই একটি একক স্ট্রিংয়ে পুনরায় একত্রিত করে।

যদিও পূর্ববর্তী স্লিপেটের ব্যাকস্ল্যাশ অক্ষরটি একটি স্ট্রিংয়ের ভিতরে রয়েছে, এটি এস্কেপ অক্ষর নয় কারণ অন্য কোনও অক্ষর এটি অনুসরণ করে না।

একটি রাশির মান মুদ্রণ করা

মুদ্রিত বিবৃতিতে গণনা করা যেতে পারে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[6] তে: print('যোগফল হল', 7 + 3)
যোগফল ১০
```

২.৫ ট্রিপল-কোটেড স্ট্রিং

এর আগে, আমরা একক উদ্ধৃতি ('') অথবা দ্বিগুণ উদ্ধৃতি ("") দিয়ে বিভক্ত স্ট্রিং চালু করেছিলাম। **ট্রিপলউদ্ধৃত স্ট্রিংগুলি** তিনটি দ্বিগুণ উদ্ধৃতি (""""") অথবা তিনটি একক উদ্ধৃতি ('') দিয়ে শুরু এবং শেষ হয়। পাইথন কোডের স্টাইল গাইড তিনটি দ্বিগুণ উদ্ধৃতি (""""") সুপারিশ করে। তৈরি করতে এগুলি ব্যবহার করুন:

- বহুরেখা স্ট্রিং,

- একক বা দ্বিগুণ উদ্ধৃতি চিহ্ন ধারণকারী স্ট্রিং এবং
- ডকস্ট্রিং, যা নির্দিষ্ট কিছু উদ্দেশ্য নথিভুক্ত করার প্রস্তাবিত উপায় প্রোগ্রামের উপাদান।

স্ট্রিং-এ উদ্ধৃতি অন্তর্ভুক্ত করা

একক উদ্ধৃতি দ্বারা সীমাবদ্ধ একটি স্ট্রিংয়ে, আপনি ডাবলউদ্ধৃতি অক্ষর অন্তর্ভুক্ত করতে পারেন:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[1] তে: print('উদ্ধৃতি চিহ্নে "hi" প্রদর্শন করুন')
      উদ্ধৃতি চিহ্নে "হাই" প্রদর্শন করুন
```

কিন্তু একক উদ্ধৃতি নয়:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[2] তে: print('উদ্ধৃতি চিহ্নে 'হাই' প্রদর্শন করুন')
      ফাইল "<ipythoninput219bf596ccf72>", লাইন ১
      print('উদ্ধৃতি চিহ্নে 'হাই' দেখান')
      ^
      সিনট্যাক্স ত্রুটি: অবৈধ সিনট্যাক্স
```

যদি না আপনি \। এক্সপে সিকোয়েল ব্যবহার করেন:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[3] তে: print('উদ্ধৃতিচিহ্নে \'hi\' প্রদর্শন করুন')
      উদ্ধৃতি চিহ্নে 'হাই' প্রদর্শন করুন
```

একটি একক উদ্ধৃতিযুক্ত স্ট্রিংয়ের ভিতরে একটি একক উদ্ধৃতি থাকার কারণে স্লিপেট [2] একটি বাক্য গঠন ত্রুটি প্রদর্শন করেছে। আইপিথন কোডের লাইন সম্পর্কে তথ্য প্রদর্শন করে যা সিনট্যাক্স ত্রুটির কারণ হয়েছিল এবং ^ চিহ্ন দিয়ে ত্রুটিটি নির্দেশ করে। এটি SyntaxError বার্তাটিও প্রদর্শন করে:

অবৈধ বাক্য গঠন।

ডাবল কোটেশন দ্বারা সীমাবদ্ধ একটি স্ট্রিংয়ে একক কোটেশন অক্ষর থাকতে পারে:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[4] তে: print("O'Brien নামটি প্রদর্শন করুন")
```

ও'ব্ৰেইন নামটি প্রদর্শন কৰুন

কিন্তু ডাবল কোটেশন নয়, যদি না আপনি।" এক্সেপ সিকোয়েল ব্যবহার কৰেন:

কোড ইমেজ দেখতে এখানে ক্লিক কৰুন

```
[5] তে: print("উদ্ধৃতি চিহ্নে \"hi\" প্রদর্শন করুন")
```

উদ্ধৃতি চিহ্নে "হাই" প্রদর্শন কৰুন

স্ট্রিং এৰ ভেতৱে ' এবং ' ব্যবহার এড়াতে, আপনি এই ধৰণের স্ট্রিংগুলিকে ট্ৰিপল কোটেশনে আবদ্ধ কৰতে পাৱেন:

কোড ইমেজ দেখতে এখানে ক্লিক কৰুন

```
[6] তে: print("""উদ্ধৃতি চিহ্নে "hi" এবং 'bye' প্রদর্শন করুন""")
```

উদ্ধৃতি চিহ্নে "হাই" এবং 'বাই' প্রদর্শন কৰুন

মাল্টিলাইন স্ট্রিং

নিম্নলিখিত স্লিপেটটি একটি বহুরেখা ট্ৰিপলকোটেড স্ট্রিংকে বৰাদ্ব কৰে

ট্ৰিপল_কোটেড_স্ট্রিং:

কোড ইমেজ দেখতে এখানে ক্লিক কৰুন

```
[7] তে: triple_quoted_string = """এটি একটি triplequoted
```

.... দুটি লাইন জুড়ে বিস্তৃত স্ট্রিং"""

IPython জানে যে স্ট্রিংটি অসম্পূর্ণ কাৰণ আমৰা Enter চাপাৰ আগে """" শেষেৰ অংশটি টাইপ কৱিনি। সুতৰাং, IPython একটি ধাৰাবাহিকতা প্ৰম্পট প্ৰদৰ্শন কৰে: যেখানে আপনি বহু-লাইন স্ট্রিংয়েৰ পৰবৰ্তী লাইনটি ইনপুট কৰতে পাৱেন। এটি চলতে থাকে যতক্ষণ না আপনি শেষেৰ অংশ """" প্ৰবেশ কৰেন এবং Enter টিপুন। নিম্নলিখিতটি triple_quoted_string প্ৰদৰ্শন কৰে:

কোড ইমেজ দেখতে এখানে ক্লিক কৰুন

```
[8] তে: print(triple_quoted_string)
```

এটি একটি ত্ৰিকোণি

দুটি লাইন জুড়ে বিস্তৃত স্ট্রিং

পাইথন এমবেডেড নিউলাইন অক্ষৰ সহ মাল্টিলাইন স্ট্রিং সংৰক্ষণ কৰে। যখন আমৰা triple_quoted_string প্ৰিন্ট কৱাৰ পৱিতৰে মূল্যায়ন কৱি, তখন IPython এটিকে একক উদ্ধৃতিতে প্ৰদৰ্শন কৰে।

স্লিপেট [7]-এ Enter চাপার সময় একটি \n অক্ষর ব্যবহার করা হয়েছে। IPython-এ প্রদর্শিত উদ্ধৃতিগুলি নির্দেশ করে যে triple_quoted_string একটি স্ট্রিং—এগুলি স্ট্রিংয়ের বিষয়বস্তুর অংশ নয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[9] তে: triple_quoted_string
Out[9]: 'এটি একটি ট্রিপলকোটেড\nস্ট্রিং যা দুটি লাইন জুড়ে বিস্তৃত'
```

২.৬ ব্যবহারকারীর কাছ থেকে ইনপুট নেওয়া

বিল্টইন ইনপুট ফাংশনটি ব্যবহারকারীর ইনপুট অনুরোধ করে এবং গ্রহণ করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: নাম = ইনপুট ("তোমার নাম কি? ")
তোমার নাম কি? পল।

[2] তে: নাম
আউট[2]: 'পল'

[3] তে: মুদ্রণ(নাম)
পল
```

স্লিপেটটি নিষ্ঠাপন কার্যকর হয়:

- প্রথমে, ইনপুট তার স্ট্রিং আর্গুমেন্ট - একটি প্রম্পট - প্রদর্শন করে যা ব্যবহারকারীকে কী টাইপ করতে হবে তা বলে এবং ব্যবহারকারীর প্রতিক্রিয়া জানার জন্য অপেক্ষা করে। আমরা Paul টাইপ করে Enter টিপেছি। ইনপুট প্রদর্শিত প্রম্পট টেক্সট থেকে ব্যবহারকারীর ইনপুট আলাদা করতে আমরা বোল্ড টেক্সট ব্যবহার করি।
- ফাংশন ইনপুট তারপর সেই অক্ষরগুলিকে একটি স্ট্রিং হিসেবে ফেরত পাঠায় যা প্রোগ্রামটি ব্যবহার করতে পারে। এখানে আমরা সেই স্ট্রিংটি ভেরিয়েবলের নামের সাথে যুক্ত করেছি।

স্লিপেট [2] নামের মান দেখায়। নাম মূল্যায়ন করলে এর মান একক উদ্ধৃতিতে 'পল' হিসেবে প্রদর্শিত হয় কারণ এটি একটি স্ট্রিং। (স্লিপেট [3]) নাম মুদ্রণ করলে উদ্ধৃতি চিহ্ন ছাড়াই স্ট্রিং প্রদর্শিত হয়। যদি আপনি উদ্ধৃতি চিহ্ন প্রবেশ করান, তাহলে সেগুলি স্ট্রিংয়ের অংশ, যেমন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[4] তে: নাম = ইনপুট ("তোমার নাম কি? ")
তোমার নাম কি? 'পল'।
```

[5] তে: নাম
আউট[5]: "'পল'"

[6] তে: মুদ্রণ(নাম)
'পল'

ফাংশন ইনপুট সর্বদা একটি স্ট্রিং প্রদান করে

দুটি সংখ্যা পড়ে যোগ করার চেষ্টা করা নিচের কিছু অংশ বিবেচনা করুন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[7] তে: value1 = `input('প্রথম সংখ্যা লিখুন: ')`
প্রথম সংখ্যাটি লিখুন: ৭

[8] তে: value2 = `input('দ্বিতীয় সংখ্যা লিখুন: ')`
দ্বিতীয় সংখ্যাটি লিখুন: ৩

[9] তে: মান1 + মান2
আউট[9]: '73'

৭ এবং ৩ সংখ্যা যোগ করে ১০ তৈরি করার পরিবর্তে, পাইথন স্ট্রিং মান '৭' এবং '৩' "যোগ" করে, যার ফলে '৭৩' স্ট্রিং তৈরি হয়। এটি স্ট্রিং কনক্যাটেশন নামে পরিচিত। এটি একটি নতুন স্ট্রিং তৈরি করে যার মধ্যে বাম অপারেন্ডের মান থাকে এবং তারপর ডান অপারেন্ডের মান থাকে।

মান।

ব্যবহারকারীর কাছ থেকে একটি পূর্ণসংখ্যা পাওয়া

যদি আপনার একটি পূর্ণসংখ্যার প্রয়োজন হয়, তাহলে `builtin int` ফাংশন ব্যবহার করে স্ট্রিংটিকে একটি পূর্ণসংখ্যায় রূপান্তর করুন :

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[10] তে: মান = `ইনপুট ('একটি পূর্ণসংখ্যা লিখুন: ')`
একটি পূর্ণসংখ্যা লিখুন: ৭

[11] তে: মান = `int(মান)`

[12] তে: মান
আউট[12]: ৭

আমরা কোডটি স্মিপেট [10] এবং [11] তে একত্রিত করতে পারতাম:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[13] তে: another_value = int(input("আরেকটি পূর্ণসংখ্যা লিখুন: "))
আরেকটি পূর্ণসংখ্যা লিখুন: ১৩
```

```
[14] তে: অন্য_মান
আউট[14]: 13
```

ভ্যারিয়েবলের value এবং another_value-এ এখন পূর্ণসংখ্যা রয়েছে। এগুলো যোগ করলে একটি পূর্ণসংখ্যার ফলাফল তৈরি হয় (এগুলোকে সংযুক্ত করার পরিবর্তে):

```
[15] তে: মান + অন্য_মান
আউট[15]: 20
```

যদি int-এ পাস করা স্ট্রিংটি পূর্ণসংখ্যায় রূপান্তরিত না হয়, তাহলে একটি ValueError ঘটে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[16] তে: bad_value = int(input('আরেকটি পূর্ণসংখ্যা লিখুন: '))
আরেকটি পূর্ণসংখ্যা লিখুন: হ্যালো
```

মান ত্রুটি
ipythoninput16cd36e6cf8911> <module>() এ
> ১টি খারাপ_মান = int(input('আরেকটি পূর্ণসংখ্যা লিখুন: '))
ValueError: বেস ১০ সহ int() এর জন্য অবৈধ আক্ষরিক: 'হ্যালো'

ট্রেসব্যাক (সর্বশেষ কলটি শেষ

ফাংশন int একটি ফ্লোটিংপয়েন্ট মানকে একটি পূর্ণসংখ্যায় রূপান্তর করতে পারে:

```
[17] তে: int(10.5)
আউট[17]: 10
```

স্ট্রিংগুলিকে ফ্লোটিংপয়েন্ট সংখ্যায় রূপান্তর করতে, বিল্টইন ফ্লোট ফাংশনটি ব্যবহার করুন।

২.৭ সিদ্ধান্ত গ্রহণ: যদি বিবৃতি এবং তুলনা অপারেটর

একটি **শর্ত** হলো একটি বুলিয়ান এক্সপ্রেশন যার মান **True** অথবা **False**। নিম্নলিখিতটি নির্ধারণ করে যে 7 4 এর চেয়ে বড় কিনা এবং 7 4 এর চেয়ে ছোট কিনা:

```
[1]: 7 > 4
আউট[1]: সত
```

সত্য এবং মিথ্যা হল পাইথন কীওয়ার্ড। একটি কীওয়ার্ডকে শনাক্তকারী হিসাবে ব্যবহার করলে একটি সিনট্যাক্স ত্রুটি। সত্য এবং মিথ্যা উভয়ই বড় হাতের অক্ষরে লেখা।

আপনি প্রায়শই নিম্নলিখিত তুলনামূলক অপারেটরগুলি ব্যবহার করে শর্ত তৈরি করবেন টেবিল:

বীজগণিতীয় অপারেটর	পাইথন অপারেটর	নমুনা অবস্থা	অর্থ
>	>	$x > y$	x, y এর চেয়ে বড়
<>	<>	$x < y$	x, y এর চেয়ে ছোট
□	\geq	$x \geq y$	x , অথবা এর চেয়ে বড় y এর সমান
□	\leq	$x \leq y$	x এর চেয়ে কম বা সমান
=	$==$	$x == y$	x হল y এর সমান
≠	$!=$	$x != y$	x, y এর সমান নয়।

অপারেটর $>$, $<$, \geq এবং \leq সকলেরই একই অগ্রাধিকার। অপারেটর $==$ এবং $!=$ উভয়ই একই অগ্রাধিকার আছে, যা $>$, $<$, \geq এবং \leq এর চেয়ে কম। একটি বাক্য গঠন ত্রুটি যখন কোন অপারেটর $==$, $!=$, $>$ এবং $<$ এর জোড়ার মধ্যে ফাঁকা স্থান ধারণ করে তখন ঘটে।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3] তে: 7 > = 4

"<ipythonininput35c6e2897f3b3>" ফাইল, লাইন ১

q > = 8

^

সিনট্যাক্স ত্রুটি: অবৈধ সিনট্যাক্স

অপারেটরগুলিতে চিহ্নগুলি উল্টে দিলে ($=!$, $=>$ এবং $=<$ লিখে) আরেকটি সিনট্যাক্স ত্রুটি দেখা দেয়।

if বিবৃতি ব্যবহার করে সিদ্ধান্ত নেওয়া: স্ক্রিপ্টের সাথে পরিচয় করিয়ে দেওয়া

আমরা এখন if স্টেটমেন্টের একটি সহজ সংস্করণ উপস্থাপন করছি , যা একটি শর্ত ব্যবহার করে একটি স্টেটমেন্ট (অথবা স্টেটমেন্টের একটি ছপ্প) কার্যকর করার সিদ্ধান্ত নেবে। এখানে আমরা ব্যবহারকারীর দুটি পূর্ণসংখ্যা পড়ব এবং পরপর ছয়টি if স্টেটমেন্ট ব্যবহার করে তাদের তুলনা করব, প্রতিটি তুলনা অপারেটরের জন্য একটি করে। যদি প্রদত্ত if স্টেটমেন্টের শর্তটি True হয়, তাহলে সংশ্লিষ্ট মুদ্রিত স্টেটমেন্টটি কার্যকর হবে; অন্যথায়, এটি এডিয়ে যাবে।

সংক্ষিপ্ত কোড স্লিপেটগুলি কার্যকর করার জন্য এবং তাৎক্ষণিক ফলাফল দেখার জন্য IPython ইন্টারেক্শন মোড সহায়ক।

যখন আপনার কাছে অনেকগুলি বিবৃতি একটি ছপ্প হিসাবে কার্যকর করার জন্য থাকে, তখন আপনি সাধারণত সেগুলিকে .py (পাইথনের সংক্ষিপ্ত রূপ) এক্সেনশন সহ একটি ফাইলে সংরক্ষিত স্ক্রিপ্ট হিসাবে লিখবেন—যেমন এই উদাহরণের স্ক্রিপ্টের জন্য fig02_01.py।
স্ক্রিপ্টগুলিকে প্রোগ্রামও বলা হয়। এই বইতে স্ক্রিপ্টগুলি সনাক্ত এবং কার্যকর করার নির্দেশাবলীর জন্য, chapter 1 এর IPython
দেখুন।
টেস্টড্রাইভ।

প্রতিবার এই স্ক্রিপ্টটি কার্যকর করার সময়, ছয়টি শর্তের মধ্যে তিনটি সত্য। এটি দেখানোর জন্য, আমরা স্ক্রিপ্টটি তিনবার কার্যকর করব -
একবার প্রথম পূর্ণসংখ্যা দ্বিতীয়টির চেয়ে কম হলে, একবার উভয় পূর্ণসংখ্যার জন্য একই মান হলে এবং একবার দ্বিতীয়টির চেয়ে বড়
হলে। তিনটি নমুনা কার্যকরকরণ স্ক্রিপ্টের পরে প্রদর্শিত হবে।

প্রতিবার যখন আমরা নীচের স্ক্রিপ্টের মতো একটি স্ক্রিপ্ট উপস্থাপন করি, তখন আমরা চিত্রের আগে এটি পরিচয় করিয়ে দিই, তারপর চিত্রের
পরে স্ক্রিপ্টের কোড ব্যাখ্যা করি। আপনার সুবিধার্থে আমরা লাইন নম্বরগুলি দেখাই - এগুলি পাইথনের অংশ নয়। IDE গুলি আপনাকে লাইন
নম্বরগুলি প্রদর্শন করবে কিনা তা চয়ন করতে সক্ষম করে। এই উদাহরণটি চালানোর জন্য, এই অধ্যায়ের ch02 উদাহরণ
ফোল্ডারে পরিবর্তন করুন, তারপর

প্রবেশ করান:

আইপাইথন fig02_01.py

অথবা, যদি আপনি ইতিমধ্যেই IPython-এ থাকেন, তাহলে আপনি এই কমান্ডটি ব্যবহার করতে পারেন:

fig02_01.py চালান

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```

১ # fig02_01.py ২ """if
স্টেটমেন্ট এবং তুলনা অপারেটর ব্যবহার করে পূর্ণসংখ্যার তুলনা করা।"""
৩
৪ printf("দুটি পূর্ণসংখ্যা লিখুন, আমি আপনাকে বলব",
৫           'তারা যে সম্পর্কগুলিকে সন্তুষ্ট করে।')
৬
৭ # প্রথম পূর্ণসংখ্যা পড়ুন ৮ সংখ্যা১ =
  int(input('প্রথম পূর্ণসংখ্যা লিখুন: '))
৯
১০ # দ্বিতীয় পূর্ণসংখ্যা পড়ুন
১১ সংখ্যা২ = int(ইনপুট('দ্বিতীয় পূর্ণসংখ্যা লিখুন: '))
১২
১৩ যদি সংখ্যা ১ == সংখ্যা ২:
  ১৪     মুদ্রণ (সংখ্যা ১, 'সমান', সংখ্যা ২)
  ১৫
  ১৬ যদি সংখ্যা ১ != সংখ্যা ২:
  ১৭     মুদ্রণ (সংখ্যা ১, 'সমান নয়', সংখ্যা ২)
  ১৮
  ১৯ যদি সংখ্যা ১ < সংখ্যা ২:
  ২০     মুদ্রণ (সংখ্যা ১, 'এর চেয়ে কম', সংখ্যা ২)
  ২১
  ২২ যদি সংখ্যা ১ > সংখ্যা ২:
  ২৩     মুদ্রণ (সংখ্যা ১, 'এর চেয়ে বড়', সংখ্যা ২)
  ২৪
  ২৫ যদি সংখ্যা১ <= সংখ্যা২:
  ২৬     মুদ্রণ (সংখ্যা ১, 'এর চেয়ে কম বা সমান', সংখ্যা ২)
  ২৭
  ২৮ যদি সংখ্যা ১ >= সংখ্যা ২:
  ২৯     মুদ্রণ (সংখ্যা ১, 'এর চেয়ে বড় বা সমান', সংখ্যা ২)

```

কোড ইমেজ দেখতে এখানে ক্লিক করুন

দুটি পূর্ণসংখ্যা লিখুন এবং আমি আপনাকে তাদের মিলিত সম্পর্কগুলি বলব।
 প্রথম পূর্ণসংখ্যা লিখুন: 37
 দ্বিতীয় পূর্ণসংখ্যা লিখুন: 42
 ৩৭ ৪২ এর সমান নয়।
 ৩৭ হল ৪২ এর কম
 ৩৭ হল ৪২ এর কম বা সমান

কোড ইমেজ দেখতে এখানে ক্লিক করুন

দুটি পূর্ণসংখ্যা লিখুন এবং আমি আপনাকে তাদের মিলিত সম্পর্কগুলি বলব।

প্রথম পূর্ণসংখ্যা লিখুন: 7 দ্বিতীয় পূর্ণসংখ্যা লিখুন:

7

৭ সমান ৭

৭ হল ৭ এর কম বা সমান

৭ হল ৭ এর চেয়ে বড় বা সমান

কোড ইমেজ দেখতে এখানে ক্লিক করুন

দুটি পূর্ণসংখ্যা লিখুন এবং আমি আপনাকে তাদের মিলিত সম্পর্কগুলি বলব।

প্রথম পূর্ণসংখ্যা লিখুন: 54

দ্বিতীয় পূর্ণসংখ্যা লিখুন: ১৭

৫৪ সমান ১৭ নয়

৫৪ হল ১৭ এর চেয়ে বড়

৫৪ হল ১৭ এর চেয়ে বড় বা সমান

মন্তব্য

লাইন ১ হ্যাশ অক্ষর (#) দিয়ে শুরু হয়, যা নির্দেশ করে যে বাকি লাইনটি একটি

মন্তব্য:

চিত্র০২_০১.পিআই

সহজ রেফারেন্সের জন্য, আমরা প্রতিটি স্ক্রিপ্ট শুরু করি স্ক্রিপ্টের ফাইলের নাম নির্দেশ করে একটি মন্তব্য দিয়ে। একটি মন্তব্য একটি নির্দিষ্ট লাইনে কোডের ডানদিকে শুরু হতে পারে এবং চালিয়ে যেতে পারে সেই লাইনের শেষ পর্যন্ত।

ডকস্ট্রিংস

পাইথন কোডের স্টাইল গাইডে বলা হয়েছে যে প্রতিটি স্ক্রিপ্ট একটি ডকস্ট্রিং দিয়ে শুরু হওয়া উচিত যা স্ক্রিপ্টের উদ্দেশ্য ব্যাখ্যা করে, যেমন লাইন ২-এর একটি:

```
"""if স্টেটমেন্ট এবং তুলনা অপারেটর ব্যবহার করে পূর্ণসংখ্যার তুলনা করা।"""
```

জটিল স্ক্রিপ্টের ক্ষেত্রে, ডকস্ট্রিং প্রায়শই অনেক লাইন বিস্তৃত করে। পরবর্তী অধ্যায়গুলিতে, আপনি আপনার সংজ্ঞায়িত স্ক্রিপ্ট উপাদানগুলি বর্ণনা করার জন্য ডকস্ট্রিং ব্যবহার করবেন, যেমন নতুন ফাংশন এবং ক্লাস নামক নতুন প্রকার। আমরা IPython সহায়তা ব্যবস্থার সাহায্যে ডকস্ট্রিংগুলি কীভাবে অ্যাক্সেস করবেন তাও আলোচনা করব।

লাইন ৩ হল একটি ফাঁকা লাইন। কোড পড়া সহজ করার জন্য আপনি ফাঁকা লাইন এবং স্পেস অক্ষর ব্যবহার করেন। ফাঁকা লাইন, স্পেস অক্ষর এবং ট্যাব অক্ষর একসাথে **সাদা স্থান** হিসাবে পরিচিত। পাইথন বেশিরভাগ সাদা স্থান উপেক্ষা করে - আপনি দেখতে পাবেন যে কিছু ইন্ডেন্টেশন প্রয়োজন।

একটি লম্বা বিবৃতিকে লাইন জুড়ে বিভক্ত করা

লাইন ৪-৫

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
print('দুটি পূর্ণসংখ্যা লিখুন, আমি আপনাকে বলব',
      'তারা যে সম্পর্কগুলিকে সম্মত করে।')
```

ব্যবহারকারীকে নির্দেশাবলী প্রদর্শন করুন। এগুলি এক লাইনে ফিট করার জন্য খুব দীর্ঘ, তাই আমরা এগুলিকে দুটি স্ট্রিংয়ে ভেঙেছি। মনে রাখবেন যে আপনি কমা দ্বারা পৃথক তালিকা প্রিন্ট করার জন্য পাস করে বেশ কয়েকটি মান প্রদর্শন করতে পারেন - প্রিন্ট প্রতিটি মানকে একটি স্পেস দিয়ে পরবর্তী থেকে পৃথক করে।

সাধারণত, আপনি একটি লাইনে বিবৃতি লেখেন। আপনি \ continuation অক্ষর ব্যবহার করে একাধিক লাইনে একটি দীর্ঘ বিবৃতি ছড়িয়ে দিতে পারেন। Python আপনাকে continuation অক্ষর ব্যবহার না করেই (যেমন লাইন 4-5) বন্ধনীতে দীর্ঘ কোড লাইন বিভক্ত করার অনুমতি দেয়। Python কোডের স্টাইল গাইড অনুসারে দীর্ঘ কোড লাইন ভাঙার এটি পছন্দের উপায়। সর্বদা এমন ব্রেকিং পয়েন্ট নির্বাচন করুন যা অর্থপূর্ণ, যেমন প্রিন্ট করার পূর্ববর্তী কলে কমার পরে অথবা একটি দীর্ঘ এক্সপ্রেশনে একটি অপারেটরের আগে।

ব্যবহারকারীর কাছ থেকে পূর্ণসংখ্যার মান পড়া

এরপর, লাইন ৮ এবং ১১ ব্যবহারকারীর কাছ থেকে দুটি পূর্ণসংখ্যার মান জানতে এবং পড়তে বিল্টইন ইনপুট এবং int ফাংশন ব্যবহার করে।

যদি বিবৃতি

১৩-১৪ লাইনে if স্টেটমেন্ট

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
যদি সংখ্যা ১ == সংখ্যা ২:
    মুদ্রণ (সংখ্যা ১, 'সমান', সংখ্যা ২)
```

== তুলনা অপারেটর ব্যবহার করে নির্ধারণ করা হয় যে number1 এবং number2 ভেরিয়েবলের মান সমান কিনা।

যদি তাই হয়, তাহলে শর্টটি সত্য হবে এবং লাইন 14 একটি প্রদর্শন করে

লেখার if নির্দেশ করে যে মানগুলি সমান। যদি অবশিষ্ট if বিবৃতির শর্তগুলির মধ্যে কোনটি সত্য হয় (লাইন 16, 19, 22, 25 এবং 28), সংশ্লিষ্ট মুদ্রণটি একটি প্রদর্শন করে লেখার লাইন।

প্রতিটি if স্টেটমেন্টে if কীওয়ার্ড, পরীক্ষা করার শর্ত এবং একটি কোলন (:) থাকে যার পরে একটি ইন্ডেন্টেড বডি থাকে যাকে স্যুট বলা হয়। প্রতিটি স্যুটে এক বা একাধিক স্টেটমেন্ট থাকতে হবে। শর্তের পরে কোলন (:) ভুলে যাওয়া একটি সাধারণ সিনট্যাক্স ত্রুটি।

স্যুট ইন্ডেন্টেশন

পাইথনের জন্য আপনাকে স্যুটগুলিতে স্টেটমেন্টগুলি ইন্ডেন্ট করতে হবে। পাইথন কোডের স্টাইল গাইড ফোরম্পেস ইন্ডেন্টের সুপারিশ করে—আমরা এই বই জুড়ে সেই নিয়মটি ব্যবহার করি।
পরবর্তী অধ্যায়ে আপনি দেখতে পাবেন যে ভুল ইন্ডেন্টেশন ত্রুটির কারণ হতে পারে।

বিভিন্নিকর == এবং =

if স্টেটমেন্টের ক্ষিণে ইকুয়ালিটি অপারেটর (==) এর পরিবর্তে অ্যাসাইনমেন্ট প্রতীক (=) ব্যবহার করা একটি সাধারণ সিনট্যাক্স ত্রুটি। এটি এড়াতে, == কে "is equal to" হিসাবে এবং = কে "is assigned" হিসাবে পড়ুন।
পরবর্তী অধ্যায়ে আপনি দেখতে পাবেন যে অ্যাসাইনমেন্ট স্টেটমেন্টে = এর পরিবর্তে == ব্যবহার করলে সূক্ষ্ম সমস্যা দেখা দিতে পারে।

চেইনিং তুলনা

একটি মান একটি পরিসরের মধ্যে আছে কিনা তা পরীক্ষা করার জন্য আপনি তুলনাগুলিকে শৃঙ্খলিত করতে পারেন। নিম্নলিখিত তুলনাটি নির্ধারণ করে যে 1 থেকে 5 পরিসরের মধ্যে আছে কিনা, সমেত:

[1] তে: x = 3

[2] তে: 1 <= x <= 5

আউট[2]: সত্য

[3] তে: x = 10

[4] তে: 1 <= x <= 5

আউট[4]: মিথ্যা

আমরা এখন পর্যন্ত যে অপারেটরগুলি উপস্থাপন করেছি তার নজির

এই অধ্যায়ে প্রবর্তিত অপারেটরগুলির নজির নীচে দেখানো হল:

অপারেটরদের গ্রাম্পিং টাইপ

	বাকি আছে	
()	ঠিক	বন্ধনী
**	ডান থেকে বামে	সূচকীকরণ
* // %	বাকি আছে ঠিক	গুণ, সত্য ভাগ, তল ভাগ, ভাগশেষ
+ -	বাকি আছে ঠিক	যোগ, বিয়োগ
> <= <	বাকি আছে	কম, কম বা সমান, এর চেয়ে বড়, এর চেয়ে বড় বা সমান
>=	ঠিক	
== !=	বাকি আছে ঠিক	সমান, সমান নয়

টেবিলটিতে অপারেটরদের `toptobottom` এর ক্রমত্বাসমান অগ্রাধিকার ক্রম অনুসারে তালিকাভুক্ত করা হয়েছে। একাধিক অপারেটর সম্বলিত রাশি লেখার সময়, নিশ্চিত করুন যে তারা আপনার প্রত্যাশিত ক্রম অনুসারে মূল্যায়ন করে, অপারেটর অগ্রাধিকার চার্টটি উল্লেখ করে।

<https://docs.python.org/3/reference/expressions.html#operatorprecedence>

২.৮ বস্তু এবং গতিশীল টাইপিং

৭ (একটি পূর্ণসংখ্যা), 8.1 (একটি ভাসমান বিল্ডু সংখ্যা) এবং 'dog'-এর মতো মানগুলি সবই বস্তু।
প্রতিটি বস্তুর একটি ধরণ এবং একটি মান থাকে:

[1] তে: টাইপ(7)

আউট[1]: int

[2] তে: টাইপ(4.1)

[3] তে: টাইপ('কুকুর')

আউট[3]: str

একটি বস্তুর মান হলো বস্তুতে সংরক্ষিত তথ্য। উপরের স্লিপেটগুলি বিল্টইন ধরণের **int** (পূর্ণসংখ্যার জন্য), **float** (ফ্লোটিংপয়েন্ট সংখ্যার জন্য) এবং **str** (স্ট্রিং এর জন্য) অবজেক্ট দেখায়।

ডেরিয়েবল বলতে বস্তু বোঝায়

একটি ডেরিয়েবলের সাথে একটি বস্তু নির্ধারণ করলে সেই ডেরিয়েবলের নামটি বস্তুর সাথে **আবদ্ধ** (সম্পর্কিত) হয়।

যেমনটি আপনি দেখেছেন, আপনি আপনার কোডের ডেরিয়েবলটি ব্যবহার করে অবজেক্টের মান অ্যাক্সেস করতে পারেন:

[4] তে: $x = 7$ [5] তে: $x + 10$

আউট[5]: ১৭

[6] তে: x

আউট[6]: ৭

স্লিপেট [4] এর অ্যাসাইনমেন্টের পরে, চলক x 7 ধারণকারী পূর্ণসংখ্যার বস্তুকে **নির্দেশ** করে। স্লিপেট [6] তে দেখানো হয়েছে, স্লিপেট [5] x এর মান পরিবর্তন করে না। আপনি x পরিবর্তন করতে পারেন
নিম্নরূপ:

[7] তে: $x = x + 10$ [8] তে: x

আউট[8]: ১৭

গতিশীল টাইপিং

পাইথন **ডায়নামিক টাইপিং** ব্যবহার করে— কোড এক্সিকিউট করার সময় এটি একটি ভ্যারিয়েবলের উল্লেখ করা অবজেক্টের ধরণ নির্ধারণ করে। আমরা ভ্যারিয়েবল x কে বিভিন্ন অবজেক্টের সাথে রিবাইন্ড করে এবং তাদের প্রকারগুলি পরীক্ষা করে এটি দেখাতে পারি:

[9] তে: টাইপ(x)

আউট[9]: int

[10] তে: $x = 4.1$ [11] তে: টাইপ(x)

আউট[11]: ভাসমান

[12] তে: $x = \text{কুকুর}$ [13] তে: টাইপ(x)

আউট[13]: str

আবর্জনা সংগ্রহ

পাইথন মেমোরিতে অবজেক্ট তৈরি করে এবং প্রয়োজনে মেমোরি থেকে সেগুলো মুছে ফেলে। স্লিপেট [10] এর পর, ভেরিয়েবল x এখন একটি ফ্লোট অবজেক্টকে বোঝায়। স্লিপেট [7] এর পূর্ণসংখ্যা অবজেক্ট আর কোনও ভেরিয়েবলের সাথে আবদ্ধ থাকে না। আমরা পরবর্তী অধ্যায়ে আলোচনা করব, পাইথন স্বয়ংক্রিয়ভাবে মেমোরি থেকে এই ধরনের অবজেক্ট সরিয়ে দেয়। এই প্রক্রিয়া - যাকে আবর্জনা সংগ্রহ বলা হয় - আপনার তৈরি করা নতুন অবজেক্টের জন্য মেমোরি উপলব্ধ কিনা তা নিশ্চিত করতে **সাহায্য** করে।

।

২.৯ তথ্য বিজ্ঞানের ভূমিকা: মৌলিক বর্ণনামূলক পরিসংখ্যান

ডেটা সায়েন্সে, আপনি প্রায়শই আপনার ডেটা বর্ণনা এবং সারসংক্ষেপ করার জন্য পরিসংখ্যান ব্যবহার করবেন। এখানে, আমরা এই ধরনের বেশ কিছু **বর্ণনামূলক পরিসংখ্যানের** সাথে পরিচয় করিয়ে শুরু করব, যার মধ্যে রয়েছে:

- **সর্বনিম্ন**— মান সংগ্রহের মধ্যে সবচেয়ে ছোট মান।
- **সর্বোচ্চ**— মান সংগ্রহের মধ্যে বৃহত্তম মান।
- **পরিসর**— সর্বনিম্ন থেকে সর্বোচ্চ পর্যন্ত মানের পরিসর।
- **গণনা**— একটি সংগ্রহের মানের সংখ্যা।
- **যোগফল**—একটি সংগ্রহের মোট মান।

আমরা পরবর্তী অধ্যায়ে গণনা এবং যোগফল নির্ধারণের বিষয়টি দেখব। **বিচ্ছুরণের পরিমাপ** (যাকে পরিবর্তনশীলতার পরিমাপও বলা হয়), যেমন পরিসর, স্প্রেড আউট মানগুলি কতটা তা নির্ধারণ করতে সাহায্য করে। পরবর্তী অধ্যায়গুলিতে আমরা যে বিচ্ছুরণের অন্যান্য পরিমাপ উপস্থাপন করব তার মধ্যে রয়েছে প্রকরণ এবং মানক বিচুঃতি।

সর্বনিম্ন তিনটি মান নির্ধারণ করা

প্রথমে, দেখাবো কিভাবে ন্যূনতম তিনটি মান ম্যানুয়ালি নির্ধারণ করতে হয়। নিচের স্ক্রিপ্টটি তিনটি মান জিজ্ঞাসা করে এবং ইনপুট করে, if স্টেটমেন্ট ব্যবহার করে সর্বনিম্ন মান নির্ধারণ করে, তারপর এটি প্রদর্শন করে।

```

১ # fig02_02.py ২ """সর্বনিম্ন তিনটি
মান খুঁজুন!"""
৩
৪ নম্বর১ = int(input(' প্রথম পূর্ণসংখ্যা লিখুন: ')) ৫ নম্বর২ = int(input(' দ্বিতীয় পূর্ণসংখ্যা লিখুন: ')) ৬ নম্বর৩
= int(input(' তৃতীয় পূর্ণসংখ্যা লিখুন: '))
৭
৮ সর্বনিম্ন ৮ = সংখ্যা ১
৯
১০ যদি সংখ্যা ২ <ন্যূনতম:
১১           সর্বনিম্ন = সংখ্যা ২
১২
১৩ যদি সংখ্যা ৩ < কম:
১৪           সর্বনিম্ন = সংখ্যা ৩
১৫
১৬ প্রিন্ট ('সর্বনিম্ন মান হল', সর্বনিম্ন)

```

কোড ইমেজ দেখতে এখানে ক্লিক করুন

প্রথম পূর্ণসংখ্যা লিখুন: ১২
 দ্বিতীয় পূর্ণসংখ্যা লিখুন: ২৭
 তৃতীয় পূর্ণসংখ্যা লিখুন: ৩৬
 সর্বনিম্ন মান ১২

কোড ইমেজ দেখতে এখানে ক্লিক করুন

প্রথম পূর্ণসংখ্যা লিখুন: ২৭
 দ্বিতীয় পূর্ণসংখ্যা লিখুন: ১২
 তৃতীয় পূর্ণসংখ্যা লিখুন: ৩৬
 সর্বনিম্ন মান ১২

কোড ইমেজ দেখতে এখানে ক্লিক করুন

প্রথম পূর্ণসংখ্যা লিখুন: ৩৬
 দ্বিতীয় পূর্ণসংখ্যা লিখুন: ২৭
 তৃতীয় পূর্ণসংখ্যা লিখুন: ১২
 সর্বনিম্ন মান ১২

তিনটি মান ইনপুট করার পর, আমরা একবারে একটি মান প্রক্রিয়া করি:

- প্রথমত, আমরা ধরে নিচ্ছি যে সংখ্যা ১-এ সবচেয়ে ছোট মান রয়েছে, তাই লাইন ৪ এটিকে সর্বনিম্ন চলকের সাথে সংযুক্ত করে। অবশ্যই, এটা সম্ভব যে সংখ্যা ২ বা সংখ্যা ৩-এ

প্রকৃত ক্ষুদ্রতম মান, তাই আমাদের এখনও এই প্রতিটির সাথে ন্যূনতম ভুলনা করতে হবে।

- প্রথম if স্টেটমেন্ট (লাইন ১০-১১) তারপর number2 পরীক্ষা করে $< \text{min} <$ and if this
শর্ত সত্য, ন্যূনতম সংখ্যা 2 নির্ধারণ করে।
- দ্বিতীয় if বিবৃতি (লাইন ১৩-১৪) তারপর number3 $< \text{min}$ পরীক্ষা করে, এবং যদি এটি
শর্ত সত্য, সর্বনিম্ন সংখ্যা 3 নির্ধারণ করে।

এখন, minimum-এ সবচেয়ে ছোট মান থাকে, তাই আমরা এটি প্রদর্শন করি। আমরা স্লিপটি তিনবার কার্যকর করেছি যাতে দেখানো যায় যে ব্যবহারকারী প্রথমে, দ্বিতীয় বা তৃতীয় যেভাবেই প্রবেশ করুক না কেন, এটি সর্বদা সবচেয়ে ছোট মান খুঁজে পায়।

অন্তর্নির্মিত ফাংশন ব্যবহার করে সর্বনিম্ন এবং সর্বোচ্চ নির্ধারণ করা ন্যূনতম এবং সর্বোচ্চ

পাইথনে সাধারণ কাজ সম্পাদনের জন্য অনেকগুলি বিল্টইন ফাংশন রয়েছে। বিল্টইন ফাংশন **min** এবং **max** যথাক্রমে একটি সংগ্রহের সর্বনিম্ন এবং সর্বোচ্চ গণনা করে

মান:

[1] তে: ন্যূনতম(36, 27, 12)

আউট[1]: ১২

[2] তে: সর্বোচ্চ (36, 27, 12)

আউট[2]: 36

min এবং **max** ফাংশন দুটি যেকোনো সংখ্যক আর্গুমেন্ট গ্রহণ করতে পারে।

মূল্যের সংগ্রহের পরিসর নির্ধারণ করা

মানগুলির পরিসর হল সর্বনিম্ন থেকে সর্বোচ্চ মানের মধ্যে। এই ক্ষেত্রে, পরিসর হল ১২ থেকে ৩৬। ডেটা সায়েন্সের অনেক কিছুই আপনার ডেটা জানার জন্য নিবেদিত।

বর্ণনামূলক পরিসংখ্যান এর একটি গুরুত্বপূর্ণ অংশ, তবে আপনাকে পরিসংখ্যান কীভাবে ব্যাখ্যা করতে হয় তাও বুঝতে হবে। উদাহরণস্বরূপ, যদি আপনার 12 থেকে 36 পর্যন্ত পরিসরের 100টি সংখ্যা থাকে, তাহলে সেই সংখ্যাগুলি সেই পরিসরের উপর সমানভাবে বিতরণ করা যেতে পারে। বিপরীত প্রান্তে, আপনার 12 এর 99 মান এবং 36 এর একটি, অথবা 36 এর একটি 12 এবং 99 মানের সাথে ক্লাসিপিং থাকতে পারে।

কার্যকরী-শৈলী প্রোগ্রামিং: ত্রাস

এই বইয়ের মাধ্যমে, আমরা বিভিন্ন কার্যকরী-শৈলীর প্রোগ্রামিং ক্ষমতার সাথে পরিচয় করিয়ে দিচ্ছি। এগুলি আপনাকে এমন কোড লিখতে সক্ষম করে যা আরও সংক্ষিপ্ত, স্পষ্ট এবং ডিবাগ করা সহজ - অর্থাৎ ত্রুটিগুলি খুঁজে বের করা এবং সংশোধন করা। ন্যূনতম এবং সর্বোচ্চ ফাংশন হল এর উদাহরণ

ফাংশনাল স্টাইল প্রোগ্রামিং ধারণাকে রিডাকশন বলা হয়। তারা মানগুলির একটি সংগ্রহকে একটি একক মানে হ্রাস করে। আপনি অন্যান্য হ্রাস দেখতে পাবেন যার মধ্যে রয়েছে যোগফল, গড়, ভ্যারিয়েন্স এবং মানগুলির একটি সংগ্রহের মানক বিচুতি। আপনি কাস্টম কীভাবে সংজ্ঞায়িত করবেন তাও দেখতে পাবেন হ্রাস।

আসন্ন ডেটা সায়েন্স বিভাগের ভূমিকা

পরবর্তী দুটি অধ্যায়ে, আমরা কেন্দ্রীয় প্রবণতার পরিমাপ সহ মৌলিক বর্ণনামূলক পরিসংখ্যান সম্পর্কে আমাদের আলোচনা চালিয়ে যাব, যার মধ্যে রয়েছে গড়, মধ্যমা এবং মোড়, এবং বিচ্ছুরণের পরিমাপ, যার মধ্যে রয়েছে প্রকরণ এবং মানক বিচুতি।

২.১০ র্যাপ-আপ

এই অধ্যায়ে আমরা পাটিগণিত সম্পর্কে আলোচনা চালিয়ে গেছি। আপনি পরবর্তী ব্যবহারের জন্য মান সংরক্ষণের জন্য ভেরিয়েবল ব্যবহার করেছেন। আমরা পাইথনের পাটিগণিত অপারেটরদের সাথে পরিচয় করিয়ে দিয়েছি এবং দেখিয়েছি যে আপনাকে অবশ্যই সমস্ত এক্সপ্রেশন সরলরেখায় লিখতে হবে। আপনি ডেটা প্রদর্শনের জন্য বিল্টইন ফাংশন print ব্যবহার করেছেন। আমরা একক, দ্বিগুণ এবং ত্রিগুণ উদ্ভৃত স্ট্রিং তৈরি করেছি। আপনি বহু-রেখা স্ট্রিং তৈরি করতে এবং স্ট্রিংগুলিতে একক বা দ্বিগুণ উদ্ভৃত এন্ডেড করতে ত্রিগুণ উদ্ভৃত স্ট্রিং ব্যবহার করেছেন।

আপনি কীবোর্ডে ব্যবহারকারীর কাছ থেকে ইনপুট পেতে এবং প্রস্পট করার জন্য ইনপুট ফাংশন ব্যবহার করেছেন। আমরা স্ট্রিংগুলিকে সংখ্যাসূচক মানে রূপান্তর করার জন্য int এবং float ফাংশন ব্যবহার করেছি।

আমরা পাইথনের তুলনামূলক অপারেটরগুলি উপস্থাপন করেছি। তারপর, আপনি সেগুলিকে এমন একটি স্ক্রিপ্টে ব্যবহার করেছেন যা ব্যবহারকারীর দুটি পূর্ণসংখ্যা পড়ে এবং if বিবৃতির একটি সিরিজ ব্যবহার করে তাদের মানগুলির তুলনা করেছে।

আমরা পাইথনের গতিশীল টাইপিং নিয়ে আলোচনা করেছি এবং একটি বস্তুর ধরণ প্রদর্শনের জন্য বিল্টইন ফাংশন টাইপ ব্যবহার করেছি।

অবশেষে, আমরা মৌলিক বর্ণনামূলক পরিসংখ্যান সর্বনিম্ন এবং সর্বোচ্চ পরিচয় করিয়ে দিয়েছি এবং মান সংগ্রহের পরিসর গণনা করার জন্য সেগুলি ব্যবহার করেছি। পরবর্তী অধ্যায়ে, আমরা পাইথনের নিয়ন্ত্রণ বিবৃতি উপস্থাপন করব।

অ্যালিস্ট

গল্প. নিয়ন্ত্রণ বিবৃতি

উদ্দেশ্যমূলক ছবি

এই অধ্যায়ে, আপনি পাবেন: আর্নিং
পাথ

■ if, if else এবং if elif else দিয়ে সিদ্ধান্ত মিন। ffers & Deals

■ while এবং for ব্যবহার করে বারবার বিবৃতি কার্যকর করুন। ighlights

■ বর্ধিত অ্যাসাইনমেন্টের সাহায্যে অ্যাসাইনমেন্টের এক্সপ্রেশন ছোট করুন।

জিমিসপত্র

■ a এর জন্য ক্রিয়া পুনরাবৃত্তি করতে for স্টেটমেন্ট এবং বিল্টইন রেঞ্জ ফাংশন ব্যবহার করুন
সমর্থনান্তরণ গুলির ক্রম।

সাইন আউট করুন while দিয়ে সেন্টিনেল-নিয়ন্ত্রিত পুনরাবৃত্তি সম্পাদন করুন।

■ বুলিয়ান অপারেটর এবং, অথবা এবং না দিয়ে যৌগিক অবস্থা তৈরি করুন।

■ ব্রেক দিয়ে লুপিং বন্ধ করুন।

■ একটি লুপের পরবর্তী পুনরাবৃত্তিকে "continue" দিয়ে জোর করুন।

■ আরও সংক্ষিপ্ত স্ক্রিপ্ট লেখার জন্য ফাংশনাল স্টাইল প্রোগ্রামিং বৈশিষ্ট্যগুলি ব্যবহার করুন,
আরও স্পষ্ট, ডিবাগ করা সহজ এবং সমান্তরাল করা সহজ।

রূপরেখা

.5 while বিবৃতি

বিবৃতির জন্য .6

.6.1 পুনরাবৃত্ত, তালিকা এবং পুনরাবৃত্তকারী

.6.2 বিল্টইন রেঞ্জ ফাংশন

.9 বর্ধিত অ্যাসাইনমেন্ট

.8 সিকোয়েলনিয়ন্ত্রিত পুনরাবৃত্তি; ফর্ম্যাটেড স্ট্রিং

.9 সেন্টিনেলনিয়ন্ত্রিত পুনরাবৃত্তি

.10 বিল্টইন ফাংশন রেঞ্জ: আরও গভীরভাবে দেখুন

.11 আর্থিক পরিমাণের জন্য দশমিক টাইপ ব্যবহার করা

.12 বিরতি দিন এবং চালিয়ে যান বিবৃতি

.13 বুলিয়ান অপারেটর এবং, অথবা এবং না

.14 তথ্য বিজ্ঞানের ভূমিকা: কেন্দ্রীয় প্রবণতার পরিমাপ—গড়, মধ্যমা এবং মোড

.15 সারসংক্ষেপ

৩.১ ভূমিকা

এই অধ্যায়ে, আমরা পাইথনের নিয়ন্ত্রণ বিবৃতি উপস্থাপন করব - if, if else, if else, while, for, break এবং continue। আপনি for স্টেটমেন্টটি সিকোয়েল-নিয়ন্ত্রিত পুনরাবৃত্তি সম্পাদন করতে ব্যবহার করবেন - আপনি দেখতে পাবেন যে আইটেমের একটি ক্রম অনুসারে আইটেমের সংখ্যা for স্টেটমেন্টের পুনরাবৃত্তির সংখ্যা নির্ধারণ করে। আপনি বিল্টইন ফাংশনটি ব্যবহার করবেন।

পূর্ণসংখ্যার ক্রম তৈরি করার জন্য পরিসর।

আমরা while স্টেটমেন্টের সাহায্যে sentinelcontrolled পুনরাবৃত্তি দেখাবো। তুমি ব্যবহার করবে

সুনির্দিষ্ট আর্থিক গণনার জন্য পাইথন স্ট্যান্ডার্ড লাইব্রেরির দশমিক ধরণ। আমরা বিভিন্ন ফর্ম্যাট স্পেসিফায়ার ব্যবহার করে fstrings (অর্থাৎ, ফর্ম্যাট স্ট্রিং) তে ডেটা ফর্ম্যাট করব। আমরা বুলিয়ান অপারেটরগুলি এবং, অথবা এবং নয়, ঘোষিক অবস্থা তৈরির জন্যও দেখাব। ডেটা সায়েন্সের ভূমিকা বিভাগে, আমরা পাইথন স্ট্যান্ডার্ড লাইব্রেরির পরিসংখ্যান মডিউল ব্যবহার করে কেন্দ্রীয় প্রবণতার পরিমাপ - গড়, মধ্যমা এবং মোড - বিবেচনা করব।

.২ নিয়ন্ত্রণ বিবৃতি

পাইথন তিনটি নির্বাচন বিবৃতি প্রদান করে যা একটি শর্তের উপর ভিত্তি করে কোড কার্যকর করে যা সত্য বা মিথ্যা হিসাবে মূল্যায়ন করে:

- যদি শর্ত সত্য হয় তাহলে if স্টেটমেন্টটি একটি ক্রিয়া সম্পাদন করে অথবা যদি ক্রিয়াটি এভিয়ে ঘায় শর্টটি মিথ্যা।
- যদি কোন শর্ত সত্য হয় তাহলে if ... else স্টেটমেন্ট একটি ক্রিয়া সম্পাদন করে অথবা যদি কোন শর্ত মিথ্যা হয় তাহলে ডিফল্ট ক্রিয়া সম্পাদন করে।
- if ... elif... else বিবৃতিটি বিভিন্ন অবস্থার সত্যতা বা মিথ্যাতার উপর নির্ভর করে অনেকগুলি ডিফল্ট কর্মের মধ্যে একটি সম্পাদন করে।

যেখানেই একটি একক ক্রিয়া স্থাপন করা যেতে পারে, সেখানে একদল ক্রিয়া স্থাপন করা যেতে পারে।

পাইথন দুটি পুনরাবৃত্তি বিবৃতি প্রদান করে - while এবং for:

- যতক্ষণ পর্যন্ত শর্ত সত্য থাকে, ততক্ষণ পর্যন্ত while স্টেটমেন্টটি একটি ক্রিয়া (অথবা কর্মের একটি গ্রুপ) পুনরাবৃত্তি করে।
- for স্টেটমেন্টটি প্রতিটি আইটেমের জন্য আইটেমের ক্রমানুসারে একটি ক্রিয়া (অথবা কর্মের একটি গ্রুপ) পুনরাবৃত্তি করে।

কীওয়ার্ড

if, elif, else, while, for, True এবং False শব্দগুলো হল Python কীওয়ার্ড। একটি কীওয়ার্ডকে শনাক্তকারী হিসেবে ব্যবহার করা যেমন একটি ডেরিয়েবলের নাম একটি সিনট্যাক্স ক্ষেত্র। নিম্নলিখিত টেবিলে Python এর কীওয়ার্ড তালিকাভুক্ত করা হয়েছে।

পাইথন কীওয়ার্ড			
এবং	যেমন	জোর দিয়ে বলা	অ্যাসিস্ট অপেক্ষা
বিরতি ক্লাস চালিয়ে যান def			এর
www.EBooksWorld.ir			

এলিফ অন্যথায়

অবশেষে মিথ্যা

জন্য বিশ্বব্যাপী থেকে

যদি আমদানি

ভিতরে হল ল্যাষ্বড়া স্থানীয় নয়

না অথবা পাস রিটার্ন বুদ্ধি

সত্যিকারের চেষ্টা যখন ফলন সহ

৩.৩ যদি বিবৃতি

চলুন একটি Python if স্টেটমেন্ট এক্সিকিউট করি:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

[1] তে: গ্রেড = 85

[2] তে: যদি গ্রেড >= 60:

...: মুদ্রণ ('পাস করা হয়েছে')

...:

উত্তীর্ণ

শর্ত গ্রেড >= 60 সত্য, তাই if'-এ ইন্ডেন্টেড প্রিন্ট স্টেটমেন্ট

সৃষ্টি 'পাসড' দেখানো হয়েছে।

[সৃষ্টি ইন্ডেন্টেশন](#)

একটি সৃষ্টি ইন্ডেন্ট করা প্রয়োজন; অন্যথায়, একটি IndentationError সিনট্যাক্স ত্রুটি ঘটে:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

[3] তে: যদি গ্রেড >= 60 হয়:

...: print('Passed') #টি স্টেটমেন্ট সঠিকভাবে ইন্ডেন্ট করা হয়নি

ফাইল "<ipythoninpu3f42783904220>", লাইন 2

print('Passed') #টি স্টেটমেন্ট সঠিকভাবে ইন্ডেন্ট করা হয়নি

ইন্ডেন্টেশন ত্রুটি: একটি ইন্ডেন্টেড লাক প্রত্যাশিত

একটি স্যুটে একাধিক স্টেটমেন্ট থাকলে একটি IndentationErrorও ঘটে।

এবং এই বিবৃতিগুলির একই ইন্ডেন্টেশন নেই:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: যদি গ্রেড >= 60:

```
...:     print('পাস করা হয়েছে') # ৪টি স্পেস ইন্ডেন্ট করা হয়েছে
...: print('ভালো কাজ!') # মাত্র দুটি স্পেস ভুলভাবে ইন্ডেন্ট করা হয়েছে
ফাইল <ipythoninpu48c0d75c127bf>, লাইন 3
    print('ভালো কাজ!') # মাত্র দুটি স্পেস ভুলভাবে ইন্ডেন্ট করা হয়েছে
    ^
```

ইন্ডেন্টেশন ত্রুটি: আনিন্ডেন্ট কোনও বাইরের ইন্ডেন্টেশন স্তরের সাথে মেলে না

কখনও কখনও ত্রুটির বার্তা স্পষ্ট নাও হতে পারে। পাইথন লাইনের দিকে মনোযোগ আকর্ষণ করে, এই বিষয়টিই সাধারণত আপনার সমস্যাটি বুঝতে যথেষ্ট। আপনার কোড জুড়ে সমানভাবে ইন্ডেন্টেশন নিয়মাবলী প্রয়োগ করুন—যে প্রোগ্রামগুলি সমানভাবে নয়।
ইন্ডেন্ট করা লেখাগুলো পড়া কঠিন।

প্রতিটি অভিব্যক্তি সত্য বা মিথ্যা হিসাবে ব্যাখ্যা করা যেতে পারে

আপনি যেকোনো রাশির উপর ভিত্তি করে সিদ্ধান্ত নিতে পারেন। একটি অ-শূন্য মান হল সত্য। শূন্য হল মিথ্যা:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[5] তে: যদি 1:

```
...:     print('শূন্য নয় এমন মান সত্য, তাই এটি মুদ্রণ করবে')
...:
শূন্যবিহীন মান সত্য, তাই এটি মুদ্রণ করবে
```

[6] তে: যদি 0:

```
...:     print('শূন্য মিথ্যা, তাই এটি মুদ্রণ করবে না')
```

[7] তে:

অক্ষর ধারণকারী স্ট্রিংগুলি সত্য এবং খালি স্ট্রিংগুলি ("", "" বা "") হল মিথ্যা।

বিভান্তিকর == এবং =

অ্যাসাইনমেন্ট স্টেটমেন্টে = এর পরিবর্তে == ইকুয়ালিটি অপারেটর ব্যবহার করলে সূক্ষ্ম সমস্যা দেখা দিতে পারে।

উদাহরণস্বরূপ, এই সেশনে, স্লিপেট [1] গ্রেড সংজ্ঞায়িত করেছে

অ্যাসাইনমেন্ট:

গ্রেড = 85

যদি আমরা ভুলবশত লিখে ফেলি:

গ্রেড == ৮৫

তাহলে grade অনির্ধারিত হবে এবং আমরা একটি NameError পাবো। যদি grade পূর্ববর্তী বিবৃতির আগে সংজ্ঞায়িত করা হত, তাহলে grade == 85 কেবল True বা False মূল্যায়ন করত, এবং কোনও অ্যাসাইনমেন্ট সম্পাদন করত না। এটি একটি লজিক ত্রুটি।

৩.৪ যদি অন্যথায় এবং যদি অন্যথায় বিবৃতি

কোন শর্ত সত্য কিনা তার উপর ভিত্তি করে if else স্টেটমেন্ট বিভিন্ন সূট এক্সিকিউট করে।

অথবা মিথ্যা:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: গ্রেড = 85

[2] তে: যদি গ্রেড >= 60:

```
...:           মুদ্রণ ('পাস করা হয়েছে')
...: অন্যথায়:
...:           মুদ্রণ ('ব্যর্থ')
...:
উত্তীর্ণ
```

উপরের শর্তটি True, তাই if সূটটি 'Passed' প্রদর্শন করে। মনে রাখবেন যে print('Passed') টাইপ করার পরে যখন আপনি Enter টিপবেন, তখন IPython পরবর্তী লাইনে চারটি স্পেস ইডেন্ট করবে।

আপনাকে অবশ্যই সেই চারটি স্পেস মুছে ফেলতে হবে যাতে else: suite সঠিকভাবে i এর অধীনে সারিবদ্ধ হয়। যদি.

নিচের কোডটি ভ্যারিয়েবল গ্রেডে 57 নির্ধারণ করে, তারপর আবার if else স্টেটমেন্টটি দেখায় যাতে দেখা যায় যে শুধুমাত্র else সূটটি তখনই কার্যকর হয় যখন শর্তটি মিথ্যা:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3] তে: গ্রেড = 57

[4] তে: যদি গ্রেড >= 60:

...: মুদ্রণ ('পাস করা হয়েছে')
: অন্যথায়:
 ...: মুদ্রণ ('ব্যর্থ')

 ব্যর্থ হয়েছে

বর্তমান ইন্টারেক্ষিভ সেশনের স্লিপেটগুলির মধ্যে পিছনে এবং সামনে নেভিগেট করার জন্য আপ এবং ডাউন তীর কীগুলি ব্যবহার করুন। এন্টার টিপলে প্রদর্শিত স্লিপেটটি পুনরায় কার্যকর হয়। আসুন গ্রেড 99 এ সেট করি, স্লিপেট [4] থেকে কোডটি প্রত্যাহার করতে আপ তীর কীটি দুবার টিপুন, তারপরে সেই কোডটি স্লিপেট [6] হিসাবে পুনরায় কার্যকর করতে এন্টার টিপুন। আপনার দ্বারা কার্যকর করা প্রতিটি প্রত্যাহার করা স্লিপেট একটি নতুন আইডি পায়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[5] তে: গ্রেড = 99

[6] তে: যদি গ্রেড >= 60:

...: মুদ্রণ ('পাস করা হয়েছে')
: অন্যথায়:
 ...: মুদ্রণ ('ব্যর্থ')

 উত্তীর্ণ

শর্তসাপেক্ষ অভিযন্ত্রি

কখনও কখনও if else স্টেটমেন্টের স্যুটগুলি একটি ভেরিয়েবলের জন্য বিভিন্ন মান নির্ধারণ করে, একটি শর্তের উপর ভিত্তি করে, যেমন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[7] তে: গ্রেড = 87

[8] তে: যদি গ্রেড >= 60:

...: ফলাফল = 'পাস'
: অন্যথায়:
 ...: ফলাফল = 'ব্যর্থ'

এরপর আমরা সেই ভেরিয়েবলটি মুদ্রণ বা মূল্যায়ন করতে পারি:

[9] তে: ফলাফল

আউট[9]: 'পাস'

আপনি একটি সংক্ষিপ্ত শর্তসাপেক্ষ অভিব্যক্তি ব্যবহার করে স্লিপেট [8] এর মতো বিবৃতি লিখতে পারেন :

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[10] তে: ফলাফল = ('পাস' যদি গ্রেড \geq 60 হয় অন্যথায় 'ব্যর্থ')

[11] তে: ফলাফল

আউট[11]: 'পাস'

বন্ধনীগুলি আবশ্যিক নয়, তবে তারা স্পষ্ট করে যে বিবৃতিটি শর্তসাপেক্ষ এক্সপ্রেশনের মান ফলাফলকে নির্ধারণ করে।
প্রথমে, পাইথন শর্ত গ্রেড মূল্যায়ন করে।

$\geq 60:$

- যদি এটি সত্য হয়, তাহলে স্লিপেট [10] ফলাফলের জন্য if এর বাম দিকের এক্সপ্রেশনের মান নির্ধারণ করে,
অর্থাৎ 'Passed'। else অংশটি কার্যকর হয় না।
- যদি এটি মিথ্যা হয়, তাহলে স্লিপেট [10] অন্যটির ডানদিকে এক্সপ্রেশনের মান নির্ধারণ করে, অর্থাৎ 'ব্যর্থ'।

ইন্টারেক্টিভ মোডে, আপনি শর্তসাপেক্ষ অভিব্যক্তিটি সরাসরি মূল্যায়ন করতে পারেন, যেমন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[12] তে: 'পাস' যদি গ্রেড \geq 60 হয় অন্যথায় 'ব্যর্থ'

আউট[12]: 'পাস'

একটি স্যুটে একাধিক বিবৃতি

নিচের কোডটি if... else এর else স্যুটে দুটি স্টেটমেন্ট দেখায়।

বিবৃতি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[13] তে: গ্রেড = 49

[14] তে: যদি গ্রেড \geq 60 হয়: print('পাস করা

...: হয়েছে')

...: অন্যথায়:

...: মুদ্রণ ('ব্যর্থ')

...: print('আপনাকে আবার এই কোর্সটি করতে হবে')

...:

ব্যর্থ হয়েছে

তোমাকে আবার এই কোর্সটি করতে হবে।

এই ক্ষেত্রে, গ্রেড 60 এর কম, তাই else's suite-এর উভয় স্টেটমেন্টই কার্যকর হবে।

যদি আপনি দ্বিতীয় প্রিন্টটি ইন্ডেন্ট না করেন, তাহলে এটি অন্যের স্যুটে নেই। সুতরাং, সেই বিবৃতিটি সর্বদা কার্যকর হয়, সম্ভবত অন্তুত ভুল আউটপুট তৈরি করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[15] তে: গ্রেড = 100

[16] তে: যদি গ্রেড >= 60 হয়:

```
...:           মুদ্রণ ('পাস করা হয়েছে')
...: অন্যথায়:
...:           মুদ্রণ ('ব্যর্থ')
...: print('আপনাকে আবার এই কোর্সটি করতে হবে')
...:
উত্তীর্ণ
```

তোমাকে আবার এই কোর্সটি করতে হবে।

যদি... elif... else বিবৃতি

আপনি if... elif... else স্টেটমেন্ট ব্যবহার করে অনেক ক্ষেত্রে পরীক্ষা করতে পারেন। নিম্নলিখিত কোডটি 90 এর চেয়ে বড় বা সমান গ্রেডের জন্য "A", 80-89 রেঞ্জের গ্রেডের জন্য "B", 70-79 গ্রেডের জন্য "C", 60-69 গ্রেডের জন্য "D" এবং অন্যান্য সমস্ত গ্রেডের জন্য "F" প্রদর্শন করে।

শুধুমাত্র প্রথম True শর্তের জন্য ক্রিয়াটি কার্যকর হয়। স্মিপেট [18] C প্রদর্শন করে, কারণ

গ্রেড হল ৭৭:

[17] তে: গ্রেড = 77

[18] তে: যদি গ্রেড >= 90 হয়:

```
...:           মুদ্রণ ('A')
...: এলিফ গ্রেড >= ৮০:
...:           মুদ্রণ ('B')
...: এলিফ গ্রেড >= ৭০:
...:           মুদ্রণ('C') ...: এলিফ
গ্রেড >= ৬০: মুদ্রণ('D')
...:
...: অন্যথায়:
...:           মুদ্রণ ('F')
...:
```

গ

প্রথম শর্ত—গ্রেড >= 90—ফলস, তাই print('A') বাদ দেওয়া হয়েছে। দ্বিতীয় শর্ত—গ্রেড >= 80—ও ফলস, তাই

print('B') বাদ দেওয়া হয়েছে। তৃতীয় শর্ত—গ্রেড >= 70—সত্য, তাই print('C') কার্যকর করা হয়েছে।

তারপর বাকি সব

if... elif... else স্টেটমেন্টের কোড বাদ দেওয়া হয়েছে। একটি if... elif... else পৃথক if স্টেটমেন্টের চেয়ে দ্রুতর,
কারণ কল্পনা টেস্টিং বন্ধ হয়ে যাওয়ার সাথে সাথেই
শর্তটি সত্য।

অন্যথা ঐচ্ছিক

if... elif... else স্টেটমেন্টে else ঐচ্ছিক। এটি অন্তর্ভুক্ত করলে আপনি এমন মান পরিচালনা করতে পারবেন যা কোনও
শর্ত পূরণ করে না। যখন else ছাড়া একটি if... elif স্টেটমেন্ট এমন একটি মান পরীক্ষা করে যা তার কোনও
শর্ত তৈরি করে না, তখন প্রোগ্রামটি স্টেটমেন্টের কোনও সুটি কার্যকর করে না - if... elif... স্টেটমেন্ট কার্যকর হওয়ার
পরের ক্রমানুসারে পরবর্তী স্টেটমেন্ট। যদি আপনি else নির্দিষ্ট করেন, তাহলে আপনাকে এটি শেষ elif এর পরে
রাখতে হবে; অন্যথায়, একটি SyntaxError দেখা দেয়।

যুক্তিগত ক্রটি

স্লিপেট [16]-এ ভুলভাবে ইন্ডেন্ট করা কোড সেগমেন্টটি একটি ননফ্যাটাল লজিক ক্রটির উদাহরণ। কোডটি কার্যকর
হয়, কিন্তু এটি ভুল ফলাফল দেয়। একটি স্লিপেট একটি মারাত্মক লজিক ক্রটির জন্য, একটি ব্যতিক্রম ঘটে (যেমন 0
দ্বারা ভাগ করার প্রচেষ্টা থেকে একটি ZeroDivisionError), তাই পাইথন একটি ট্রেসব্যাক প্রদর্শন করে, তারপর স্লিপেটটি
বন্ধ করে দেয়। ইন্টারেক্শিভ মোডে একটি মারাত্মক ক্রটি শুধুমাত্র বর্তমান স্লিপেটটি বন্ধ করে দেয় - তারপর IPython
আপনার পরবর্তী ইনপুটের জন্য অপেক্ষা করে।

৩.৫ বিবৃতির সময়

while স্টেটমেন্ট আপনাকে একটি শর্ত True থাকাকালীন এক বা একাধিক ক্রিয়া পুনরাবৃত্তি করতে দেয়। আসুন
while স্টেটমেন্ট ব্যবহার করে 50 এর চেয়ে বড় 3 এর প্রথম ঘাত বের করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: গুণফল = 3
```

```
[2] তে: যখন পণ্য <= 50:
```

```
...:           পণ্য = পণ্য * 3
...:
```

```
[3] তে: পণ্য
```

```
আউট[3]: 81
```

স্লিপেট [3] পণ্যটির মান, 81 দেখতে মূল্যায়ন করে, যা 3 বৃহত্তরের প্রথম ঘাত
৫০ এর বেশি।

while স্টেটমেন্টের সুটি কিছু একটা অবশ্যই পণ্যের মান পরিবর্তন করবে, তাই

শর্টটি অবশেষে False হয়ে যায়। অন্যথায়, একটি অসীম লুপ ঘটে। টার্মিনাল, অ্যানাকোডা কমান্ড প্রম্পট বা শেল থেকে সম্পাদিত অ্যাপ্লিকেশনগুলিতে, অসীম লুপটি বন্ধ করতে Ctrl + c অথবা control + c টাইপ করুন। IDE গুলিতে সাধারণত একটি প্রোগ্রামের সম্পাদন বন্ধ করার জন্য একটি টুলবার বোতাম বা মেনু বিকল্প থাকে।

বিবৃতির জন্য ৩.৬

for স্টেটমেন্ট আপনাকে প্রতিটি আইটেমের জন্য একটি ক্রিয়া বা একাধিক ক্রিয়া পুনরাবৃত্তি করতে দেয় আইটেমের ক্রমানুসারে। উদাহরণস্বরূপ, একটি স্ট্রিং হল পৃথক অক্ষরের একটি ক্রম। আসুন 'প্রোগ্রামিং' প্রদর্শন করি যার অক্ষর দুটি স্পেস দ্বারা পৃথক করা হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1]-এ: 'প্রোগ্রামিং'-এ চরিত্রের জন্য :
...:           মুদ্রণ (অক্ষর, শেষ = ' ')
...:
প্রোগ্রামিং
```

for স্টেটমেন্টটি নিম্নরূপ কার্যকর হয়:

- স্টেটমেন্টটি প্রবেশ করানোর পর, এটি 'প্রোগ্রামিং'-এ 'P' কে for এবং in—এই ক্ষেত্রে, অক্ষরের মধ্যে টার্গেট ভেরিয়েবলে বরাদ্দ করে।
- এরপর, স্যুটের স্টেটমেন্টটি কার্যকর হবে, অক্ষরের মান প্রদর্শন করবে এবং তারপরে দুটি স্পেস থাকবে—আমরা এই বিষয়ে কিছুক্ষণের মধ্যে আরও বিস্তারিত বলব।
- স্যুটটি কার্যকর করার পর, পাইথন ক্রমের পরবর্তী আইটেমটি (অর্থাৎ, 'প্রোগ্রামিং'-এ 'r') অক্ষরে
- প্রক্রিয়াকরণের ক্রমানুসারে আরও আইটেম থাকা পর্যন্ত এটি চলতে থাকে। এই ক্ষেত্রে, 'g' অক্ষরটি প্রদর্শনের পরে বিবৃতিটি শেষ হয়, তারপরে দুটি স্পেস থাকে।

স্যুটে টার্গেট ভেরিয়েবল ব্যবহার করা, যেমনটি আমরা এখানে এর মান প্রদর্শনের জন্য করেছি, সাধারণ কিন্তু প্রয়োজনীয় নয়।

ফাংশন প্রিন্টের শেষ কীওয়ার্ড আর্গুমেন্ট

বিল্টইন ফাংশন print তার আর্গুমেন্ট(গুলি) প্রদর্শন করে, তারপর কার্সারটিকে পরবর্তী লাইনে নিয়ে যায়। আপনি আর্গুমেন্ট end দিয়ে এই আচরণটি পরিবর্তন করতে পারেন, যেমন

মুদ্রণ (অক্ষর, শেষ = ' ')

যা অক্ষরের মান এবং তার পরে দুটি স্পেস প্রদর্শন করে। সুতরাং, সমস্ত অক্ষর একই লাইনে অনুভূমিকভাবে প্রদর্শিত হয়। পাইথন একটি **কীওয়ার্ড আর্গুমেন্ট**কে end কল করে, কিন্তু end নিজেই একটি পাইথন কীওয়ার্ড নয়। কীওয়ার্ড আর্গুমেন্টগুলিকে কখনও কখনও **নামযুক্ত আর্গুমেন্ট** বলা হয়। end কীওয়ার্ড আর্গুমেন্ট এক্ষিক। যদি আপনি এটি অন্তর্ভুক্ত না করেন, তাহলে print ডিফল্টরূপে একটি নতুন লাইন ('\n') ব্যবহার করে। পাইথন কোডের জন্য স্টাইল গাইড একটি কীওয়ার্ড আর্গুমেন্টের = এর চারপাশে কোনও স্পেস না রাখার পরামর্শ দেয়।

ফাংশন প্রিন্টের সেপ কীওয়ার্ড আর্গুমেন্ট

প্রিন্টে প্রদর্শিত আইটেমগুলির মধ্যে প্রদর্শিত স্ট্রিংটি নির্দিষ্ট করতে আপনি কীওয়ার্ড আর্গুমেন্ট **sep** (বিভাজকের জন্য সংক্ষিপ্ত) ব্যবহার করতে পারেন। যখন আপনি এই আর্গুমেন্টটি নির্দিষ্ট করেন না, তখন print ডিফল্টরূপে একটি স্পেস অক্ষর ব্যবহার করে। আসুন তিনটি সংখ্যা প্রদর্শন করি, প্রতিটি সংখ্যাকে কেবল একটি স্পেসের পরিবর্তে একটি কমা এবং একটি স্পেস দ্বারা পৃথক করা হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[2] তে: print(10, 20, 30, sep=', ') 10, 20, 30

ডিফল্ট স্পেস মুছে ফেলার জন্য, sep="" (অর্থাৎ, একটি খালি স্ট্রিং) ব্যবহার করুন।

৩.৬.১ পুনরাবৃত্ত, তালিকা এবং পুনরাবৃত্তকারী

for স্টেটমেন্টের in কীওয়ার্ডের ডানদিকের ক্রমটি অবশ্যই একটি **পুনরাবৃত্তযোগ্য** হতে হবে— অর্থাৎ, এমন একটি বস্তু যেখান থেকে for স্টেটমেন্ট একবারে একটি আইটেম নিতে পারে যতক্ষণ না আর কোনও আইটেম অবশিষ্ট থাকে। পাইথনের স্ট্রিং ছাড়াও অন্যান্য পুনরাবৃত্তযোগ্য ক্রম প্রকার রয়েছে। সবচেয়ে সাধারণগুলির মধ্যে একটি হল একটি **তালিকা**, যা বর্গাকার বন্ধনীতে ([এবং]) আবদ্ধ আইটেমগুলির একটি কমা দ্বারা পৃথক সংগ্রহ। নিম্নলিখিত কোডটি একটি তালিকায় মোট পাঁচটি পূর্ণসংখ্যা দেয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3] তে: মোট = 0

[4] তে: [2, 3, 0, 17, 9] তে সংখ্যার জন্য :

...: মোট = মোট + সংখ্যা

...:

[5] তে: মোট

আউট[5]: 25

প্রতিটি সিকোয়েল্সের একটি ইটারেটর থাকে। for স্টেটমেন্ট "behind the scenes" ইটারেটর ব্যবহার করে প্রতিটি ধারাবাহিক আইটেম প্রক্রিয়া করার জন্য যতক্ষণ না আর কিছু থাকে। ইটারেটর একটি বুকমার্কের মতো—এটি সর্বদা জানে যে এটি সিকোয়েল্সের কোথায় আছে, তাই যখন এটি করার প্রয়োজন হয় তখন এটি পরবর্তী আইটেমটি ফেরত দিতে পারে। আমরা "সিকোয়েল্স: তালিকা এবং টুপলস" অধ্যায়ে তালিকাগুলি বিস্তারিতভাবে আলোচনা করেছি। সেখানে, আপনি দেখতে পাবেন যে একটি তালিকার আইটেমগুলির ক্রম গুরুত্বপূর্ণ এবং একটি তালিকার আইটেমগুলি পরিবর্তনযোগ্য (অর্থাৎ, পরিবর্তনযোগ্য)।

৩.৬.২ অন্তর্নির্মিত পরিসর ফাংশন

আসুন a for স্টেটমেন্ট এবং builtin range ফাংশন ব্যবহার করে 0 থেকে 9 পর্যন্ত মানগুলি সঠিকভাবে 10 বার পুনরাবৃত্তি করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6] তে: [রেঞ্জের কাউন্টারের জন্য](#) (10):

```
...:           মুদ্রণ (কাউন্টার, শেষ = ' ')
...:
0 1 2 3 4 5 6 7 8 9
```

ফাংশন কল রেঞ্জ(10) একটি পুনরাবৃত্ত বস্তু তৈরি করে যা 0 থেকে শুরু করে এবং আর্গমেন্ট মান (10) পর্যন্ত অব্যাহত ধারাবাহিক পূর্ণসংখ্যার একটি ক্রম উপস্থাপন করে - এই ক্ষেত্রে, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9। রেঞ্জ দ্বারা উৎপাদিত শেষ পূর্ণসংখ্যা প্রক্রিয়াকরণ শেষ করার পরে for স্টেটমেন্টটি প্রস্থান করে। Iterators এবং iterable বস্তু হল Python এর দুটি কার্যকরী স্টাইল প্রোগ্রামিং বৈশিষ্ট্য। আমরা বই জুড়ে এর আরও পরিচয় করিয়ে দেব।

একের পর এক ক্রটি

একটি সাধারণ ধরণের অফবাইওন ক্রটি ঘটে যখন আপনি ধরে নেন যে জেনারেটেড সিকোয়েল্সে রেঞ্জের আর্গমেন্ট মান অন্তর্ভুক্ত করা হয়েছে। উদাহরণস্বরূপ, যদি আপনি 0 থেকে 9 সিকোয়েল্স তৈরি করার চেষ্টা করার সময় রেঞ্জের আর্গমেন্ট হিসাবে 9 প্রদান করেন, তাহলে রেঞ্জ কেবল 0 থেকে 8 জেনারেট করে।

৩.৭ বর্ধিত নিয়োগ

অগমেন্টেড অ্যাসাইনমেন্ট হল অ্যাসাইনমেন্ট এক্সপ্রেশনকে সংক্ষেপে বলা হয় যেখানে অ্যাসাইনমেন্টের = এর বাম এবং ডানে একই ভেরিয়েবলের নাম প্রদর্শিত হয়, যেমনটি total এর ক্ষেত্রে দেখা যায়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

স্লিপেট [2] একটি **সংযোজন বর্ধিত অ্যাসাইনমেন্ট (+=)** ব্যবহার করে এটি পুনরায় বাস্তবায়ন করে।

বিবৃতি:

... কোড ইমেজ দেখতে এখানে ক্লিক করুন ...

[1] তে: মোট = 0

[2] তে: [1, 2, 3, 4, 5] তে সংখ্যার জন্য :

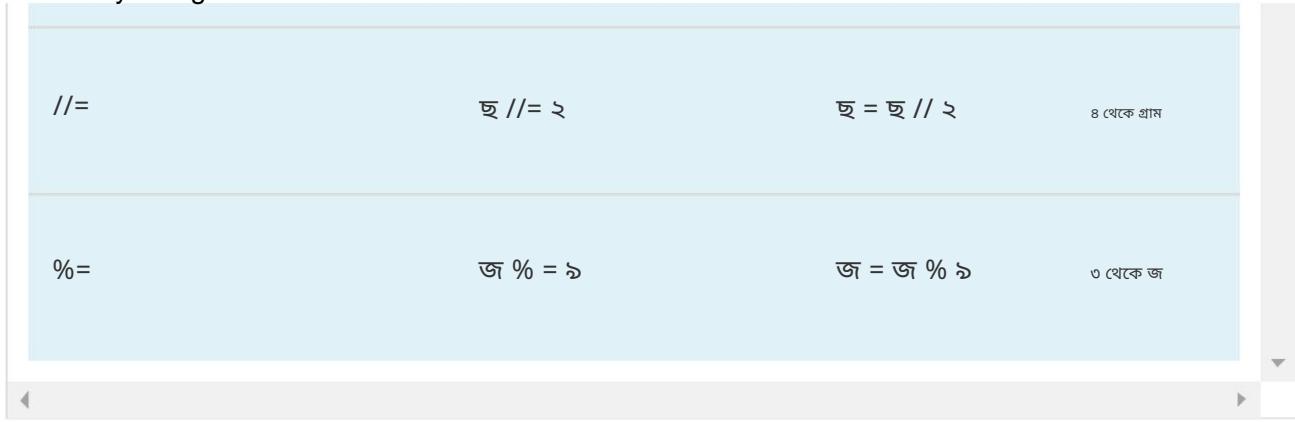
মোট += সংখ্যা # মোটের সাথে সংখ্যা যোগ করুন
....

[3] তে: মোট

আউট[3]: ১৫

স্লিপেট [2]-এ += রাশিটি প্রথমে বর্তমান মোট সংখ্যার সাথে সংখ্যার মান যোগ করে, তারপর নতুন মান মোট সংরক্ষণ করে। নীচের টেবিলটি নমুনা বর্ধিত অ্যাসাইনমেন্ট দেখায়:

বর্ধিত অ্যাসাইনমেন্ট	নমুনা	ব্যাখ্যা বরাদ্দ
$+=$	$g += q$	$g = g + q$ ১০ থেকে g
$=$	$g = 8$	$g = g$ ৮ ১ থেকে ডি
$*=$	এবং $*= c$	$g = g * c$ ২০ থেকে ই
$**=$	$c **= 3$	$g = g ** 3$ ৮ থেকে চ
$/=$	$g /= 2$	$g = g / 2$ ৮.৫ থেকে গ্রাম



৩.৮ ক্রম-নিয়ন্ত্রিত পুনরাবৃত্তি; ফর্ম্যাটেড স্ট্রিং

এই বিভাগ এবং পরবর্তী বিভাগ দুটি শ্রেণির গড় সমস্যার সমাধান করে। নিম্নলিখিত প্রয়োজনীয়তা বিবৃতিটি বিবেচনা করুন:

দশজন শিক্ষার্থীর একটি ক্লাস একটি কুইজে অংশগ্রহণ করেছিল। তাদের গ্রেড (০-১০০ পরিসরের পূর্ণসংখ্যা) হল ৯৮, ৭৬, ৭১, ৮৭, ৮৩, ৯০, ৫৭, ৭৯, ৮২, ৯৪। কুইজে ক্লাসের গড় নির্ণয় করো।

এই সমস্যা সমাধানের জন্য নিম্নলিখিত স্ট্রিপ্টটি গ্রেডের চলমান মোট সংখ্যা ধরে রাখে, গড় গণনা করে এবং ফলাফল প্রদর্শন করে। আমরা ১০টি গ্রেড একটি তালিকায় রেখেছি, তবে আপনি কীবোর্ডে ব্যবহারকারীর কাছ থেকে গ্রেড ইনপুট করতে পারেন (যেমনটি আমরা পরবর্তী উদাহরণে করেব) অথবা একটি ফাইল থেকে পড়তে পারেন (যেমনটি আপনি "ফাইল এবং ব্যতিক্রম" অধ্যায়ে কীভাবে করবেন তা দেখতে পাবেন)। আমরা অধ্যায় ১৬-তে SQL এবং NoSQL ডাটাবেস থেকে ডেটা কীভাবে পড়তে হয় তা দেখাই।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```

১ # class_average.py ২ """ক্রম-নিয়ন্ত্রিত
পুনরাবৃত্তি সহ ক্লাস গড় প্রোগ্রাম!"""
৩
৪ # প্রাথমিককরণ পর্ব ৫ মোট = ০ # গ্রেডের
যোগফল ৬ গ্রেড_কাউন্টার = ৭ গ্রেড = [৯৮, ৭৬, ৭১,
৮৭, ৮৩, ৯০, ৫৭, ৭৯, ৮২, ৯৪] # ৮
৯ গ্রেডের তালিকা
১০
১১ # প্রক্রিয়াকরণ পর্যায়
গ্রেডের জন্য ১১ :
১২     মোট += গ্রেড # চলমান মোটের সাথে বর্তমান গ্রেড যোগ করুন
১৩     grade_counter += ১ # নির্দেশ করে যে আরও একটি গ্রেড প্রক্রিয়া d ছিল
১৪ # সমষ্টি পর্ব ১৫ গড় = মোট / গ্রেড_কাউন্টার
১৬ মুদ্রণ (f'শ্রেণীর গড় {গড়}'")

```

কোড ইমেজ দেখতে এখানে ক্লিক করুন

ব্লাসের গড় ৮১.৭

৫-৬ নম্বর লাইনে total এবং grade_counter ভ্যারিয়েবল তৈরি করা হয়েছে এবং প্রতিটিকে ০ তে ইনিশিয়ালাইজ করা হয়েছে।
লাইন ৭

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
গ্রেড = [৯৮, ৭৬, ৭১, ৮৭, ৮৩, ৯০, ৫৭, ৭৯, ৮২, ৯৪] # ১০টি গ্রেডের তালিকা
```

ভেরিয়েবল গ্রেড তৈরি করে এবং ১০টি পূর্ণসংখ্যা গ্রেডের একটি তালিকা দিয়ে এটিকে আরভ্ন করে।

তালিকার প্রতিটি গ্রেডকে for স্টেটমেন্ট প্রক্রিয়া করে। লাইন ১১ মোট গ্রেডের সাথে বর্তমান গ্রেড যোগ করে। তারপর, লাইন ১২ এখন পর্যন্ত প্রক্রিয়াকৃত গ্রেডের সংখ্যা ট্র্যাক করার জন্য ভেরিয়েবল grade_counter-এ ১ যোগ করে। তালিকার সমষ্টি ১০টি গ্রেড প্রক্রিয়া করা হয়ে গেলে পুনরাবৃত্তি শেষ হয়। পাইথন কোডের স্টাইল গাইড প্রতিটি নিয়ন্ত্রণ বিবৃতির উপরে এবং নীচে একটি ফাঁকা লাইন রাখার পরামর্শ দেয় (যেমন লাইন ৮ এবং ১৩-তে)। যখন for বিবৃতিটি শেষ হয়, লাইন ১৫ গড় গণনা করে এবং লাইন ১৬ এটি প্রদর্শন করে। এই অধ্যায়ের পরে, আমরা তালিকার আইটেমগুলির গড় আরও সংক্ষিপ্তভাবে গণনা করার জন্য ফাংশনালস্টাইল প্রোগ্রামিং ব্যবহার করি।

ফরম্যাটেড স্ট্রিং এর ভূমিকা

১৬ নম্বর লাইনে একটি স্ট্রিংয়ে গড়ের মান সন্নিবেশ করে এই স্ক্রিপ্টের ফলাফল ফরম্যাট করার জন্য নিম্নলিখিত সহজ **fstring** (ফরম্যাটেড স্ট্রিং এর সংক্ষিপ্ত রূপ) ব্যবহার করা হয়েছে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

'শ্রেণীর গড় হল {গড়}'

স্ট্রিং এর শুরুর উদ্ধৃতি চিহ্নের আগে f অক্ষরটি নির্দেশ করে যে এটি একটি fstring। কোকড়া বন্ধনী ({ এবং }) দ্বারা বিভক্ত স্থানধারক ব্যবহার করে আপনি মানগুলি কোথায় সন্নিবেশ করতে হবে তা নির্দিষ্ট করেন। স্থানধারকটি

{গড়}

চলক গড়ের মানকে একটি স্ট্রিং উপস্থাপনায় রূপান্তর করে, তারপর প্রতিস্থাপন করে

{average} সেই প্রতিস্থাপন টেক্সট সহ। Replacementtext এক্সপ্রেশনগুলিতে মান, চলক বা অন্যান্য এক্সপ্রেশন থাকতে পারে, যেমন গণনা বা ফাংশন কল। লাইন ১৬-তে, আমরা গড় এর পরিবর্তে total / grade_counter ব্যবহার করতে পারতাম, লাইন ১৫-এর প্রয়োজনীয়তা বাদ দিয়ে।

৩.৯ সেন্টিনেল-নিয়ন্ত্রিত আবর্তন

ক্লাসএভারেজ সমস্যাটিকে সাধারণীকরণ করা যাক। নিম্নলিখিত প্রয়োজনীয়তাগুলি বিবেচনা করুন
বিবৃতি:

একটি ক্লাস-গড় প্রোগ্রাম তৈরি করুন যা প্রতিবার প্রোগ্রামটি কার্যকর করার সময় নির্দিষ্ট সংখ্যক গ্রেড প্রক্রিয়া করে।

প্রয়োজনীয়তা বিবৃতিতে গ্রেডগুলি কী বা কতগুলি তা উল্লেখ করা হয়নি, তাই আমরা ব্যবহারকারীকে গ্রেডগুলি প্রবেশ করতে বলব। প্রোগ্রামটি একটি নির্দিষ্ট সংখ্যক গ্রেড প্রক্রিয়া করে। ব্যবহারকারী একবারে একটি করে গ্রেড প্রবেশ করান যতক্ষণ না সমস্ত গ্রেড প্রবেশ করানো হয়, তারপর একটি সেন্টিনেল মান (যাকে সিগন্যাল মান, একটি ডামি মান বা একটি পতাকা মানও বলা হয়) প্রবেশ করান যাতে বোঝা যায় যে আর কোনও গ্রেড নেই।

সেন্টিনেল-নিয়ন্ত্রিত পুনরাবৃত্তি বাস্তবায়ন

নিম্নলিখিত স্ক্রিপ্টটি sentinelcontrolled পুনরাবৃত্তির সাথে ক্লাস গড় সমস্যা সমাধান করে।
লক্ষ্য করুন যে আমরা শূন্য দিয়ে ভাগ করার সম্ভাবনা পরীক্ষা করছি। যদি সনাক্ত না করা হয়, তাহলে এটি একটি মারাত্মক লজিক ত্রুটির কারণ হতে পারে। "ফাইল এবং ব্যতিক্রম" অধ্যায়ে, আমরা এমন প্রোগ্রাম লিখি যা
এই ধরনের ব্যতিক্রমগুলি সনাক্ত করে এবং যথাযথ পদক্ষেপ নেয়।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```

১ # class_average_sentinel.py ২ """সেন্টিনেল-নিয়ন্ত্রিত
পুনরাবৃত্তি সহ ক্লাস গড় প্রোগ্রাম!"""
৩
৪ # প্রাথমিককরণ পর্ব ৫ মোট = ০ # গ্রেডের যোগফল ৬
গ্রেড_কাউন্টার = ০ # প্রবেশ করানো গ্রেডের সংখ্যা ৭

৮ # প্রক্রিয়াকরণ পর্যায়
৯ গ্রেড = int(input('গ্রেড লিখুন, শেষে ১:')) # একটি গ্রেড পান
১০
১১ যাখন গ্রেড! = ১:
১২     মোট += গ্রেড
১৩     grade_counter += ১ grade =
১৪     int(input('গ্রেড লিখুন, শেষ পর্যন্ত ১:'))
১৫
১৬ # সমাপ্তি পর্ব
১৭ যদি গ্রেড_কাউন্টার != ০ হয়:

```

- ১৮ গড় = মোট / গ্রেড_কাউন্টার প্রিন্ট ('শ্রেণীর গড় {গড়:.২f}')
 ১৯
 আরও ২০ টি:
 ২১ মুদ্রণ করুন ('কোনও গ্রেড প্রবেশ করানো হয়নি')

কোড ইমেজ দেখতে এখানে ক্লিক করুন

গ্রেড লিখুন, ১ থেকে শেষ পর্যন্ত: ৯৭ গ্রেড লিখুন, ১ থেকে
 শেষ পর্যন্ত: ৮৮ গ্রেড লিখুন, ১ থেকে শেষ পর্যন্ত: ৭২ গ্রেড
 লিখুন, ১ থেকে শেষ পর্যন্ত: ১ ক্লাসের গড় ৮৫.৬৭

সেন্টিনেল-নিয়ন্ত্রিত পুনরাবৃত্তির জন্য প্রোগ্রাম লজিক

sentinelcontrolled iteration-এ, প্রোগ্রামটি while স্টেটমেন্টে পৌঁছানোর আগে প্রথম মান (লাইন ৯) পড়ে। লাইন ৯-এর মান ইনপুট নির্ধারণ করে যে প্রোগ্রামের নিয়ন্ত্রণ প্রবাহ while's suite-এ প্রবেশ করবে কিনা (লাইন 12-14)। যদি লাইন 11-এর শর্তটি False হয়, তাহলে ব্যবহারকারী sentinel মান (1) প্রবেশ করিয়েছেন, তাই স্যুটটি কার্যকর হয় না কারণ ব্যবহারকারী কোনও গ্রেড প্রবেশ করাননি। যদি শর্তটি True হয়, তাহলে স্যুটটি কার্যকর হয়, মোটের সাথে গ্রেড মান যোগ করে এবং বৃদ্ধি করে।

গ্রেড_কাউন্টার।

এরপর, লাইন ১৪ ব্যবহারকারীর কাছ থেকে আরেকটি গ্রেড ইনপুট করে এবং ব্যবহারকারীর দ্বারা প্রবেশ করানো সাম্প্রতিক গ্রেড ব্যবহার করে শর্ত (লাইন ১১) আবার পরীক্ষা করা হয়। প্রোগ্রামটি while শর্ত পরীক্ষা করার ঠিক আগে গ্রেডের মান সর্বদা ইনপুট করা হয়, তাই আমরা নির্ধারণ করতে পারি যে কেবলমাত্র ইনপুট করা মানটি গ্রেড হিসাবে প্রক্রিয়াকরণের আগে সেন্টিনেল কিনা।

যখন সেন্টিনেল মান ইনপুট করা হয়, তখন লুপটি বন্ধ হয়ে যায় এবং প্রোগ্রামটি মোটে -1 যোগ করে না। একটি সেন্টিনেলকন্ট্রোলড লুপে যা ব্যবহারকারীর ইনপুট সম্পাদন করে, যেকোনো প্রস্পট (লাইন ৯ এবং 14) ব্যবহারকারীকে সেন্টিনেল মান মনে করিয়ে দেবে।

দুই দশমিক স্থান দিয়ে শ্রেণীর গড় বিন্যাস করা

এই উদাহরণে দশমিক বিন্দুর ডানদিকে দুটি সংখ্যা দিয়ে ক্লাস গড় ফর্ম্যাট করা হয়েছে। একটি fstring-এ, আপনি এক্ষিকভাবে একটি কোলন (:) সহ একটি replacementtext এক্সপ্রেশন এবং একটি **format specifier** অনুসরণ করতে পারেন যা প্রতিস্থাপন টেক্সট কীভাবে ফর্ম্যাট করতে হয় তা বর্ণনা করে। বিন্যাস স্পেসিফায়ার .2f (লাইন 19) গড়কে দশমিক বিন্দু (.2) এর ডানদিকে দুটি সংখ্যা সহ একটি ভাসমান বিন্দু সংখ্যা (f) হিসাবে ফর্ম্যাট করে। এই উদাহরণে, গ্রেডগুলির যোগফল ছিল 257, যাকে 3 দিয়ে ভাগ করলে 85.666666666 পাওয়া যায়.... .2f দিয়ে গড় ফর্ম্যাট করলে এটি শততম অবস্থানে পূর্ণ হয়, প্রতিস্থাপন তৈরি হয়

টেক্সট ৮৫.৬৭। দশমিক বিল্ডুর ডানদিকে শুধুমাত্র একটি সংখ্যা সহ একটি গড় একটি **পরবর্তী শূন্য** দিয়ে ফর্ম্যাট করা হবে (যেমন, ৮৫.৫০)। "স্ট্রিংস: একটি গভীর চেহারা" অধ্যায়ে আরও অনেক স্ট্রিং ফর্ম্যাটিং বৈশিষ্ট্য নিয়ে আলোচনা করা হয়েছে।

৩.১০ বিল্ট-ইন ফাংশন রেঞ্জ: আরও গভীর চেহারা

ফাংশন রেঞ্জের দুটি এবং তিনটি আর্গুমেন্ট ভার্সনও রয়েছে। আপনি যেমন দেখেছেন, রেঞ্জের একটি আর্গুমেন্ট ভার্সন 0 থেকে আর্গুমেন্টের মান পর্যন্ত ধারাবাহিক পূর্ণসংখ্যার একটি ক্রম তৈরি করে, কিন্তু অন্তর্ভুক্ত করে না।
ফাংশন রেঞ্জের দুটি আর্গুমেন্ট ভার্সন তার প্রথম আর্গুমেন্টের মান থেকে দ্বিতীয় আর্গুমেন্টের মান পর্যন্ত ধারাবাহিক পূর্ণসংখ্যার একটি ক্রম তৈরি করে, যেমন:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

[1] তে: পরিসরে সংখ্যার জন্য (5, 10):

୯୮୧୬୯୫

ফাংশন রেঞ্জের তিনটি আর্গুমেন্ট সংস্করণ তার প্রথম আর্গুমেন্টের মান থেকে দ্বিতীয় আর্গুমেন্টের মান পর্যন্ত পূর্ণসংখ্যার একটি ক্রম তৈরি করে, কিন্তু অন্তর্ভুক্ত করে না, যা তৃতীয় আর্গুমেন্টের মান দ্বারা বৃদ্ধি পায়, যা [ধাপ হিসাবে](#) পরিচিত:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[2] তে: পরিসরে সংখ্যার জন্য (0, 10, 2):

...
...
মুদ্রণ (সংখ্যা, শেষ = ' ')

०८४६८

যদি তৃতীয় যুক্তি নেতিবাচক হয়, তাহলে ক্রমটি প্রথম যুক্তির মান থেকে দ্বিতীয় যুক্তির মান বাদ দিয়ে তৃতীয় যুক্তির মান হ্রাস করে, যেমন:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

[3] তে: পরিসরে সংখ্যার জন্য (10, 0, 2):

...
...
...
৬ ৪ ২

۸۶۸۰۹۲

৩.১১ আর্থিক পরিমাণের জন্য টাইপ ডেসিমাল ব্যবহার

এই বিভাগে, আমরা সুনির্দিষ্ট আর্থিক গণনার জন্য দশমিক ক্ষমতাগুলি উপস্থাপন করছি। আপনি যদি ব্যাংকিং বা অন্যান্য ক্ষেত্রের সাথে যুক্ত হন যেখানে "টুথেপেনি" নির্ভুলতার প্রয়োজন হয়, তাহলে আপনার দশমিকের ক্ষমতাগুলি গভীরভাবে অনুসন্ধান করা উচিত।

বেশিরভাগ বৈজ্ঞানিক এবং অন্যান্য গাণিতিক অ্যাপ্লিকেশনের ক্ষেত্রে যেখানে দশমিক বিলু সহ সংখ্যা ব্যবহার করা হয়, পাইথনের অন্তর্নির্মিত ভাসমান বিলু সংখ্যাগুলি ভাল কাজ করে। উদাহরণস্বরূপ, যখন আমরা 98.6 এর "স্বাভাবিক" শরীরের তাপমাত্রার কথা বলি, তখন আমাদের অনেকগুলি সংখ্যার সাথে সুনির্দিষ্ট হওয়ার প্রয়োজন হয় না। যখন আমরা থার্মোমিটারে তাপমাত্রা দেখি এবং এটি 98.6 হিসাবে পড়ি, তখন প্রকৃত মান 98.5999473210643 হতে পারে। এখানে মূল কথা হল যে এই সংখ্যাটিকে 98.6 বলা বেশিরভাগ শরীরের তাপমাত্রার অ্যাপ্লিকেশনের জন্য যথেষ্ট।

ফ্লোটিংপয়েন্ট মানগুলি বাইনারি ফর্ম্যাটে সংরক্ষণ করা হয় (আমরা প্রথম অধ্যায়ে বাইনারি চালু করেছি এবং অনলাইন "সংখ্যা সিস্টেম" পরিশিষ্টে এটি সম্পর্কে বিস্তারিত আলোচনা করেছি)। কিছু ফ্লোটিংপয়েন্ট মান কেবল তখনই আনুমানিকভাবে উপস্থাপন করা হয় যখন সেগুলিকে বাইনারিতে রূপান্তরিত করা হয়। উদাহরণস্বরূপ, ডলার এবং সেন্ট মান 112.31 সহ চলক পরিমাণ বিবেচনা করুন। যদি আপনি পরিমাণ প্রদর্শন করেন, তাহলে এটিতে আপনার নির্ধারিত মানটি ঠিক বলে মনে হচ্ছে:

[1] তে: পরিমাণ = 112.31

[2] তে: মুদ্রণ (পরিমাণ)
১১২.৩১

তবে, যদি আপনি দশমিক বিলুর ডানদিকে ২০টি সংখ্যার নির্ভুলতা দিয়ে পরিমাণ মুদ্রণ করেন, তাহলে আপনি দেখতে পাবেন যে মেমরিতে প্রকৃত ফ্লোটিংপয়েন্ট মান ঠিক ১১২.৩১ নয় - এটি কেবল একটি আনুমানিকতা:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

[3] তে: print(f'{amount:.20f}')
১১২.৩১০০০০০০০০০০০০০২২৭৩৭৪

অনেক অ্যাপ্লিকেশনের জন্য দশমিক বিলু সহ সংখ্যার সুনির্দিষ্ট উপস্থাপনা প্রয়োজন। ব্যাংকের মতো প্রতিষ্ঠান যারা প্রতিদিন লক্ষ লক্ষ এমনকি কোটি কোটি লেনদেন করে তাদের লেনদেন "পেনির সাথে" আবদ্ধ করতে হয়। ফ্লোটিংপয়েন্ট সংখ্যাগুলি কিছু আর্থিক পরিমাণকে উপস্থাপন করতে পারে কিন্তু সমস্ত নয়, সম্পূর্ণ নির্ভুলতার সাথে।

পাইথন **স্ট্যান্ডার্ড লাইব্রেরি** আপনার পাইথন কৌডে "চাকা পুনর্বীকরণ" এড়াতে অনেক পূর্বনির্ধারিত ক্ষমতা প্রদান করে। আর্থিক গণনা এবং অন্যান্য অ্যাপ্লিকেশনের জন্য যেখানে সংখ্যার সুনির্দিষ্ট উপস্থাপনা এবং হেরফের প্রয়োজন হয়।

দশমিক বিল্ডু সহ, পাইথন স্ট্যান্ডার্ড লাইব্রেরি **দশমিক টাইপ** প্রদান করে, যা টোথেপেনি নির্ভুলতার সমস্যা সমাধানের জন্য একটি বিশেষ কোডিং স্কিম ব্যবহার করে। এই স্কিমটিতে সংখ্যা ধরে রাখার জন্য অতিরিক্ত মেমরি এবং গণনা সম্পাদনের জন্য অতিরিক্ত প্রক্রিয়াকরণ সময় প্রয়োজন তবে আর্থিক গণনার জন্য প্রয়োজনীয় নির্ভুলতা প্রদান করে।

ব্যাংকগুলিকে অন্যান্য সমস্যাগুলির সাথেও মোকাবিলা করতে হয়, যেমন অ্যাকাউন্টের দৈনিক সুদের হিসাব করার সময় একটি ন্যায় রাউন্ডিং অ্যালগরিদম ব্যবহার করা। টাইপ ডেসিমাল এই ধরনের ক্ষমতা প্রদান করে।

^১ <https://docs.python.org/3.7/library/index.html>.

^২ আরও দশমিক মডিউল বৈশিষ্ট্যের জন্য, দেখুন

<https://docs.python.org/3.7/library/decimal.html>.

দশমিক মডিউল থেকে দশমিক প্রকার আমদানি করা হচ্ছে

আমরা বেশ কয়েকটি বিল্টইন টাইপ ব্যবহার করেছি—**int** (পূর্ণসংখ্যার জন্য, যেমন 10), **float** (ফ্লোটিংপয়েন্ট সংখ্যার জন্য, যেমন 7.5) এবং **str** ('পাইথন' এর মতো স্ট্রিং এর জন্য)। দশমিক টাইপ পাইথনে তৈরি করা হয় না। **বরং** এটি পাইথন স্ট্যান্ডার্ড লাইব্রেরির অংশ, যা **মডিউল** নামক সম্পর্কিত ক্ষমতার ফর্মে বিভক্ত। দশমিক মডিউল দশমিক টাইপ এবং এর ক্ষমতা সংজ্ঞায়িত করে।

দশমিক টাইপ ব্যবহার করতে, আপনাকে প্রথমে সম্পূর্ণ দশমিক মডিউলটি **আমদানি** করতে হবে, যেমনটি

দশমিক আমদানি করুন

এবং দশমিক ধরণটিকে দশমিক দশমিক হিসাবে উল্লেখ করুন, অথবা আপনাকে **আমদানি** থেকে আমদানি ব্যবহার করে আমদানি করার জন্য একটি নির্দিষ্ট ক্ষমতা নির্দেশ করতে হবে, যেমনটি আমরা এখানে করছি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: দশমিক আমদানি থেকে দশমিক

এটি দশমিক মডিউল থেকে শুধুমাত্র দশমিক টাইপটি আমদানি করে যাতে আপনি এটি আপনার কোডে ব্যবহার করতে পারেন। আমরা পরবর্তী অধ্যায় থেকে অন্যান্য আমদানি ফর্মগুলি নিয়ে আলোচনা করব।

দশমিক তৈরি করা

আপনি সাধারণত একটি স্ট্রিং থেকে একটি দশমিক তৈরি করেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[5] তে: মূল = দশমিক ('1000.00')

[6] তে: প্রধান

আউট[6]: দশমিক('1000.00')

[7] তে: হার = দশমিক ('0.05')

[8] তে: হার

আউট[8]: দশমিক('0.05')

আমরা শীঘ্ৰই এই চলকগুলিকে মূলধন এবং হারকে একটি চক্ৰবৃদ্ধি সুদে ব্যবহার কৰিব
হিসাব।

দশমিক পাটিগণিত

দশমিক মানক গাণিতিক অপারেটর +, , *, /, //, ** এবং %, এবং সংশ্লিষ্ট বৰ্বৰত অ্যাসাইনমেন্টগুলিকে সমৰ্থন কৰে:

[9] তে: x = দশমিক('10.5')

[10] তে: y = দশমিক('2')

[11] তে: x + y

আউট[11]: দশমিক ('12.5')

[12] তে: x // y

আউট[12]: দশমিক('5')

[13] তে: x += y

[14] তে: x

আউট[14]: দশমিক ('12.5')

আপনি দশমিক এবং পূর্ণসংখ্যার মধ্যে পাটিগণিত কৰিবলৈ পারেন, কিন্তু দশমিক এবং ফ্লোটিংপয়েন্ট সংখ্যার
মধ্যে নয়।

চক্ৰবৃদ্ধি-সুদের সমস্যার প্ৰয়োজনীয়তা বিবৃতি

সুনির্দিষ্ট আৰ্থিক গণনার জন্য দশমিক ধৰণ ব্যবহার কৰে চক্ৰবৃদ্ধি সুদের গণনা কৰা যাক। নিম্নলিখিত
প্ৰয়োজনীয়তা বিবৃতিটি বিবেচনা কৰুন:

একজন ব্যক্তি একটি সঞ্চয় অ্যাকাউন্টে \$1000 বিনিয়োগ কৰেন যা 5% সুদ প্ৰদান কৰে। ধৰে নিচি যে ব্যক্তি
অ্যাকাউন্টে জমাৰ উপৰ সমস্ত সুদ ৱেথে গোছেন, প্ৰতি বছৰ শেষে 10 বছৰের জন্য অ্যাকাউন্টে থাকা অৰ্থেৰ পৱিমাণ
গণনা কৰুন এবং প্ৰদৰ্শন কৰুন। এই পৱিমাণগুলি নিৰ্ধাৰণেৰ জন্য নিম্নলিখিত সূত্ৰটি ব্যবহার কৰুন:

$$a = p(1 + r)$$

কোথায়

p হল মূল বিনিয়োগকৃত পরিমাণ (অর্থাৎ, মূলধন),

r হল বার্ষিক সুদের হার,

n হলো বছরের সংখ্যা এবং

a হল নবম বছরের শেষে জমার পরিমাণ।

চক্রবৃদ্ধি সুদের হিসাব করা

এই সমস্যা সমাধানের জন্য, আসুন আমরা স্লিপেট [5] এবং [7] তে সংজ্ঞায়িত ভেরিয়েবল প্রিলিপাল এবং রেট ব্যবহার করি, এবং একটি ফর স্টেটমেন্ট ব্যবহার করি যা জমা থাকা অর্থের 10 বছরের প্রতিটির জন্য সুদের গণনা সম্পাদন করে। প্রতিটি বছরের জন্য, লুপটি বছরের নম্বর এবং সেই বছরের শেষে জমার পরিমাণ সহ একটি ফর্ম্যাটেড স্ট্রিং প্রদর্শন করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[15] তে: বছরের পরিসরে (1, 11):

$$\dots : \quad \text{পরিমাণ} = \text{মূলধন} * (\text{১} + \text{হার})^{** \text{ মুদ্রণ}(f\{\text{বছর}:>2\}\{\text{পরিমাণ}:>10.2f\})} \quad \text{বছর}$$

...:

...:

১ ১০৫০.০০

২ ১১০২.৫০

৩ ১১৫৭.৬২

৪ ১২১৫.৫১

৫ ১২৭৬.২৮

৬ ১৩৪০.১০

৭ ১৪০৭.১০

৮ ১৪৭৭.৪৬

৯ ১৫৫১.০৩

১০ ১৬২৮.৮৯

প্রয়োজনীয়তা বিবৃতি থেকে বীজগণিতীয় রাশি $(1 + r)$ লেখা হয়

$(\text{১} + \text{রেট})^{**}$

বছর

যেখানে চলক হার r কে প্রতিনিধিত্ব করে এবং চলক বছর n কে প্রতিনিধিত্ব করে।

জমার বছর এবং পরিমাণ বিন্যাস করা

কোড ইমেজ দেখতে এখানে লিংক করুন

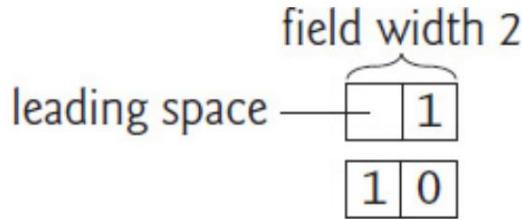
মুদ্রণ (f'{বছর:>2}{পরিমাণ:>10.2f}')

লুপের আউটপুট ফরম্যাট করার জন্য দুটি স্থানধারক সহ একটি fstring ব্যবহার করে।

স্থানধারক

{বছর:>2}

ফরম্যাট স্পেসিফায়ার >2 ব্যবহার করে বোঝানো হয় যে বছরের মানটি প্রস্তুত করে অক্ষরের অবস্থানের সংখ্যা নির্দিষ্ট করে। একক অক্ষের বছরের মান ১-৯ এর জন্য, ফরম্যাট স্পেসিফায়ার >2 মান অনুসরণ করে একটি স্পেস অক্ষের প্রদর্শন করে, এইভাবে প্রথম কলামে বছরগুলিকে ডানদিকে সারিবদ্ধ করে। নিম্নলিখিত চিত্রটি ২ এর ক্ষেত্রের প্রস্তুত ফর্ম্যাট করা ১ এবং ১০ সংখ্যাগুলি দেখায়:

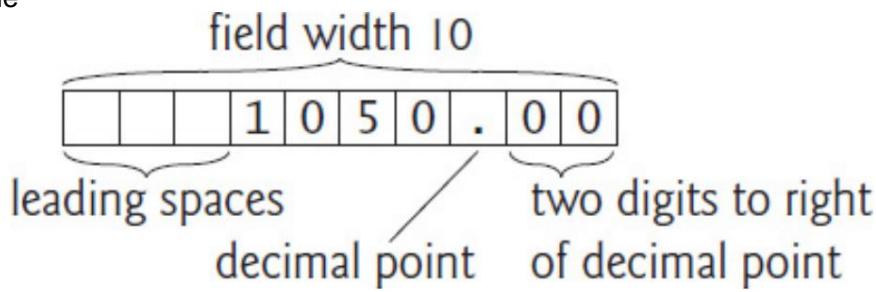


আপনি < দিয়ে মানগুলি বাম সারিবদ্ধ করতে পারেন।

স্থানধারকটিতে বিন্যাস স্পেসিফায়ার 10.2f

{পরিমাণ:>১০.২f}

পরিমাণকে একটি ভাসমান বিল্ডু সংখ্যা (f) ডান প্রান্তিক (>) হিসেবে বিন্যাস করা হয়, যেখানে দশমিক বিল্ডু (.2) এর ডানদিকে একটি দশমিক বিল্ডু এবং দুটি সংখ্যা থাকে। এইভাবে পরিমাণগুলি বিন্যাস করা তাদের দশমিক বিল্ডুগুলিকে উল্লম্বভাবে সারিবদ্ধ করে, যেমনটি আর্থিক পরিমাণের ক্ষেত্রে সাধারণত হয়। 10 অক্ষরের অবস্থানে, তিনটি ডানদিকের অক্ষের হল সংখ্যার দশমিক বিল্ডু এবং তার পরে দুটি সংখ্যা তার ডানদিকে। বাকি সাতটি অক্ষরের অবস্থান হল অগ্রবর্তী স্থান এবং দশমিক বিল্ডুর বাম দিকের সংখ্যা। এই উদাহরণে, সমষ্টি ডলারের পরিমাণের দশমিক বিল্ডুর বাম দিকে চারটি সংখ্যা রয়েছে, তাই প্রতিটি সংখ্যা তিনটি অগ্রবর্তী স্থান দিয়ে বিন্যাস করা হয়েছে। নিম্নলিখিত চিত্রটি 1050.00 মানের বিন্যাস দেখায়:



৩.১২ বিবৃতি বিরতি এবং অব্যাহত রাখুন

ব্রেক এবং কন্টিনিউ স্টেটমেন্ট লুপের নিয়ন্ত্রণ প্রবাহকে পরিবর্তন করে। কিছুক্ষণের মধ্যে বা অবিলম্বে ব্রেক স্টেটমেন্ট কার্যকর করলে সেই স্টেটমেন্টটি বেরিয়ে যায়। নিম্নলিখিত কোডে, রেঞ্জ পূর্ণসংখ্যা ক্রম 0-99 তৈরি করে, কিন্তু লুপটি শেষ হয়ে যায় যখন সংখ্যা ১০:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: **পরিসরে সংখ্যার জন্য** (100):

...: যদি সংখ্যা == ১০:

...: বিরতি

...: মুদ্রণ (সংখ্যা, শেষ = ' ')

...:

০ ১ ২ ৩ ৪ ৫ ৬ ৭ ৮ ৯

একটি স্ক্রিপ্টে, for লুপের পরের স্টেটমেন্টের সাথে এক্সিকিউশন চলতে থাকবে। while এবং for স্টেটমেন্টের প্রতিটিতে একটি এক্ষিক্ষিত else ক্লজ থাকে যা শুধুমাত্র তখনই কার্যকর হয় যখন লুপটি স্বাভাবিকভাবে বন্ধ হয়ে যায় - অর্থাৎ, ব্রেকের ফলে নয়।

কিছুক্ষণের মধ্যে অথবা for লুপে continue স্টেটমেন্ট এক্সিকিউট করলে লুপের বাকি অংশ এড়িয়ে যায়। কিছুক্ষণের মধ্যে, কন্টিনিউ পরীক্ষা করে দেখা হয় যে লুপ এক্সিকিউট করা চালিয়ে যাওয়া উচিত কিনা। for-এ, লুপ ক্রমের পরবর্তী আইটেমটি (যদি থাকে) প্রক্রিয়া করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[2] তে: **পরিসরে সংখ্যার জন্য** (10):

...: যদি সংখ্যা == ৫:

...: চালিয়ে যান

...: মুদ্রণ (সংখ্যা, শেষ = ' ')

...:

০ ১ ২ ৩ ৪ ৬ ৭ ৮ ৯

৩.১৩ বুলিয়ান অপারেটর এবং অথবা এবং না

শর্তসাপেক্ষ অপারেটর >, <, >=, <=, == এবং != ব্যবহার করে সহজ শর্ত তৈরি করা যেতে পারে যেমন grade >= 60। আরও জটিল শর্ত তৈরি করতে যা সহজ শর্তগুলিকে একত্রিত করে, বুলিয়ান অপারেটর ব্যবহার না করে and, or ব্যবহার করুন।

বুলিয়ান অপারেটর এবং

একটি কন্ট্রোল স্টেটমেন্টের স্যুট কার্যকর করার আগে দুটি শর্ত সত্য কিনা তা নিশ্চিত করার জন্য, শর্তগুলিকে একত্রিত করতে বুলিয়ান এবং অপারেটর ব্যবহার করুন। নিম্নলিখিত কোড দুটি ভেরিয়েবল সংজ্ঞায়িত করে, তারপর একটি শর্ত পরীক্ষা করে যা সত্য যদি এবং শুধুমাত্র যদি উভয় সরল শর্ত সত্য হয় - যদি সরল শর্তের যেকোনো একটি (অথবা উভয়) মিথ্যা হয়, তাহলে সম্পূর্ণ এবং এক্সপ্রেশনটি মিথ্যা:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: লিঙ্গ = 'মহিলা'

[2] তে: বয়স = 70

[3] তে: যদি লিঙ্গ == 'মহিলা' এবং বয়স >= 65:

...
মুদ্রণ ('বয়স্ক মহিলা')

...
বয়স্ক মহিলা

if স্টেটমেন্টের দুটি সহজ শর্ত রয়েছে:

- লিঙ্গ == 'মহিলা' নির্ধারণ করে যে একজন ব্যক্তি মহিলা কিনা এবং
- বয়স >= 65 নির্ধারণ করে যে ব্যক্তি একজন প্রবীণ নাগরিক কিনা।

and অপারেটরের বাম দিকের সরল অবস্থাটি প্রথমে মূল্যায়ন করে কারণ == এর অগ্রাধিকার and এর চেয়ে বেশি। প্রয়োজনে, and এর ডান দিকের সরল অবস্থাটি পরবর্তীতে মূল্যায়ন করে, কারণ >= এর অগ্রাধিকার and এর চেয়ে বেশি। (আমরা শীঘ্রই আলোচনা করব কেন and অপারেটরের ডান দিকটি শুধুমাত্র বাম দিকটি সত্য হলেই মূল্যায়ন করে।) সম্পূর্ণ if বিবৃতির শর্তটি সত্য যদি এবং শুধুমাত্র উভয় সরল শর্তই সত্য হয়। অতিরিক্ত বন্ধনী ঘোগ করে সম্মিলিত অবস্থাটি আরও স্পষ্ট করা যেতে পারে।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

(লিঙ্গ == 'মহিলা') এবং (বয়স >= ৬৫)

expression1 এবং expression2-এর জন্য False এবং True মানের সমষ্টি—যেমন টেবিল

সত্য সারণী বলা হয়:

এক্সপ্রেশন১ এক্সপ্রেশন২ এক্সপ্রেশন১ এবং এক্সপ্রেশন২		
মিথ্যা	মিথ্যা	মিথ্যা
মিথ্যা	সত্য	মিথ্যা
সত্য	মিথ্যা	মিথ্যা
সত্য	সত্য	সত্য

বুলিয়ান অপারেটর অথবা

দুটি শর্তের একটি অথবা উভয়ই সত্য কিনা তা পরীক্ষা করতে **বুলিয়ান বা অপারেটর** ব্যবহার করুন।

নিম্নলিখিত কোডটি এমন একটি শর্ত পরীক্ষা করে যা সত্য যদি উভয় বা উভয় সরল শর্তই হয়

সত্য—সমগ্র শর্তটি মিথ্যা, শুধুমাত্র যদি উভয় সরল শর্তই মিথ্যা হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: সেমিস্টার_গড় = 83

[5] তে: চূড়ান্ত_পরীক্ষা = 95

[6] তে: **যদি** semester_agreage >= 90 **অথবা** final_exam >= 90:

...:
 print('ছাত্র A গ্রেড পেয়েছে')

...:
শিক্ষার্থী A গ্রেড পেয়েছে

সিপেট [6] তে দুটি সহজ শর্তও রয়েছে:

- **semester_average >= 90** নির্ধারণ করে যে একজন শিক্ষার্থীর গড় A ছিল কিনা (90)

- `final_exam >= 90` নির্ধারণ করে যে একজন শিক্ষার্থীর `finale exam` গ্রেড A ছিল কিনা।

নীচের সত্য সারণীটি বুলিয়ান বা অপারেটরের সারসংক্ষেপ তুলে ধরে। অপারেটর এবং এর উচ্চতর অথবা এর চেয়ে অগ্রাধিকার।

expression1 expression2 expression1 বা expression2		
মিথ্যা	মিথ্যা	মিথ্যা
মিথ্যা	সত্য	সত্য
সত্য	মিথ্যা	সত্য
সত্য	সত্য	সত্য

শর্ট-সার্কিট মূল্যায়নের মাধ্যমে কর্মক্ষমতা উন্নত করা

পাইথন যখনই জানতে পারে যে সম্পূর্ণ কিনা তখনই একটি এবং এক্সপ্রেশন মূল্যায়ন করা বন্ধ করে দেয় শর্টটি মিথ্যা। একইভাবে, পাইথন একটি or এক্সপ্রেশন মূল্যায়ন করা বন্ধ করে দেয় যখনই এটি সম্পূর্ণ অবস্থাটি সত্য কিনা তা জানে। একে শর্টসার্কিট মূল্যায়ন বলা হয়। তাই

অবস্থা

কোড ইমেজ দেখতে এখানে ক্লিক করুন

লিঙ্গ == 'মহিলা' এবং বয়স >= ৬৫

লিঙ্গ 'মহিলা'-এর সমান না হলে অবিলম্বে মূল্যায়ন বন্ধ করে দেয় কারণ সমগ্র অভিব্যক্তি অবশ্যই মিথ্যা হতে হবে। যদি লিঙ্গ 'মহিলা'-এর সমান হয়, তাহলে মৃত্যুদণ্ড কার্যকর করা হবে, কারণ বয়স 65 এর চেয়ে বেশি বা সমান হলে সম্পূর্ণ রাশিটি সত্য হবে।

একইভাবে, শর্ট

কোড ইমেজ দেখতে এখানে লিংক করুন

সেমিস্টার_গড় >= ৯০ অথবা ফাইনাল_পরীক্ষা >= ৯০

যদি semester_average 90 এর চেয়ে বড় বা সমান হয় তবে তাংক্ষণিকভাবে মূল্যায়ন বন্ধ করে দেয় কারণ সম্পূর্ণ এক্সামেনটি অবশ্যই True হতে হবে। যদি semester_average 90 এর চেয়ে কম হয়, তাহলে এক্সিকিউশন চলতে থাকে, কারণ final_exam 90 এর চেয়ে বড় বা সমান হলে এক্সামেনটি True হতে পারে।

যেসব এক্সামেনে এবং ব্যবহার করা হয়, সেগুলোতে False হওয়ার সম্ভাবনা বেশি, এমন শর্তটিকে বাম দিকের শর্ত হিসেবে ব্যবহার করা হয়। অথবা অপারেটর এক্সামেনে, যেসব কন্ডিশনে True হওয়ার সম্ভাবনা বেশি, সেগুলোকে বাম দিকের শর্ত হিসেবে ব্যবহার করা হয়। এই কৌশলগুলি একটি প্রোগ্রামের এক্সিকিউশন করাতে পারে।
সময়।

বুলিয়ান অপারেটর নয়

বুলিয়ান অপারেটর কোনও শর্তের অর্থ "বিপরীত" করে না - সত্য মিথ্যা হয়ে যায় এবং মিথ্যা সত্য হয়ে যায়। এটি একটি ইউনারি অপারেটর - এর কেবল একটি অপারেন্ট রয়েছে।

যদি মূল শর্তটি (not অপারেটর ছাড়া) False হয়, যেমন নিম্নলিখিত কোডে, তাহলে কার্যকর করার পথ বেছে নেওয়ার জন্য আপনি একটি শর্তের আগে not অপারেটর স্থাপন করেন:

কোড ইমেজ দেখতে এখানে লিংক করুন

[7] তে: গ্রেড = ৮৭

[8] তে: যদি গ্রেড না হয় == ১:

```
...:           print(' পরবর্তী গ্রেড হল', গ্রেড)
...:
```

পরবর্তী গ্রেড হল

৮৭

প্রায়শই, আপনি "নট" ব্যবহার এড়াতে পারেন শর্তটিকে আরও "স্বাভাবিক" বা সুবিধাজনক উপায়ে প্রকাশ করে।

উদাহরণস্বরূপ, পূর্ববর্তী if বিবৃতিটিকে "" হিসাবেও লেখা যেতে পারে

অনুসরণ করে:

কোড ইমেজ দেখতে এখানে লিংক করুন

[9] তে: if grade != ১: print(' পরবর্তী গ্রেড

```
...:           হল', grade)
...:
```

পরবর্তী গ্রেড হল

৮৭

নীচের সত্য টেবিলটি not অপারেটরের সারসংক্ষেপ তুলে ধরেছে।

অভিব্যক্তি, অভিব্যক্তি নয়	
মিথ্যা	সত্য
সত্য	মিথ্যা

নিম্নলিখিত সারণিতে প্রবর্তিত অপারেটরগুলির অগ্রাধিকার এবং গোষ্ঠীকরণ দেখানো হয়েছে যাতে
অনেক দূরে, উপর থেকে নীচে, অগ্রাধিকারের ক্রমস্থান ক্রমে।

অপারেটর		গ্রাফিং
()		বাম থেকে ডানে
**		ডান থেকে বামে
* / // %		বাম থেকে ডানে
+		বাম থেকে ডানে
<= > > == !=		বাম থেকে ডানে
না		বাম থেকে ডানে

এবং	বাম থেকে ডানে
অথবা	বাম থেকে ডানে

৩.১৪ তথ্য বিজ্ঞানের ভূমিকা: কেন্দ্রীয় প্রবণতার পরিমাপ—মধ্যম, মধ্যম এবং মোড

এখানে আমরা পরিসংখ্যান ব্যবহার করে তথ্য বিশ্লেষণের বিষয়ে আমাদের আলোচনা চালিয়ে যাচ্ছি, যার মধ্যে রয়েছে আরও কিছু বর্ণনামূলক পরিসংখ্যান, যার মধ্যে রয়েছে:

- **গড়—মানগুলির একটি সেটের গড় মান।**
- **মধ্যম—** যখন সমস্ত মান সাজানো ক্রমে সাজানো হয় তখন মধ্যম মান।
- **মোড—**সবচেয়ে ঘন ঘন ঘটমান মান।

এগুলো হলো **কেন্দ্রীয় প্রবণতার পরিমাপ—প্রতিটি** হলো একটি একক মান তৈরির একটি উপায় যা মানগুলির একটি সেটের মধ্যে একটি "কেন্দ্রীয়" মানকে প্রতিনিধিত্ব করে, অর্থাৎ, এমন একটি মান যা কিছু অর্থে অন্যগুলির বৈশিষ্ট্য।

চলুন পূর্ণসংখ্যার তালিকার গড়, মধ্যম এবং মোড গণনা করি। পরবর্তী সেশনটি গ্রেড নামে একটি তালিকা তৈরি করে, তারপর বিল্টইন **sum** এবং **len** ফাংশন ব্যবহার করে "হাতে" গড় গণনা করে - **sum** গ্রেডের মোট সংখ্যা (397) গণনা করে এবং **len** গ্রেডের সংখ্যা (5) প্রদান করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: গ্রেড = [85, 93, 45, 89, 85]

[2] তে: sum(grades) / len(grades)

আউট[2]: ৭৯.৮

পূর্ববর্তী অধ্যায়ে বর্ণনামূলক পরিসংখ্যান গণনা এবং যোগফল উল্লেখ করা হয়েছে—যা পাইথনে বিল্টইন ফাংশন **len** এবং **sum** হিসেবে প্রয়োগ করা হয়েছে। **min** এবং **max** ফাংশনের মতো (পূর্ববর্তী অধ্যায়ে প্রবর্তিত), **sum** এবং **len** উভয়ই ফাংশনালস্টাইল প্রোগ্রামিং ত্রাসের উদাহরণ — **এরা** মানগুলির সংগ্রহকে একটি একক মানে ত্রাস করে—যথাক্রমে এই মানগুলির যোগফল এবং মানের সংখ্যা।

.৪ এর classeaverage উদাহরণে, আমরা স্ক্রিপ্টের ১০-১৫ নম্বর লাইন মুছে ফেলতে পারতাম এবং ১৬ নম্বর লাইনে average স্লিপেট [2] এর গণনা দিয়ে প্রতিস্থাপন করতে পারতাম।

পাইথন স্ট্যান্ডার্ড লাইব্রেরির **পরিসংখ্যান মডিউলটি** গড়, মধ্যমা এবং মোড গণনা করার জন্য ফাংশন প্রদান করে - এগুলি হ্রাস। এই ক্ষমতাগুলি ব্যবহার করতে, প্রথমে পরিসংখ্যান মডিউলটি আমদানি করুন:

[3] তে: [আমদানি](#) পরিসংখ্যান

তারপর, আপনি "পরিসংখ্যান" দিয়ে মডিউলের ফাংশনগুলি অ্যাক্সেস করতে পারবেন। এরপর কল করার জন্য ফাংশনের নাম লিখুন। নিম্নলিখিতটি পরিসংখ্যান মডিউলের গড়, মধ্যমা এবং মোড ফাংশন ব্যবহার করে গ্রেড তালিকার গড়, মধ্যমা এবং মোড গণনা করে :

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: statistics.mean(grades)
আউট[4]: ৭৯.৮

[5] তে: statistics.median(grades)
আউট[5]: ৮৫

[6] তে: statistics.mode(grades)
আউট[6]: ৮৫

প্রতিটি ফাংশনের আর্গুমেন্ট অবশ্যই পুনরাবৃত্তিযোগ্য হতে হবে—এই ক্ষেত্রে, তালিকার গ্রেড। মধ্যমা এবং মোড সঠিক কিনা তা নিশ্চিত করতে, আপনি **builtin sorted** ফাংশন ব্যবহার করে গ্রেডের একটি কপি পেতে পারেন যার মান ক্রমবর্ধমান ক্রমে সাজানো আছে:

[7] তে: সাজানো (গ্রেড)
আউট[7]: [৪৫, ৮৫, ৮৫, ৮৯, ৯৩]

গ্রেড তালিকার মান বিজোড় সংখ্যক (৫) আছে, তাই মধ্যমা মধ্যম মান প্রদান করে (৮৫)। যদি তালিকার মান সংখ্যা জোড় হয়, তাহলে মধ্যমা দুটি মধ্যম মানের গড় প্রদান করে। সাজানো মানগুলি অধ্যয়ন করলে, আপনি দেখতে পাবেন যে ৮৫ হল মোড কারণ এটি সবচেয়ে ঘন ঘন ঘটে (দুবার)। মোড ফাংশনটি StatisticsError এর জন্য একটি কারণ তৈরি করে তালিকা যেমন

[৮৫, ৯৩, ৮৫, ৮৯, ৮৫, ৯৩]

যেখানে দুটি বা ততোধিক "সবচেয়ে ঘন ঘন" মান থাকে। এই ধরণের মানগুলির সেটকে বলা হয়

৩.১৫ র্যাপ-আপ

এই অধ্যায়ে, আমরা পাইথনের নিয়ন্ত্রণ বিবৃতি নিয়ে আলোচনা করেছি, যার মধ্যে রয়েছে if, if... else, if... elif... else, while, for, break এবং continue। আপনি দেখেছেন যে for স্টেটমেন্ট sequencecontrolled iteration সম্পাদন করে—এটি প্রতিটি আইটেমকে একটি iterable-এ প্রক্রিয়া করে, যেমন পূর্ণসংখ্যার পরিসর, একটি স্ট্রিং বা একটি তালিকা। আপনি 0 থেকে শুরু করে তার আর্গুমেন্ট অন্তর্ভুক্ত না করে পূর্ণসংখ্যার ক্রম তৈরি করতে এবং a for স্টেটমেন্ট করতার পুনরাবৃত্তি হয় তা নির্ধারণ করতে বিল্টইন ফাংশন range ব্যবহার করেছেন।

আপনি while স্টেটমেন্টের সাথে sentinelcontrolled iteration ব্যবহার করে একটি লুপ তৈরি করেছেন যা একটি sentinel মান না পাওয়া পর্যন্ত কার্যকর হতে থাকে। আপনি প্রথম আর্গুমেন্টের মান থেকে দ্বিতীয় আর্গুমেন্টের মান পর্যন্ত পূর্ণসংখ্যার ক্রম তৈরি করতে বিল্টইন ফাংশন range রেঞ্জের twoargument সংস্করণ ব্যবহার করেছেন, কিন্তু অন্তর্ভুক্ত করেননি। আপনি threeargument সংস্করণটিও ব্যবহার করেছেন যেখানে তৃতীয় আর্গুমেন্টটি একটি রেঞ্জের পূর্ণসংখ্যার মধ্যে ধাপ নির্দেশ করে।

আমরা সুনির্দিষ্ট আর্থিক গণনার জন্য দশমিক প্রকারটি চালু করেছি এবং চক্রবৃদ্ধি সুদ গণনা করার জন্য এটি ব্যবহার করেছি। আপনি ফর্ম্যাটেড আউটপুট তৈরি করতে fstrings এবং বিভিন্ন ফর্ম্যাট স্পেসিফায়ার ব্যবহার করেছেন। লুপগুলিতে নিয়ন্ত্রণের প্রবাহ পরিবর্তন করার জন্য আমরা break এবং continue বিবৃতি চালু করেছি। আমরা বুলিয়ান অপারেটরগুলি নিয়ে আলোচনা করেছি এবং, অথবা এবং নয় এমন শর্ত তৈরি করার জন্য যা সহজ শর্তগুলিকে একত্রিত করে।

পরিশেষে, আমরা বর্ণনামূলক পরিসংখ্যান নিয়ে আমাদের আলোচনা অব্যাহত রেখেছি কেন্দ্রীয় প্রবণতার পরিমাপ - গড়, মধ্যমা এবং মোড - প্রবর্তন করে এবং পাইথন স্ট্যান্ডার্ড লাইব্রেরির পরিসংখ্যান মডিউল থেকে ফাংশন ব্যবহার করে সেগুলি গণনা করে।

পরবর্তী অধ্যায়ে, আপনি কাস্টম ফাংশন তৈরি করবেন এবং পাইথনের গণিত এবং র্যান্ডম মডিউল থেকে বিদ্যমান ফাংশনগুলি ব্যবহার করবেন। আমরা বেশ কয়েকটি পূর্বনির্ধারিত ফাংশনাল-প্রোগ্রামিং ত্রাস দেখাব এবং আপনি অতিরিক্ত ফাংশনাল প্রোগ্রামিং ক্ষমতা দেখতে পাবেন।

লে-লিস্ট

গল্প . কার্যাবলী

উদ্দেশ্যমূলক মতামত

এই অধ্যায়ে, আপনি পাথ অর্জন
করবেন

- কাস্টম ফাংশন তৈরি করুন। অফার্স এবং
ডিল
- কোড পুনঃব্যবহারের জন্য এবং "চাকা পুনর্বীকরণ" এড়াতে পাইথন স্ট্যান্ডার্ড লাইব্রেরি মডিউল, যেমন র্যান্ডম এবং গণিত, আমদানি করুন
এবং
ব্যবহার করুন।

ফাংশনগুলির মধ্যে ভেট্টা স্থানান্তর করুন।

সমর্থন এলোমেলো সংখ্যার একটি পরিসর তৈরি করুন।

সাইন আউট র্যান্ডম নম্বর জেনারেশন ব্যবহার করে সিমুলেশন কৌশলগুলি দেখুন।

- পুনরুৎপাদনযোগ্যতা নিশ্চিত করতে এলোমেলো সংখ্যা জেনারেটরটি বীজ করুন।
- একটি টুপলে মান প্যাক করুন এবং একটি টুপল থেকে মান আনপ্যাক করুন।
- একটি ফাংশন থেকে একটি টুপলের মাধ্যমে একাধিক মান ফেরত পাঠান।
- আপনার প্রোগ্রামে আপনি কোথায় ব্যবহার করতে পারবেন তা কীভাবে একজন শনাক্তকারীর স্কোপ নির্ধারণ করে তা বুঝুন
এটা।
- ডিফল্ট প্যারামিটার মান সহ ফাংশন তৈরি করুন।
- কীওয়ার্ড আর্গুমেন্ট সহ ফাংশন কল করুন।
- এমন ফাংশন তৈরি করুন যা যেকোনো সংখ্যক আর্গুমেন্ট গ্রহণ করতে পারে।
- একটি বস্তুর পদ্ধতি ব্যবহার করুন।
- একটি রিকার্সিভ ফাংশন লিখুন এবং ব্যবহার করুন।

.1 ভূমিকা

.2 ফাংশন সংজ্ঞায়িত করা

.3 একাধিক পরামিতি সহ ফাংশন

.8 র্যান্ডম নাম্বার জেনারেশন

.5 কেস স্টাডি: সুযোগের খেলা

.6 পাইথন স্ট্যান্ডার্ড লাইব্রেরি

.7 গণিত মডিউল ফাংশন

.8 আবিষ্কারের জন্য আইপিথন ট্যাব সমাপ্তি ব্যবহার করা

.9 ডিফল্ট প্যারামিটার মান

.10 কীওয়ার্ড আর্গুমেন্ট

.11 স্বেচ্ছাচারী ঘুঙ্কি তালিকা

.12 পদ্ধতি: বন্ধুর সাথে সম্পর্কিত ফাংশন

.13 সুযোগের নিয়ম

.14 আমদানি: আরও গভীর দৃষ্টিভঙ্গি

.15 ফাংশনগুলিতে ঘুঙ্কি উপস্থাপন: আরও গভীরভাবে দেখা

.16 পুনরাবৃত্তি

.17 ফাংশনাল স্টাইল প্রোগ্রামিং

.18 তথ্য বিজ্ঞানের ভূমিকা: বিচ্ছুরণের পরিমাপ

.19 সারসংক্ষেপ

8.1 ভূমিকা

এই অধ্যায়ে, আমরা কাস্টম ফাংশন এবং সম্পর্কিত বিষয়গুলির সাথে পাইথনের মৌলিক বিষয়গুলি নিয়ে আলোচনা চালিয়ে যাব। আমরা পাইথন স্ট্যান্ডার্ড লাইব্রেরির র্যান্ডম মডিউল এবং র্যান্ডমবার জেনারেশন ব্যবহার করে একটি সিঙ্ক্রেশন সাইডেড ডাই রোলিং সিমুলেট করব। আমরা একটি স্লিপেট কাস্টম ফাংশন এবং র্যান্ডমবার জেনারেশন একত্রিত করব যা ডাইস গেম ক্র্যাপস বাস্তবায়ন করে। এই উদাহরণে, আমরা পাইথনের টুপল সিকোয়েন্স টাইপও প্রবর্তন করব এবং একটি ফাংশন থেকে একাধিক মান ফেরত দেওয়ার জন্য টুপল ব্যবহার করব। পুনরুৎপাদনযোগ্যতা নিশ্চিত করতে আমরা র্যান্ডম নম্বর জেনারেটর সিডিং নিয়ে আলোচনা করব।

আপনি পাইথন স্ট্যান্ডার্ড লাইব্রেরির গণিত মডিউলটি আমদানি করবেন, তারপর এটি ব্যবহার করে IPython ট্যাব সমাপ্তি সম্পর্কে শিখবেন, যা আপনার কোডিং এবং আবিষ্কার প্রক্রিয়াগুলিকে দ্রুততর করবে। আপনি ডিফল্ট প্যারামিটার মান সহ ফাংশন তৈরি করবেন, কীওয়ার্ড আর্গুমেন্ট সহ ফাংশন কল করবেন এবং ইচ্ছামত আর্গুমেন্ট তালিকা সহ ফাংশন সংজ্ঞায়িত করবেন। আমরা অবজেক্টের কলিং পদ্ধতিগুলি প্রদর্শন করব। আমরা আলোচনা করব যে কীভাবে একটি শনাক্তকারীর স্কোপ নির্ধারণ করে যে আপনি আপনার প্রোগ্রামে কোথায় এটি ব্যবহার করতে পারেন।

আমরা মডিউল আমদানি করার পদ্ধতিটি আরও গভীরভাবে দেখব। আপনি দেখতে পাবেন যে আর্গুমেন্টগুলি ফাংশনের সাথে সম্পর্কিত। আমরা একটি রিকার্সিভ ফাংশনও প্রদর্শন করব এবং পাইথনের ফাংশনালস্টাইল প্রোগ্রামিং ক্ষমতা উপস্থাপন শুরু করব।

ডেটা সায়েন্সের ভূমিকা বিভাগে, আমরা বর্ণনামূলক পরিসংখ্যান নিয়ে আমাদের আলোচনা চালিয়ে যাব, বিচ্ছুরণের পরিমাপ—প্রকরণ এবং মানক বিচ্ছুরণ—প্রবর্তনের মাধ্যমে এবং পাইথন স্ট্যান্ডার্ড লাইব্রেরির পরিসংখ্যান থেকে ফাংশন ব্যবহার করে সেগুলি গণনা করে।
মডিউল।

৪.২ ফাংশন সংজ্ঞায়িত করা

আপনি অনেক বিল্টইন ফাংশন (int, float, print, input, type, sum, len, min এবং max) এবং পরিসংখ্যান মডিউল থেকে কিছু ফাংশন (mean, median এবং মোড়)। প্রতিটি একটি একক, সুনির্দিষ্ট কাজ সম্পাদন করে। আপনি প্রায়শই কাস্টম ফাংশন সংজ্ঞায়িত এবং কল করবেন।
পরবর্তী সেশনটি একটি বর্গ ফাংশন সংজ্ঞায়িত করে যা তার আর্গুমেন্টের বর্গ গণনা করে। তারপর এটি ফাংশনটিকে দুবার কল করে - একবার int মান 7 বর্গ করতে (int মান 49 তৈরি করে) এবং একবার ফ্লোট মান 2.5 বর্গ করতে (ফ্লোট মান 6.25 তৈরি করে):

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: def square(number):
...:     """সংখ্যার বর্গ গণনা করো।"""
...:     return number ** 2
...:
```

[2] তে: বর্গ(7)

আউট[2]: 49

[3] তে: বর্গ(2.5)

আউট[3]: 6.25

প্রথম স্লিপেটে ফাংশন সংজ্ঞায়িত বিবৃতিগুলি কেবল একবার লেখা হয়, তবে একটি প্রোগ্রাম জুড়ে বিভিন্ন পয়েন্ট থেকে এবং যতবার খুশি "তাদের কাজ করতে" বলা যেতে পারে। 'হ্যালো' এর মতো অ-সংখ্যাসূচক ঘুঙ্কি দিয়ে square কল করলে TypeError হয় কারণ exponentiation অপারেটর (** শুধুমাত্র সংখ্যাসূচক মান নিয়ে কাজ করে।

একটি কাস্টম ফাংশন সংজ্ঞায়িত করা

একটি ফাংশনের সংজ্ঞা (যেমন স্লিপেট [1]-এ square) def কীওয়ার্ড দিয়ে শুরু হয়, তারপরে ফাংশনের নাম (square), বন্ধনীর একটি সেট এবং একটি কোলন (:) থাকে। ভেরিয়েবল শনাক্তকারীর মতো, প্রচলিত ফাংশনের নামগুলি একটি ছোট হাতের অক্ষর দিয়ে শুরু হওয়া উচিত এবং বহু-শব্দের নামগুলিতে আভারঙ্গনগুলি প্রতিটি শব্দকে আলাদা করা উচিত।

প্রয়োজনীয় বন্ধনীতে ফাংশনের প্যারামিটার তালিকা থাকে— কমা দিয়ে পৃথক করা প্যারামিটারের একটি তালিকা যা ফাংশনটির কাজ সম্পাদনের জন্য প্রয়োজনীয় ডেটা উপস্থাপন করে।

ফাংশন স্কোয়ারের শুধুমাত্র একটি প্যারামিটার আছে যার নাম সংখ্যা—যে মানটি বর্গ করতে হবে। যদি বন্ধনীগুলি খালি থাকে, তাহলে ফাংশনটি তার কাজ সম্পাদনের জন্য প্যারামিটার ব্যবহার করে না।

কোলন (:) এর পরে ইন্ডেন্ট করা রেখাগুলি হল ফাংশনের ব্লক, যা একটি দ্বারা গঠিত ঐচ্ছিক docstring এর পরে ফাংশনের কাজ সম্পাদনকারী স্টেটমেন্ট থাকে। আমরা শীত্রুই একটি ফাংশনের ব্লক এবং একটি কন্ট্রোল স্টেটমেন্টের সুটের মধ্যে পার্থক্যটি তুলে ধরব।

একটি কাস্টম ফাংশনের ডকস্ট্রিং নির্দিষ্ট করা

পাইথন কোডের স্টাইল গাইডে বলা হয়েছে যে ফাংশনের ব্লকের প্রথম লাইনটি একটি ডকস্ট্রিং হওয়া উচিত যা ফাংশনের উদ্দেশ্য সংক্ষেপে ব্যাখ্যা করে:

"""সংখ্যার বর্গ গণনা করো!"""

আরও বিস্তারিত জানার জন্য, আপনি একটি বহু-লাইন ডকস্ট্রিং ব্যবহার করতে পারেন—স্টাইল গাইডটি একটি সংক্ষিপ্ত ব্যাখ্যা দিয়ে শুরু করার পরামর্শ দেয়, তারপরে একটি ফাঁকা লাইন এবং অতিরিক্ত বিবরণ।

একটি ফাংশনের কলারে একটি ফলাফল ফেরত দেওয়া

যখন একটি ফাংশন কার্যকর করা শেষ করে, তখন এটি তার কলারে নিয়ন্ত্রণ ফেরত দেয়—অর্থাৎ, কোডের লাইন যা ফাংশনটিকে কল করেছিল। স্কয়ার ব্লকে, **রিটার্ন** স্টেটমেন্ট:

রিটার্ন নম্বর ** ২

প্রথমে সংখ্যাটিকে বর্গ করে, তারপর ফাংশনটি বন্ধ করে এবং ফলাফলটি কলারকে ফেরত দেয়। এই উদাহরণে, প্রথম কলারটি স্লিপেট [2] তে রয়েছে, তাই IPython ফলাফলটি Out[2] তে প্রদর্শন করে। দ্বিতীয় কলারটি স্লিপেট [3] তে রয়েছে, তাই IPython ফলাফলটি প্রদর্শন করে আউট[3]।

ফাংশন কলগুলি এক্সপ্রেশনেও এমবেড করা যেতে পারে। নিম্নলিখিত কোডটি প্রথমে square কল করে, তারপর print ফলাফল প্রদর্শন করে:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

```
[4] তে: print(' ৭ এর বর্গ হল', square(7))
```

```
৭ এর বর্গ ৪৯।
```

একটি ফাংশন থেকে তার কলারে নিয়ন্ত্রণ ফিরিয়ে আনার আরও দুটি উপায় রয়েছে:

- এক্সপ্রেশন ছাড়া রিটার্ন স্টেটমেন্ট এক্সিকিউট করলে ফাংশনটি বন্ধ হয়ে যায় এবং পরোক্ষভাবে কলকারীকে None মান ফেরত দেয়। পাইথন ডকুমেন্টেশনে বলা হয়েছে যে None একটি মানের অনুপস্থিতি প্রতিনিধিত্ব করে। None শর্তে False হিসাবে মূল্যায়ন করে।
- যখন কোন ফাংশনে কোন রিটার্ন স্টেটমেন্ট থাকে না, তখন ফাংশনের ব্লকের শেষ স্টেটমেন্টটি কার্যকর করার পরে এটি পরোক্ষভাবে None মানটি প্রদান করে।

স্থানীয় ভেরিয়েবল

যদিও আমরা স্কয়ার ব্লকে ভেরিয়েবল সংজ্ঞায়িত করিনি, তবুও এটি করা সম্ভব। একটি ফাংশনের প্যারামিটার এবং তার ব্লকে সংজ্ঞায়িত ভেরিয়েবলগুলি হল **স্থানীয় ভেরিয়েবল - এগুলি** কেবল ফাংশনের ভিতরে ব্যবহার করা যেতে পারে এবং শুধুমাত্র ফাংশনটি কার্যকর করার সময় বিদ্যমান থাকে।
একটি স্থানীয় ভেরিয়েবলকে তার ফাংশন ব্লকের বাইরে অ্যাক্সেস করার চেষ্টা করলে একটি NameError দেখা দেয়, যা নির্দেশ করে যে ভেরিয়েবলটি সংজ্ঞায়িত নয়।

IPython এর সাহায্য ব্যবস্থার মাধ্যমে একটি ফাংশনের ডকস্ট্রিং অ্যাক্সেস করা

আইপিথন আপনাকে আপনার কোডে ব্যবহার করার জন্য তৈরি মডিউল এবং ফাংশন সম্পর্কে জানতে সাহায্য করতে পারে, সেইসাথে আইপিথন নিজেই। উদাহরণস্বরূপ, কোনও ফাংশনের ডকস্ট্রিং দেখতে, ফাংশনটি কীভাবে ব্যবহার করবেন তা শিখতে, ফাংশনের নাম টাইপ করুন এবং তারপরে একটি **প্রশ্নবোধক চিহ্ন (?)** লিখুন:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

[5] তে: বর্গক্ষেত্র?

স্বাক্ষর: বর্গক্ষেত্র (সংখ্যা)

ডকস্ট্রিং: সংখ্যার বর্গ গণনা করুন।

ফাইল: ~/ডকুমেন্টস/উদাহরণ/ch04/<ipythoninput17268c8ff93a9>

প্রকার: ফাংশন

আমাদের বর্গ ফাংশনের জন্য, প্রদর্শিত তথ্যের মধ্যে রয়েছে:

- ফাংশনের নাম এবং প্যারামিটার তালিকা—যা এর স্বাক্ষর নামে পরিচিত।
- ফাংশনের ডকস্ট্রিং।
- ফাংশনের সংজ্ঞা সম্বলিত ফাইলের নাম। একটি ইন্টারেক্টিভ সেশনে একটি ফাংশনের জন্য, এই লাইনটি সেই স্লিপেটের তথ্য দেখায় যা ফাংশনটিকে সংজ্ঞায়িত করে - "<ipythoninput17268c8ff93a9>" এর মধ্যে 1 এর অর্থ স্লিপেট [1]।
- আপনি যে ধরণের আইটেমের জন্য IPython এর সাহায্য ব্যবস্থা অ্যাক্সেস করেছেন - এই ক্ষেত্রে, একটি ফাংশন।

যদি ফাংশনের সোর্স কোড IPython থেকে অ্যাক্সেসযোগ্য হয়—যেমন বর্তমান সেশনে সংজ্ঞায়িত কোনও ফাংশন অথবা .py ফাইল থেকে সেশনে আমদানি করা কোনও ফাংশন—তবে আপনি ফাংশনের সম্পূর্ণ সোর্সকোড সংজ্ঞা প্রদর্শন করতে ?? ব্যবহার করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6] তে: বর্গ??

স্বাক্ষর: বর্গক্ষেত্র (সংখ্যা)

উৎস:

ডিফ বর্গক্ষেত্র (সংখ্যা):

"""সংখ্যার বর্গ গণনা করো!"""

রিটার্ন নম্বর ** ২

ফাইল: ~/ডকুমেন্টস/উদাহরণ/ch04/<ipythoninput17268c8ff93a9>

প্রকার: ফাংশন

যদি আইপিথন থেকে সোর্স কোড অ্যাক্সেসযোগ্য না হয়, তাহলে ?? কেবল ডকস্ট্রিংটি দেখায়।

যদি ডকস্ট্রিংটি উইন্ডোতে ফিট করে, তাহলে IPython পরবর্তী In [] প্রম্পটটি প্রদর্শন করে। যদি একটি ডকস্ট্রিং খুব বেশি লম্বা হয়, তাহলে IPython উইন্ডোর নীচে একটি কোলন (:) প্রদর্শন করে আরও কিছু আছে তা নির্দেশ করে—পরবর্তী স্লিপেট প্রদর্শন করতে Space কী টিপুন। আপনি যথাক্রমে আপ এবং ডাউন তীর কী দিয়ে ডকস্ট্রিংয়ের মাধ্যমে পিছনে এবং সামনে নেভিগেট করতে পারেন। IPython ডকস্ট্রিংয়ের শেষে (END) প্রদর্শন করে। যেকোনো : এ q ("quit" এর জন্য) টিপুন অথবা পরবর্তী ইন [] প্রম্পটে ফিরে যেতে (END) প্রম্পটটি টিপুন। একটি পেতে

IPython এর বৈশিষ্ট্যগুলি বুঝতে, যেকোনো In [] প্রম্পটে ? টাইপ করুন, Enter টিপুন, তারপর সাহায্য ডকুমেন্টেশনের সারসংক্ষেপ পড়ুন।

৪.৩ একাধিক প্যারামিটার সহ ফাংশন

আসুন একটি সর্বোচ্চ ফাংশন সংজ্ঞায়িত করি যা তিনটি মানের মধ্যে বৃহত্তমটি নির্ধারণ করে এবং প্রদান করে — পরবর্তী সেশনটি যথাক্রমে পূর্ণসংখ্যা, ফ্লোটিংপয়েন্ট সংখ্যা এবং স্ট্রিং সহ ফাংশনটিকে তিনবার কল করে।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: def maximum(value1, value2, value3):
...:     """সর্বোচ্চ তিনটি মান প্রদান করো!"""
...:     সর্বোচ্চ_মান = মান১ যদি মান২
...:     হয় > সর্বোচ্চ_মান: সর্বোচ্চ_মান = মান২ যদি
...:             মান৩ হয় > সর্বোচ্চ_মান:
...:             সর্বোচ্চ_মান = মান৩ সর্বোচ্চ_মান ফেরত
...:             দেয়
...:
...:
```

[2] তে: সর্বোচ্চ (12, 27, 36)

আউট[2]: 36

[3] তে: সর্বোচ্চ (12.3, 45.6, 9.7)

আউট[3]: 45.6

[4] তে: maximum('হলুদ', 'লাল', 'কমলা')

আউট[4]: 'হলুদ'

আমরা if স্টেটমেন্টের উপরে এবং নীচে ফাঁকা লাইন রাখিনি, কারণ ইন্টারেক্শন মোডে ফাঁকা লাইনে রিটার্ন টিপলে ফাংশনের সংজ্ঞা সম্পূর্ণ হয়।

আপনি ints এবং floats এর মতো মিশ্র ধরণের সাথে maximum কল করতে পারেন:

[5] তে: সর্বোচ্চ (13.5, 3, 7)

আউট[5]: 13.5

maximum(13.5, 'hello', 7) কল করলে TypeError দেখা যায় কারণ greaterthan (>) অপারেটরের সাথে স্ট্রিং এবং সংখ্যা একে অপরের সাথে তুলনা করা যায় না।

ফাংশন ম্যাক্সিমামের সংজ্ঞা

ফাংশন maximum একটি কমা দ্বারা পৃথক করা তালিকার তিনটি প্যারামিটার নির্দিষ্ট করে। স্লিপেট [2]'s

আর্গুমেন্ট ১২, ২৭ এবং ৩৬ যথাক্রমে value1, value2 এবং value3 প্যারামিটারের জন্য নির্ধারিত।

বৃহত্তম মান নির্ধারণ করতে, আমরা একবারে একটি মান প্রক্রিয়া করি:

- প্রাথমিকভাবে, আমরা ধরে নিচ্ছি যে value1-এ সবচেয়ে বড় মান রয়েছে, তাই আমরা এটি স্থানীয় চলক max_value-এ বরাদ্দ করি। অবশ্যই, এটা সম্ভব যে value2 বা value3-এ আসল সবচেয়ে বড় মান রয়েছে, তাই আমদের এখনও max_value-এর সাথে এই প্রতিটির তুলনা করতে হবে।
- প্রথম if স্টেটমেন্টটি value2 > max_value পরীক্ষা করে, এবং যদি এই শর্তটি True হয় তবে max_value-কে value2 নির্ধারণ করে।
- দ্বিতীয় if স্টেটমেন্টটি value3 > max_value পরীক্ষা করে, এবং যদি এই শর্তটি True হয় তবে max_value-কে value3 নির্ধারণ করে।

এখন, max_value-তে সবচেয়ে বড় মান থাকে, তাই আমরা এটি ফেরত দেব। যখন control কলারে ফিরে আসে, তখন ফাংশনের স্লেকে প্যারামিটার value1, value2 এবং value3 এবং ডেরিয়েবল max_value - যা সব স্থানীয় ডেরিয়েবল - আর বিদ্যমান থাকে না।

পাইথনের অন্তর্নির্মিত সর্বোচ্চ এবং সর্বনিম্ন ফাংশন

অনেক সাধারণ কাজের জন্য, আপনার প্রয়োজনীয় ক্ষমতাগুলি পাইথনে ইতিমধ্যেই বিদ্যমান।

উদাহরণস্বরূপ, বিল্টইন ম্যাক্স এবং মিন ফাংশনগুলি তাদের দুই বা ততোধিক আর্গুমেন্টের মধ্যে যথাক্রমে বৃহত্তম এবং ক্ষুদ্রতম নির্ধারণ করতে জানে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6] তে: `max('হলুদ', 'লাল', 'কমলা', 'নীল', 'সবুজ')`

আউট[6]: 'হলুদ'

[7] তে: `ন্যূনতম(15, 9, 27, 14)`

আউট[7]: 9

এই ফাংশনগুলির প্রতিটি একটি পুনরাবৃত্তিযোগ্য আর্গুমেন্টও পেতে পারে, যেমন একটি তালিকা বা একটি স্ট্রিং।
পাইথন স্ট্যান্ডার্ড লাইব্রেরির মডিউল থেকে বিল্টইন ফাংশন অথবা ফাংশন ব্যবহার করে নিজের লেখার পরিবর্তে
ডেডেলপমেন্টের সময় কমানো যায় এবং প্রোগ্রামের নির্ভরযোগ্যতা, বহনযোগ্যতা এবং কর্মক্ষমতা বৃদ্ধি করা যায়। পাইথনের
বিল্টইন ফাংশন এবং মডিউলের তালিকার জন্য, দেখুন

৪.৪ এলোমেলো সংখ্যার প্রজন্ম

এখন আমরা একটি জনপ্রিয় ধরণের প্রোগ্রামিং অ্যাপ্লিকেশন - সিমুলেশন এবং গেম প্লেয়িং - এর একটি সংক্ষিপ্ত বিবর্তন দেখব। আপনি পাইথন স্ট্যান্ডার্ড লাইব্রেরির **র্যান্ডম মডিউলের** মাধ্যমে **সুযোগের উপাদানটি** পরিচয় করিয়ে দিতে পারেন।

ছয় পার্শ্বযুক্ত ডাই গড়িয়ে ফেলা

একটি ছয়পার্শ্বযুক্ত ডাই রোলিং সিমুলেট করার জন্য ১-৬ রেঞ্জের মধ্যে ১০টি এলোমেলো পূর্ণসংখ্যা তৈরি করা যাক:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: র্যান্ডম আমদানি করুন
[2] তে: রোল ইন রেঞ্জের জন্য (10): print(random.randrange(1,
...:           7), end=' ')
...:
8 2 5 5 8 6 8 6 1 5
```

প্রথমে, আমরা র্যান্ডম ইমপোর্ট করি যাতে আমরা মডিউলের ক্ষমতা ব্যবহার করতে পারি। **র্যান্ডরেঞ্জ ফাংশনটি** প্রথম আর্গুমেন্ট মান থেকে দ্বিতীয় আর্গুমেন্ট মান পর্যন্ত একটি পূর্ণসংখ্যা তৈরি করে, কিন্তু অন্তর্ভুক্ত করে না। for স্টেটমেন্টটি প্রত্যাহার করার জন্য উপরের তীর কী ব্যবহার করুন, তারপর এটি পুনরায় কার্যকর করার জন্য Enter টিপুন। লক্ষ্য করুন যে বিভিন্ন মান প্রদর্শিত হচ্ছে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[3] তে: রোল ইন রেঞ্জের জন্য (10):
...:           মুদ্রণ(random.randrange(1, 7), end=' ')
...:
8 5 8 5 1 8 1 8 6 5
```

কখনও কখনও, আপনি একটি র্যান্ডম সিকোয়েন্সের **পুনরুৎপাদনযোগ্যতার** গ্যারান্টি দিতে চাইতে পারেন —উদাহরণস্বরূপ, ডিবাগিংয়ের জন্য। এই বিভাগের শেষে, আমরা র্যান্ডম মডিউলের বীজ ব্যবহার করব এটি করার জন্য ফাংশন।

ছয় পার্শ্বযুক্ত ডাই ৬০,০০,০০০ বার ঘূর্ণায়মান

যদি র্যান্ডরেঞ্জ সত্যিই এলোমেলোভাবে পূর্ণসংখ্যা তৈরি করে, তাহলে এর পরিসরের প্রতিটি সংখ্যার প্রতিবার কল করার সময় ফিরে আসার **সন্তাবনা** (অথবা সন্তাবনা বা সন্তাবনা) সমান থাকে। ডাই ফেস ১-৬ সমান সন্তাবনার সাথে ঘটে তা দেখানোর জন্য, নিম্নলিখিত স্ক্রিপ্টটি 6,000,000 ডাই রোল সিমুলেট করে। যখন আপনি স্ক্রিপ্টটি চালান, তখন প্রতিটি ডাই ফেস প্রায় 1,000,000 বার হওয়া উচিত, যেমন নমুনা আউটপুটে।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

ମୁଖ	ଫିକୋଯାଲି
୧	୯୯୮୬୮୬
୨	୧୦୦୧୪୮୧
୩	୯୯୯୯୦୦
୪	୧୦୦୦୮୫୦
୫	୯୯୯୯୯୫୦
୬	୯୯୯୯୯୧୧୨

ক্রিপ্টটি নেস্টেড কন্ট্রোল স্টেটমেন্ট (for স্টেটমেন্টে নেস্টেড একটি if elif স্টেটমেন্ট) ব্যবহার করে প্রতিটি ডাইফেস কতবার প্রদর্শিত হবে তা নির্ধারণ করে। for স্টেটমেন্টটি 6,000,000 বার পুনরাবৃত্তি করে। 6000000 মানকে আরও পঠনযোগ্য করার জন্য আমরা Python এর আন্ডারস্কোর (_) ডিজিট সেপারেটর ব্যবহার করেছি। এক্সপ্রেশন range(6,000,000) ভুল হবে। ফাংশন কলে কমা আলাদা আর্গুমেন্ট দেয়, তাই Python range(6,000,000) কে 6, 0 এবং 0 এই তিনটি আর্গুমেন্ট দিয়ে range-এর কল হিসেবে বিবেচনা করবে।

প্রতিটি ডাই রোলের জন্য, ক্রিপ্টটি উপযুক্ত কাউন্টার ভ্যারিয়েবলে 1 যোগ করে। প্রোগ্রামটি চালান এবং ফলাফলগুলি পর্যবেক্ষণ করুন। এই প্রোগ্রামটি এক্সিকিউশন সম্পূর্ণ করতে কয়েক সেকেন্ড সময় নিতে পারে। আপনি দেখতে পাবেন, প্রতিটি এক্সিকিউশন ভিন্ন ফলাফল তৈরি করে। মনে রাখবেন যে আমরা if elif স্টেটমেন্টে else ক্লজ প্রদান করিনি।

পুনরুৎপাদনযোগ্যতার জন্য র্যান্ডম-নম্বর জেনারেটর তৈরি করা

ফাংশন randrange আসলে একটি অভ্যন্তরীণ গণনার উপর ভিত্তি করে ছদ্ম-র্যান্ডম সংখ্যা তৈরি করে যা একটি সংখ্যাসূচক মান দিয়ে শুরু হয় যা একটি seed নামে পরিচিত। বারবার randrange কল করলে এমন সংখ্যার একটি ক্রম তৈরি হয় যা র্যান্ডম বলে মনে হয়, কারণ প্রতিবার যখন আপনি একটি নতুন ইন্টারেক্ষিভ সেশন শুরু করেন বা র্যান্ডম মডিউলের ফাংশন ব্যবহার করে এমন একটি ক্রিপ্ট চালান, তখন Python অভ্যন্তরীণভাবে একটি ভিন্ন ক্রম ব্যবহার করে। যখন আপনি এমন প্রোগ্রামগুলিতে লজিক ক্রিটগুলি ডিবাগ করেন যা^১ এলোমেলোভাবে তৈরি ডেটা ব্যবহার করে, তখন অন্যান্য মান দিয়ে প্রোগ্রামটি পরীক্ষা করার আগে লজিক ক্রিটগুলি বাদ না দেওয়া পর্যন্ত র্যান্ডম সংখ্যার একই ক্রম ব্যবহার করা সহায়ক হতে পারে। এটি করার জন্য, আপনি র্যান্ডম নম্বর জেনারেটর নিজেই বীজ করার জন্য র্যান্ডম মডিউলের ক্রম ব্যবহার করতে পারেন — এটি randrange কে আপনার নির্দিষ্ট করা বীজ থেকে তার ছদ্ম-র্যান্ডম সংখ্যা ক্রম গণনা শুরু করতে বাধ্য করে। পরবর্তী সেশনে, স্নিপেট [5] এবং [8] একই ফলাফল তৈরি করে, কারণ স্নিপেট [4] এবং [7] একই ক্রম ব্যবহার করে (32):

^১ ডকুমেন্টেশন অনুসারে, পাইথন সিস্টেম ঘড়ি বা অপারেটিং সিস্টেম-নির্ভর র্যান্ডমনেস উৎসের উপর ভিত্তি করে বীজ মান নির্ধারণ করে। ক্রিপ্টোগ্রাফির মতো সুরক্ষিত র্যান্ডম সংখ্যার প্রয়োজন এমন অ্যালগিকেশনগুলির জন্য, ডকুমেন্টেশন র্যান্ডম মডিউলের পরিবর্তে সিক্রেটস মডিউল ব্যবহার করার পরামর্শ দেয়।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: random.seed(32)

[5] তে: রোল ইন রেঞ্জের জন্য (10): print(random.randrange(1,

...: 7), end='')

...:

১ ২ ৩ ৪ ৫ ৬ ৭ ৮ ৯ ১ ০ ১

[6] তে: রোল ইন রেঞ্জের জন্য (10):

...: মুদ্রণ(random.randrange(1,7),end='')

...:

১ ৩ ৫ ৩ ১ ৫ ৬ ৪ ৩ ৫

[7] তে: random.seed(32)

[8] তে: রোল ইন রেঞ্জের জন্য (10): print(random.randrange(1,

....: 7), end='')

....:

১ ২ ২ ৩ ৬ ২ ৪ ১ ৬ ১

স্লিপেট [6] বিভিন্ন মান তৈরি করে কারণ এটি কেবল স্লিপেট [5] এ শুরু হওয়া সিউডোর্যান্ডম সংখ্যা ক্রমটি চালিয়ে যায়।

৪.৫ কেস স্টাডি: সুযোগের খেলা

এই বিভাগে, আমরা "ক্র্যাপস" নামে পরিচিত জনপ্রিয় পাশা খেলাটি অনুকরণ করব। এখানে প্রয়োজনীয়তার বিবৃতিটি দেওয়া হল:

তুমি দুটি ছয়পার্শ্বযুক্ত পাশা গড়বে, প্রতিটিতে যথাক্রমে এক, দুই, তিন, চার, পাঁচ এবং ছয়টি দাগ থাকবে। পাশা যখন স্থির হয়, তখন দুটি উর্ধ্বমুখী পাশার ঘোগফল গণনা করা হয়। যদি প্রথম রোলে ঘোগফল ৭ বা ১১ হয়, তাহলে তুমি জিতবে। যদি প্রথম রোলে ঘোগফল ২, ৩ বা ১২ হয় (যাকে "ক্র্যাপস" বলা হয়), তাহলে তুমি হেরে যাবে (অর্থাৎ, "ঘর" জিতে যাবে)। যদি প্রথম রোলে ঘোগফল ৪, ৫, ৬, ৮, ৯ বা ১০ হয়, তাহলে সেই ঘোগফল তোমার "পয়েন্ট" হয়ে যাবে। জয়ের জন্য, তোমাকে "আপনার পয়েন্ট তৈরি" না করা পর্যন্ত (অর্থাৎ, একই পয়েন্ট মান গড়ে তোলা) পাশা গড়িয়ে চলতে হবে।

আপনার বক্তব্য তুলে ধরার আগে ৭ গড়িয়ে পড়লে আপনি হেরে যাবেন।

নিম্নলিখিত স্ক্রিপ্টটি গেমটির অনুকরণ করে এবং বেশ কয়েকটি নমুনা সম্পাদন দেখায়, যেখানে প্রথম রোলে জয়, প্রথম রোলে হেরে যাওয়া, পরবর্তী রোলে জয় এবং পরবর্তী রোলে হেরে যাওয়ার চিত্র তুলে ধরা হয়েছে।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```

১ # fig04_02.py ২ """পাশা
খেলা ক্র্যাপস অনুকরণ করা হচ্ছে!"""
৩ টি এলোমেলোভাবে আমদানি করুন
৪
৫ ডিফল্ট রোল_ডাইস(): ৬
    """দুটি ডাইস ঘূরিয়ে তাদের ফেস ভ্যালুগুলো টুপল হিসেবে ফিরিয়ে আনুন!""" die1 = random.randrange(1, 7) die2 =
  ৭     random.randrange(1, 7) রিটার্ন (die1, die2) # ডাই ফেস
  ৮     ভ্যালুগুলোকে টুপলে প্যাক করুন
  ৯
  ১০
  ১১ ডিফল্ট ডিসপ্লে_ডাইস(ডাইস):
  ১২     """দুটি ডাইসের একটি রোল প্রদর্শন করুন!""" die1, die2 = dice # টিপলটিকে die1
  ১৩     এবং die2 ভেরিয়েবলে আনপ্যাক করুন
  ১৪     মুদ্রণ (f'Player রোলড {die1} + {ডাই২}) = {ঘোগফল(ডাইস)}'")
```

১৫

১৬টি ডাই_ভ্যালু = রোল_ডাইস() # প্রথম রোল

১৭টি ডিসপ্লে_ডাইস(ডাই_ভ্যালু)

১৮

১৯ # প্রথম খেলার উপর ভিত্তি করে খেলার অবস্থা এবং পয়েন্ট নির্ধারণ করুন

২০টি ডাইসের_সমষ্টি = যোগফল(ডাই_মান)

২১

২২ যদি (১, ১) ডাইসের_সমষ্টি : # জয়

২৩ খেলার অবস্থা = 'জয়ী'

২৪টি এলিফ sum_of_dice (২, ৩, ১২): # হারানো

২৫ খেলার অবস্থা = 'হারিয়ে গেছে'

২৬টি অন্য: # মনে রাখার বিষয়

২৭ গেম_স্ট্যাটাস = 'চালিয়ে যান'

২৮ আমার_পয়েন্ট = পাশার_সমষ্টি

২৯ মুদ্রণ ('পয়েন্ট হল', আমার_পয়েন্ট)

৩০

৩১ # খেলোয়াড় জেতা বা হারানো পর্যন্ত ঘূর্ণায়মান থাকুন

৩২ while game_status == 'চালিয়ে যান':

৩৩ ডাই_ভ্যালু = রোল_ডাইস()

৩৪ ডিসপ্লে_ডাইস(ডাই_মান)

৩৫ পাশার_সমষ্টি = যোগফল(ডাই_মান)

৩৬

৩৭ যদি sum_of_dice == আমার_পয়েন্ট: # পয়েন্ট তৈরি করে জয়ী হও

৩৮ খেলার অবস্থা = 'জয়ী'

৩৯ elif sum_of_dice == 7: # 7 ঘূর্ণায়মান করে হেরে যাওয়া

৪০ খেলার_স্থিতি = 'হারিয়ে গেছে'

৪১

৪২ # "জয়" বা "পরাজয়" বার্তা প্রদর্শন করুন

৪৩ যদি গেম_স্ট্যাটাস == 'জিতেছে':

৪৪টি প্রিন্ট ('খেলোয়াড় জয়ী')

৪৫টি অন্য:

৪৬ মুদ্রণ ('খেলোয়াড় হেরে যায়')



কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

খেলোয়াড় $2 + 5 = 7$ ঘূর্ণিত

খেলোয়াড় জিতেছে

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

খেলোয়াড় $1 + 2 = 3$ ঘূর্ণিত

খেলোয়াড় হেরে যায়

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

খেলোয়াড় $5 + 8 = 9$ ঘূর্ণিত

পয়েন্ট ৯।

খেলোয়াড় $8 + 8 = 8$ ঘূর্ণিত

খেলোয়াড় $2 + 3 = 5$ ঘূর্ণিত

খেলোয়াড় $5 + 8 = 9$ ঘূর্ণিত

খেলোয়াড় জিতেছে

কোড ইমেজ দেখতে এখানে ক্লিক করুন

খেলোয়াড় $1 + 5 = 6$ ঘূর্ণিত

পয়েন্ট হল ৬

খেলোয়াড় $1 + 6 = 7$ ঘূর্ণিত

খেলোয়াড় হেবে যায়

ফাংশন roll_dice—একটি টুপলের মাধ্যমে একাধিক মান ফেরত দেওয়া

ফাংশন `roll_dice` (লাইন ৫-৯) প্রতিটি রোলে দুটি ডাইস রোল করার সিমুলেট করে। ফাংশনটি একবার সংজ্ঞায়িত করা হয়, তারপর প্রোগ্রামের বিভিন্ন স্থান থেকে কল করা হয় (লাইন ১৬ এবং ৩৩)। খালি প্যারামিটার তালিকাটি নির্দেশ করে যে `roll_dice`-এর কার্য সম্পাদনের জন্য আর্গুমেন্টের প্রয়োজন হয় না।

এর কাজ।

আপনি এখন পর্যন্ত যে বিল্টইন এবং কাস্টম ফাংশনগুলি কল করেছেন সেগুলি প্রতিটি একটি করে মান প্রদান করে।

কখনও কখনও একাধিক মান ফেরত দেওয়া কার্যকর হয়, যেমন `roll_dice`-এ, যা উভয় ডাই মান (লাইন ৯) কে একটি টুপল হিসেবে ফেরত দেয়—একটি অপরিবর্তনীয় (অর্থাৎ, অপরিবর্তনীয়) মানের ক্রম। একটি টুপল তৈরি করতে, লাইন ৯-এর মতো এর মানগুলিকে কমা দিয়ে আলাদা করুন:

(১, ২)

এটিকে টুপল প্যাকিং বলা হয়। বন্ধনীগুলি ঐচ্ছিক, তবে স্পষ্টতার জন্য আমরা সেগুলি ব্যবহার করার পরামর্শ দিচ্ছি। পরবর্তী অধ্যায়ে আমরা টুপলগুলি সম্পর্কে বিস্তারিত আলোচনা করব।

ফাংশন `display_dice`

একটি টুপলের মান ব্যবহার করার জন্য, আপনি তাদের একটি কমা দ্বারা পৃথক করা ভেরিয়েবলের তালিকায় বরাদ্দ করতে পারেন, যা টুপলটি **আনপ্যাক** করে। ডাইসের প্রতিটি রোল প্রদর্শনের জন্য, `display_dice` ফাংশন (লাইন 11-14 এ সংজ্ঞায়িত এবং লাইন 17 এবং 34 এ বলা হয়) এটি প্রাপ্ত টুপল আর্গুমেন্টটি আনপ্যাক করে (লাইন 13)। = এর বাম দিকের ভেরিয়েবলের সংখ্যাটি টুপলের উপাদানের সংখ্যার সাথে মিলতে হবে; অন্যথায়, একটি `ValueError` ঘটে। লাইন 14 একটি ফর্ম্যাটেড স্ট্রিং প্রিন্ট করে যেখানে ডাই মান এবং তাদের যোগফল উভয়ই থাকে। আমরা ডাইসের যোগফল গণনা করি

টুপলটিকে বিল্টইন সাম ফাংশনে পাস করা—একটি তালিকার মতো, একটি টুপল হল একটি ক্রম।

মনে রাখবেন যে roll_dice এবং display_dice ফাংশনগুলি তাদের ব্লকগুলি একটি ডকস্ট্রিং দিয়ে শুরু করে যা ফাংশনটি কী করে তা বর্ণনা করে। এছাড়াও, উভয় ফাংশনে স্থানীয় ভেরিয়েবল die1 এবং die2 রয়েছে। এই ভেরিয়েবলগুলি "সংঘর্ষিত" হয় না কারণ এগুলি বিভিন্ন ফাংশনের ব্লকের অন্তর্গত। প্রতিটি স্থানীয় ভেরিয়েবল কেবল সেই ব্লকেই অ্যাক্সেসযোগ্য যা এটিকে সংজ্ঞায়িত করে।

প্রথম রোল

যখন স্ক্রিপ্টটি কার্যকর করা শুরু হয়, তখন ১৬-১৭ নম্বর লাইনে ডাইস গড়িয়ে ফলাফল দেখা যায়। লাইন ২০ নম্বর লাইন ২২-২৯-এ ব্যবহারের জন্য ডাইসের যোগফল গণনা করে। আপনি প্রথম রোল বা পরবর্তী যেকোনো রোলে জিততে বা হারতে পারেন। পরিবর্তনশীল game_status জয়/পরাজয়ের হিসাব রাখে।
অবস্থা।

২২ নম্বর লাইনে [ইন](#) অপারেটর

(৭, ১১) তে পাশার_সমষ্টি

পরীক্ষা করে যে টুপল (৭, ১১) তে sum_of_dice এর মান আছে কিনা। যদি এই শর্তটি সত্য হয়, তাহলে আপনি ৭ অথবা ১১ রোল করেছেন। এই ক্ষেত্রে, আপনি প্রথম রোলে জিতেছেন, তাই স্ক্রিপ্টটি game_status কে 'WON' তে সেট করে। অপারেটরের ডান অপারেন্ড যেকোনো পুনরাবৃত্তিযোগ্য হতে পারে। একটি মান পুনরাবৃত্তিযোগ্য নয় কিনা তা নির্ধারণ করার জন্য একটি [not in](#) অপারেটরও রয়েছে। পূর্ববর্তী সংক্ষিপ্ত শর্তটি এর সমতুল্য

কোড ইমেজ দেখতে এখানে [ক্লিক করুন](#)

(ডাইসের_সমষ্টি == ৭) অথবা (ডাইসের_সমষ্টি == ১১)

একইভাবে, ২৪ নম্বর লাইনের শর্ত

(২, ৩, ১২) তে পাশার_সমষ্টি

পরীক্ষা করে যে টুপলে (2, 3, 12) sum_of_dice এর মান আছে কিনা। যদি তাই হয়, তাহলে আপনি প্রথম রোলে হেরে গেছেন, তাই স্ক্রিপ্টটি game_status কে 'LOST' তে সেট করে।

অন্য যেকোনো পাশার যোগফলের জন্য (৪, ৫, ৬, ৮, ৯ অথবা ১০):

- লাইন ২৭ গেম_স্ট্যাটাসকে 'CONTINUE' তে সেট করে যাতে আপনি রোলিং চালিয়ে যেতে পারেন।

- ২৮ নম্বর লাইনে my_point-এ ডাইসের যোগফল সংরক্ষণ করা হয়েছে যাতে আপনাকে জিততে হলে কী রোল করতে হবে তা ট্র্যাক করা যায় এবং
- লাইন ২৯ আমার_পয়েন্ট প্রদর্শন করে।

পরবর্তী রোলগুলি

যদি game_status 'CONTINUE' (লাইন 32) এর সমান হয়, তাহলে আপনি জিতেছেন বা হেরেছেন না, তাই while স্টেটমেন্টের সৃষ্টি (লাইন 33-40) কার্যকর হয়। প্রতিটি লুপ পুনরাবৃত্তি roll_dice কল করে, ডাই মান প্রদর্শন করে এবং তাদের যোগফল গণনা করে। যদি sum_of_dice my_point (লাইন 37) বা 7 (লাইন 39) এর সমান হয়, তাহলে স্ক্রিপ্টটি game_status কে যথাক্রমে 'WON' বা 'LOST' এ সেট করে এবং লুপটি বন্ধ হয়ে যায়। অন্যথায়, while লুপটি কার্যকর হতে থাকে।

পরবর্তী রোলের সাথে।

চূড়ান্ত ফলাফল প্রদর্শন করা হচ্ছে

যখন লুপটি শেষ হয়ে যায়, তখন স্ক্রিপ্টটি if else স্টেটমেন্টে (লাইন 43-46) এগিয়ে যায়, যা 'খেলোয়াড় জয়ী' প্রিন্ট করে যদি game_status 'WON' হয়, অথবা 'খেলোয়াড় হারে'। অন্যথায়।

৪.৬ পাইথন স্ট্যান্ডার্ড লাইব্রেরি

সাধারণত, আপনি পাইথন প্রোগ্রামগুলি এমন ফাংশন এবং ক্লাস (অর্থাৎ, কাস্টম টাইপ) একত্রিত করে লেখেন যা আপনি আগে থেকে বিদ্যমান ফাংশন এবং মডিউলগুলিতে সংজ্ঞায়িত ক্লাস, যেমন পাইথন স্ট্যান্ডার্ড লাইব্রেরি এবং অন্যান্য লাইব্রেরিতে, দিয়ে তৈরি করেন। একটি মূল প্রোগ্রামিং লক্ষ্য হল "চাকা পুনর্বীকরণ" এড়ানো।

মডিউল হলো এমন একটি ফাইল যা সম্পর্কিত ফাংশন, ডেটা এবং ক্লাসগুলিকে গোষ্ঠীভুক্ত করে। পাইথন স্ট্যান্ডার্ড লাইব্রেরির দশমিক মডিউল থেকে ডেসিমাল টাইপ আসলে একটি ক্লাস। আমরা chapter 1 এ সংক্ষিপ্তভাবে ক্লাসগুলি চালু করেছি এবং "ObjectOriented Programming" অধ্যায়ে সেগুলি বিস্তারিতভাবে আলোচনা করেছি। একটি [প্যাকেজ](#) সম্পর্কিত মডিউলগুলিকে গোষ্ঠীভুক্ত করে। এই বইতে, আপনি অনেক পূর্ব-বিদ্যমান মডিউল এবং প্যাকেজগুলির সাথে কাজ করবেন এবং আপনি আপনার নিজস্ব মডিউল তৈরি করবেন—আসলে, আপনার তৈরি করা প্রতিটি পাইথন সোর্সকোড (.py) ফাইলই একটি মডিউল। প্যাকেজ তৈরি করা এই বইয়ের আওতার বাইরে। এগুলি সাধারণত একটি বৃহৎ লাইব্রেরির কার্যকারিতাকে ছোট ছোট উপসেটে সংগঠিত করতে ব্যবহৃত হয় যা রক্ষণাবেক্ষণ করা সহজ এবং সুবিধার জন্য আলাদাভাবে আমদানি করা যেতে পারে। উদাহরণস্বরূপ, matplotlib ভিজুয়ালাইজেশন লাইব্রেরি যা আমরা `ecution 5.17` এ ব্যবহার করি তার ব্যাপক কার্যকারিতা রয়েছে (এর ডকুমেন্টেশন 2300 পৃষ্ঠারও বেশি), তাই আমরা আমাদের উদাহরণগুলিতে (`pyplot` এবং অ্যানিমেশন) কেবলমাত্র প্রয়োজনীয় উপসেটগুলি আমদানি করব।

পাইথন স্ট্যান্ডার্ড লাইব্রেরিতে মূল পাইথন ভাষা সরবরাহ করা হয়েছে। এর প্যাকেজগুলি

এবং মডিউলগুলিতে দৈনন্দিন প্রোগ্রামিং কাজের বিস্তৃত পরিসরের ক্ষমতা রয়েছে।

আপনি স্ট্যান্ডার্ড লাইব্রেরি মডিউলগুলির একটি সম্পূর্ণ তালিকা এখানে দেখতে পাবেন

^২ পাইথন টিউটোরিয়াল এটিকে ব্যাটারি অন্তর্ভুক্ত পদ্ধতি হিসেবে উল্লেখ করে।

<https://docs.python.org/3/library/>

আপনি ইতিমধ্যেই দশমিক, পরিসংখ্যান এবং র্যান্ডম মডিউল থেকে ক্ষমতা ব্যবহার করেছেন। পরবর্তী বিভাগে, আপনি গণিত মডিউল থেকে গণিত ক্ষমতা ব্যবহার করবেন। বইয়ের উদাহরণগুলিতে আপনি আরও অনেক পাইথন স্ট্যান্ডার্ড লাইব্রেরি মডিউল দেখতে পাবেন, যার মধ্যে নিম্নলিখিত টেবিলের অনেকগুলি রয়েছে:

কিছু জনপ্রিয় পাইথন স্ট্যান্ডার্ড লাইব্রেরি মডিউল

সংগ্রহ—তথ্য কাঠামো

তালিকা, টিপল, অভিধানের বাইরে এবং
সেট।

ক্রিপ্টোগ্রাফি মডিউল—নিরাপদ ট্রালমিশনের জন্য ডেটা
এনক্রিপ্ট করা।

CSV—কমা দ্বারা পৃথক করা মান ফাইলগুলি
প্রক্রিয়াকরণ (যেমন এক্সেলের ফাইলগুলি)।

তারিখ সময় - তারিখ এবং সময়
কারসাজি। সময়ে মডিউল করে
এবং ক্যালেন্ডার।

দশমিক—স্থিরবিন্দু এবং ভাসমান-বিন্দু পাটিগণিত, আর্থিক
গণনা সহ।

doctest—এব্বেড ভ্যালিডেশন পরীক্ষা

এবং সহজ ইউনিট পরীক্ষার জন্য ডকস্ট্রি-এ প্রত্যাশিত
ফলাফল।

গণিত—সাধারণ গণিত ফ্রেক্চুর

এবং অপারেশন।

OS—অপারেটিং এর সাথে ইন্টারঅ্যাক্ট করা
সিস্টেম।

প্রোফাইল, পিস্ট্যাট, টাইমইট—
কর্মক্ষমতা বিশ্লেষণ।

এলোমেলো—ছদ্ম-র্যান্ডম সংখ্যা।

পুনরায়—প্যাটার্ন মিলের জন্য নিয়মিত
এক্সপ্রেশন।

sqlite3—SQLite রিলেশনাল
ডাটাবেস অ্যাক্সেস।

পরিসংখ্যান—গাণিতিক

পরিসংখ্যান ফাংশন যেমন গড়, মধ্যমা, মোড এবং
প্রকরণ।

স্ট্রং—স্ট্রং প্রক্রিয়াকরণ।

sys—কমান্ডলাইন আর্গুমেন্ট

গেটেক্স্ট এবং লোকেল—
আন্তর্জাতিকীকরণ এবং স্থানীয়করণ
মডিউল।

json—ওয়েব পরিষেবা এবং NoSQL ডকুমেন্ট
ডাটাবেসের সাথে ব্যবহৃত জাভাস্ক্রিপ্ট অবজেক্ট নোটেশন
(JSON) প্রক্রিয়াকরণ।

প্রক্রিয়াজাতকরণ; স্ট্যান্ডার্ড ইনপুট,
স্ট্যান্ডার্ড আউটপুট এবং স্ট্যান্ডার্ড
ক্রটি স্ট্রিম।

tkinter—গ্রাফিক্যাল ব্যবহারকারী
ইন্টারফেস (GUI) এবং ক্যানভাস-ভিত্তিক
গ্রাফিক্যাল।

কচ্ছপ—কচ্ছপের গ্রাফিক্যাল।

ওয়েব ব্রাউজার—পাইথনে সুবিধাজনকভাবে
ওয়েব পৃষ্ঠা প্রদর্শনের জন্য
অ্যাপস।

৪.৭ গণিত মডিউল ফাংশন

গণিত মডিউল বিভিন্ন সাধারণ গাণিতিক গণনা সম্পাদনের জন্য ফাংশন সংজ্ঞায়িত করে। পূর্ববর্তী অধ্যায় থেকে মনে রাখবেন
যে নিম্নলিখিত ফর্মের একটি আমদানি বিবৃতি আপনাকে মডিউলের নাম এবং একটি বিল্ডু(.) এর মাধ্যমে একটি
মডিউলের সংজ্ঞা ব্যবহার করতে সক্ষম করে:

[1] তে: [আমদানি](#) গণিত

উদাহরণস্বরূপ, নিম্নলিখিত স্লিপেটটি গণিত মডিউলের `sqrt` ফাংশন কল করে 900 এর বর্গমূল গণনা করে, যা তার ফলাফলকে
ফ্লোট মান হিসাবে প্রদান করে:

[2] তে: `math.sqrt(900)`
আউট[2]: 30.0

একইভাবে, নিম্নলিখিত স্লিপেটটি গণিত মডিউলের `fabs` ফাংশন কল করে 10 এর পরম মান গণনা করে, যা তার ফলাফলকে
ফ্লোট মান হিসাবে প্রদান করে:

[3] তে: `math.fabs(10)`
আউট[3]: 10.0

কিছু গণিত মডিউল ফাংশন নীচে সংক্ষিপ্ত করা হল—আপনি সম্পূর্ণ তালিকাটি এখানে দেখতে পারেন

ফাংশন বর্ণনা

উদাহরণ

সিল(x)

x কে ক্ষুদ্রতম পূর্ণসংখ্যায় পূর্ণ করে, কম নয়
 x এর চেয়ে

সিল(9.2) হল

১০.০

মেঝে (x)

x কে বৃহত্তর নয় বরং বৃহত্তম পূর্ণসংখ্যায় পূর্ণ করে
 x এর চেয়ে

মেঝে (9.2) হল

৯.০

পাপ (x)

 x এর ত্রিকোণমিতিক সাইন (রেডিয়ানে x)

sin(0.0) হল 0.0

cos(x) এর সমীকরণ

 x এর ত্রিকোণমিতিক কোসাইন (রেডিয়ানে x)

cos(0.0) হল 1.0

ট্যান(x)

 x এর ত্রিকোণমিতিক ট্যানজেন্ট (রেডিয়ানে x)

tan(0.0) হল 0.0

এক্সপ্রেস (এক্স)

সূচকীয় ফাংশন e

exp(1.0) হল

২.৭১৮২৮২

exp(2.0) হল

৭.৩৮৯০৫৬

লগ(২.৭১৮২৮২)

হল ১.০

লগ(x) x এর প্রাকৃতিক লগারিদম (ভিত্তি e) লগ(৭.৩৮৯০৫৬)
হল 2.0

$\log_{10}(x)$ x এর লগারিদম (ভিত্তি 10) $\log_{10}(10.0)$ হল
১.০

$\log_{10}(x)$ x এর লগারিদম (ভিত্তি 10) $\log_{10}(100.0)$
হল 2.0

পাও(এক্স,
এবং) x কে y (x) এর ঘাতে উন্নীত করা হয়েছে
পাও (২.০, ৭.০) ১২৮.০
পাও (৯.০, .৫) ৩.০

বর্গমূল(x) x এর বর্গমূল $\sqrt{900.0}$ হল
৩০.০ $\sqrt{9.0}$ হল
৩.০

ফ্যাবস(এক্স) x—এর পরম মান সর্বদা একটি ফ্লোট প্রদান করে।
পাইথনে বিল্টইন ফাংশন abs রয়েছে, যা তার
আর্গমেন্টের উপর ভিত্তি করে একটি int বা float প্রদান
করে। $\text{fabs}(5.1)$ হল
৫.১ $\text{fabs}(5.1)$ হল
৫.১

fmod(x,) একটি ভাসমান বিল্ড সংখ্যা হিসেবে x/y এর অবশিষ্টাংশ
এবং $\text{fmod}(9.8, 8.0)$ হল ১.৮

৪.৮ আবিষ্কারের জন্য আইপাইথন ট্যাব সমাপ্তি ব্যবহার

আপনি **ট্যাবের** মাধ্যমে IPython ইন্টারেক্শন মোডে একটি মডিউলের ডকুমেন্টেশন দেখতে পারেন

সমাপ্তি—একটি আবিষ্কার বৈশিষ্ট্য যা আপনার কোডিং এবং আবিষ্কার প্রক্রিয়াগুলিকে গতিশীল করে।

একটি শনাক্তকারীর একটি অংশ টাইপ করে Tab চাপার পর, IPython শনাক্তকারীটি সম্পূর্ণ করে

আপনার জন্য অথবা আপনার টাইপ করা তথ্য দিয়ে শুরু হওয়া শনাক্তকারীদের একটি তালিকা প্রদান করে। এটি আপনার অপারেটিং সিস্টেম প্ল্যাটফর্ম এবং আপনি কী আমদানি করেছেন তার উপর ভিত্তি করে পরিবর্তিত হতে পারে তোমার আইপিথন সেশন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: আমদানি গণিত

[2] তে: ma<Tab>

মানচিত্র	%ম্যাক্রো	%%মার্কডাউন
গণিত	%জাদু	%ম্যাটপ্লটলিব
সর্বোচ্চ()	%পুরুষ	

আপনি উপরের এবং নীচের তীরচিহ্নগুলি ব্যবহার করে শনাক্তকারীগুলির মধ্যে স্ক্রোল করতে পারেন। যেমন আপনি করবেন, IPython একটি শনাক্তকারীকে হাইলাইট করে এবং In [] প্রস্পেক্টের ডানদিকে এটি দেখায়।

একটি মডিউলে শনাক্তকারী দেখা

একটি মডিউলে সংজ্ঞায়িত শনাক্তকারীর তালিকা দেখতে, মডিউলের নাম এবং একটি বিন্দু(.) টাইপ করুন, তারপর ট্যাব টিপুন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3] তে: গণিত।<ট্যাব>

অ্যাকোস()	() এর জন্য	কপিসাইন() এবং	এক্সপিএম১()
অ্যাকোশ()	atan2() সম্পর্কে	কোস()	উভরাখিকারসূরে()
লবণ()	আতানহ()	কোশ()	ইআরএফসি()
অসিনহ()	সিল()	ডিপ্রিং() এক্সপ্রেস()	ফ্যাক্টরিয়াল() >

যদি বর্তমানে প্রদর্শিত শনাক্তকারীর চেয়ে বেশি শনাক্তকারী থাকে, তাহলে IPython > প্রদর্শন করে।

প্রতীক (কিছু প্ল্যাটফর্মে) ডান প্রান্তে, এই ক্ষেত্রে factorial() এর ডানদিকে।

তালিকাটি স্ক্রোল করার জন্য আপনি উপরে এবং নীচের তীরচিহ্নগুলি ব্যবহার করতে পারেন। তালিকায় শনাক্তকারী:

- বন্ধনীর পরে যেগুলো আছে সেগুলো হলো ফাংশন (অথবা পদ্ধতি, যেমনটা আপনি পরে দেখবেন)।
- একক শব্দ শনাক্তকারী (যেমন কর্মচারী) যা বড় হাতের অক্ষর দিয়ে শুরু হয় এবং

বহু-শব্দ শনাক্তকারী যেখানে প্রতিটি শব্দ একটি বড় হাতের অক্ষর দিয়ে শুরু হয় (যেমন CommissionEmployee) শ্রেণীর নাম উপস্থাপন করে (পূর্ববর্তী তালিকায় কোনও নাম নেই)। পাইথন কোডের জন্য স্টাইল গাইড যা সুপারিশ করে, এই নামকরণ রীতিটি **ক্যামেলকেস** নামে পরিচিত কারণ বড় হাতের অক্ষরগুলি উটের কুঁজের মতো আলাদাভাবে দেখা যায়।

- বন্ধনীবিহীন ছোট হাতের শনাক্তকারী, যেমন pi (পূর্ববর্তী তালিকায় দেখানো হয়নি) এবং e, চলক। শনাক্তকারী pi 3.141592653589793 এ মূল্যায়ন করা হয়, এবং শনাক্তকারী e 2.718281828459045 এ মূল্যায়ন করা হয়।
গণিত মডিউলে, pi এবং e যথাক্রমে গাণিতিক ধ্রবক এবং এ প্রতিনিধিত্ব করে।

পাইথনের কোন ধ্রবক নেই, যদিও পাইথনের অনেক বস্তু অপরিবর্তনীয় (অপরিবর্তনীয়)। তাই যদিও পাই এবং ই বাস্তব জগতের ধ্রবক, তবুও আপনার তাদের নতুন মান নির্ধারণ করা উচিত নয়, কারণ এটি তাদের মান পরিবর্তন করবে। অন্যান্য ভেরিয়েবল থেকে ধ্রবককে আলাদা করতে সাহায্য করার জন্য, স্টাইল গাইড আপনার কাস্টম ধ্রবকগুলিকে সমস্ত বড় অক্ষর দিয়ে নামকরণ করার পরামর্শ দেয়।

বর্তমানে হাইলাইট করা ফাংশন ব্যবহার করা

শনাক্তকারীর মাধ্যমে নেভিগেট করার সময়, যদি আপনি বর্তমানে হাইলাইট করা একটি ফাংশন ব্যবহার করতে চান, তাহলে কেবল বন্ধনীতে এর আর্গুমেন্ট টাইপ করা শুরু করুন। IPython তারপর স্বয়ংক্রিয় সমাপ্তির তালিকা লুকিয়ে রাখে। যদি আপনার বর্তমানে হাইলাইট করা আইটেম সম্পর্কে আরও তথ্যের প্রয়োজন হয়, তাহলে আপনি নামের পরে একটি প্রশ্ন চিহ্ন (?) টাইপ করে এবং সহায়তা ডকুমেন্টেশন দেখতে Enter টিপে এর ডকস্ট্রিং দেখতে পারেন। নিম্নলিখিতটি fabs ফাংশনের ডকস্ট্রিং দেখায়:

কোড ইমেজ দেখতে এখানে [ক্লিক করুন](#)

```
[4] তে: mathfabs?
ডকস্ট্রিং:
ফ্যাবস(এক্স)

float x এর পরম মান প্রদান করো।
প্রকার:      বিল্টইন_ফাংশন_অথবা_পদ্ধতি
```

উপরে দেখানো `builtin_function_or_method` নির্দেশ করে যে fabs একটি Python Standard Library মডিউলের অংশ। এই ধরনের মডিউলগুলিকে Python-এ বিল্ট-ইন বলে মনে করা হয়।
এই ক্ষেত্রে, fabs হল গণিত মডিউলের একটি অন্তর্নির্মিত ফাংশন।

৪.৯ ডিফল্ট প্যারামিটার মান

একটি ফাংশন সংজ্ঞায়িত করার সময়, আপনি নির্দিষ্ট করতে পারেন যে একটি প্যারামিটারের একটি ডিফল্ট প্যারামিটার মান আছে। ফাংশন কল করার সময়, যদি আপনি একটি ডিফল্ট প্যারামিটার মান সহ একটি প্যারামিটারের জন্য আর্গুমেন্ট বাদ দেন, তাহলে সেই প্যারামিটারের ডিফল্ট মান স্বয়ংক্রিয়ভাবে পাস হয়ে যায়। ডিফল্ট প্যারামিটার মান সহ একটি ফাংশন rectangle_area সংজ্ঞায়িত করা যাক:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: def rectangle_area(length=2, width=3): """একটি আয়তক্ষেত্রের ক্ষেত্রফল ফেরত দিন!"""
....:     দৈর্ঘ্য * প্রস্থ ফেরত দিন
....:
....:
```

আপনি একটি প্যারামিটারের নাম অনুসরণ করে একটি = এবং একটি মান ব্যবহার করে একটি ডিফল্ট প্যারামিটার মান নির্দিষ্ট করতে পারেন—এই ক্ষেত্রে, দৈর্ঘ্য এবং প্রস্থের জন্য ডিফল্ট প্যারামিটার মান যথাক্রমে 2 এবং 3। ডিফল্ট প্যারামিটার মান সহ যেকোনো প্যারামিটার অবশ্যই প্যারামিটারের ডানদিকে প্যারামিটার তালিকায় উপস্থিত হতে হবে যার ডিফল্ট নেই।

rectangle_area-তে নিম্নলিখিত কলটিতে কোনও আর্গুমেন্ট নেই, তাই IPython উভয় ডিফল্ট প্যারামিটার মান ব্যবহার করে যেমন আপনি rectangle_area(2, 3) কল করেছেন:

```
[2] তে: আয়তক্ষেত্র_ক্ষেত্র()
আউট[2]: 6
```

rectangle_area-তে নিম্নলিখিত কলটিতে কেবল একটি আর্গুমেন্ট রয়েছে। আর্গুমেন্টগুলি বাম থেকে ডানে প্যারামিটারগুলিতে বরাদ্দ করা হয়, তাই 10 দৈর্ঘ্য হিসাবে ব্যবহৃত হয়। ইন্টারপ্রেটারটি প্রস্থের জন্য ডিফল্ট প্যারামিটার মান 3 পাস করে যেমন আপনি কল করেছেন
আয়তক্ষেত্র_ক্ষেত্র (10, 3):

```
[3] তে: rectangle_area(10)
আউট[3]: 30
```

rectangle_area-তে নিম্নলিখিত কলটিতে দৈর্ঘ্য এবং প্রস্থ উভয়ের জন্যই আর্গুমেন্ট রয়েছে, তাই IPython ডিফল্ট প্যারামিটার মান উপেক্ষা করে:

```
[4] তে: rectangle_area(10, 5)
আউট[4]: 50
```

8.10 কীওয়ার্ড আর্গুমেন্ট

ফাংশন কল করার সময়, আপনি যেকোনো ক্রমে আর্গুমেন্ট পাস করার জন্য **কীওয়ার্ড আর্গুমেন্ট** ব্যবহার করতে পারেন। কীওয়ার্ড আর্গুমেন্ট প্রদর্শনের জন্য, আমরা rectangle_area ফাংশনটি পুনরায় সংজ্ঞায়িত করি—এবার ডিফল্ট প্যারামিটার মান ছাড়াই:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[1] তে: def rectangle_area(length, width):
...:     """একটি আয়তক্ষেত্রের ক্ষেত্রফল ফেরত দিন।"""
...:     দৈর্ঘ্য * প্রস্থ
...:     ফেরত দিন
...:
```

একটি কলে প্রতিটি কীওয়ার্ড আর্গুমেন্টের ফর্ম parametername=value থাকে। নিম্নলিখিত কলটি দেখায় যে কীওয়ার্ড আর্গুমেন্টের ক্রম কোন ব্যাপার না - ফাংশন সংজ্ঞায় সংশ্লিষ্ট প্যারামিটারের অবস্থানের সাথে তাদের মিল করার প্রয়োজন নেই:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[2] তে: আয়তক্ষেত্র_ক্ষেত্র (প্রস্থ=5, দৈর্ঘ্য=10)
আউট[3]: ৫০
```

প্রতিটি ফাংশন কলে, আপনাকে ফাংশনের অবস্থানগত আর্গুমেন্টের পরে কীওয়ার্ড আর্গুমেন্ট স্থাপন করতে হবে—অর্থাৎ, যে কোনও আর্গুমেন্টের জন্য আপনি প্যারামিটারের নাম নির্দিষ্ট করেননি।

এই ধরনের আর্গুমেন্টগুলি ফাংশনের প্যারামিটার বাম থেকে ডানে বরাদ্দ করা হয়, আর্গুমেন্ট তালিকার আর্গুমেন্টের অবস্থানের উপর ভিত্তি করে। কীওয়ার্ড আর্গুমেন্টগুলি ফাংশন কলের পঠনযোগ্যতা উন্নত করার জন্যও সহায়ক, বিশেষ করে অনেকগুলি ফাংশনের জন্য যুক্তি।

৪.১১ সালিশী যুক্তি তালিকা

যেসব ফাংশনে **আর্গুমেন্ট তালিকা থাকে**, যেমন বিল্টইন ফাংশন min এবং max, সেগুলো যেকোনো সংখ্যক আর্গুমেন্ট গ্রহণ করতে পারে। নিচের min কলটি বিবেচনা করুন:

```
সর্বনিম্ন (৮৮, ৭৫, ৯৬, ৫৫, ৮৩)
```

ফাংশনের ডকুমেন্টেশনে বলা হয়েছে যে min এর দুটি প্রয়োজনীয় প্যারামিটার (arg1 এবং arg2 নামকরণ করা হয়েছে) এবং *args ফর্মের একটি ঐচ্ছিক তৃতীয় প্যারামিটার রয়েছে, যা নির্দেশ করে যে ফাংশনটি যেকোনো সংখ্যক অতিরিক্ত আর্গুমেন্ট গ্রহণ করতে পারে। প্যারামিটার নামের আগে * পাইথনকে অবশিষ্ট যেকোনো আর্গুমেন্টকে একটি টিপলে প্যাক করতে বলে যা args প্যারামিটারে পাস করা হয়। উপরের কলে, প্যারামিটার arg1 88 পায়,

প্যারামিটার arg2 75 পায় এবং প্যারামিটার args tuple (96, 55, 83) পায়।

একটি নির্বিচারে যুক্তি তালিকা ব্যবহার করে একটি ফাংশন সংজ্ঞায়িত করা

চলুন এমন একটি গড় ফাংশন সংজ্ঞায়িত করি যা যেকোনো সংখ্যক আর্গুমেন্ট গ্রহণ করতে পারে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: def average(*args): sum(args) / len(args)
...:           রিটার্ন করুন
...:
```

প্যারামিটার নাম args প্রচলিতভাবে ব্যবহৃত হয়, তবে আপনি যেকোনো শনাক্তকারী ব্যবহার করতে পারেন। যদি ফাংশনটিতে একাধিক প্যারামিটার থাকে, তাহলে *args প্যারামিটারটি অবশ্যই সবচেয়ে ডানদিকে থাকা উচিত। প্যারামিটার।

এখন, বিভিন্ন দৈর্ঘ্যের ইচ্ছামত আর্গুমেন্ট তালিকা ব্যবহার করে গড়কে বেশ কয়েকবার কল করা যাক:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[2] তে: গড় (5, 10)
আউট[2]: ৭.৫
```

```
[3] তে: গড় (5, 10, 15)
আউট[3]: ১০.০
```

```
[4] তে: গড় (5, 10, 15, 20)
আউট[4]: ১২.৫
```

গড় গণনা করার জন্য, args tuple এর উপাদানগুলির যোগফল (builtin function sum দ্বারা ফেরত) কে tuple এর উপাদানগুলির সংখ্যা (builtin function len দ্বারা ফেরত) দিয়ে ভাগ করুন। আমাদের গড় সংজ্ঞায় লক্ষ্য করুন যে যদি args এর দৈর্ঘ্য 0 হয়, তাহলে একটি ZeroDivisionError ঘটে। পরবর্তী অধ্যায়ে, আপনি দেখতে পাবেন কিভাবে একটি tuple এর উপাদানগুলিকে আনপ্যাক না করে অ্যাক্সেস করতে হয়।

একটি Iterable এর পৃথক উপাদানগুলিকে ফাংশন আর্গুমেন্ট হিসেবে পাস করা

আপনি একটি টুপল, তালিকা বা অন্যান্য পুনরাবৃত্ত উপাদানগুলিকে পৃথক ফাংশন আর্গুমেন্ট হিসাবে পাস করার জন্য আনপ্যাক করতে পারেন। * অপারেটর, যখন একটি ফাংশন কলে একটি পুনরাবৃত্ত যুক্তিতে প্রয়োগ করা হয়, তখন এর উপাদানগুলিকে আনপ্যাক করে। নিম্নলিখিত কোডটি একটি পাঁচটি উপাদানের গ্রেড তালিকা তৈরি করে, তারপর *গ্রেড এক্সপ্রেশন ব্যবহার করে এর উপাদানগুলিকে গড় হিসাবে আনপ্যাক করে।

যুক্তি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[5] তে: গ্রেড = [88, 75, 96, 55, 83]

[6] তে: গড় (*গ্রেড)

আউট[6]: 79.4

উপরে দেখানো কলাটি গড় (88, 75, 96, 55, 83) এর সমতুল্য।

৪.১২ পদ্ধতি: বস্তুর সাথে সম্পর্কিত ফাংশন

একটি **পদ্ধতি** হল কেবল একটি ফাংশন যা আপনি ফর্ম ব্যবহার করে একটি বস্তুতে কল করেন

বস্তু_নাম.পদ্ধতি_নাম(যুক্তি)

উদাহরণস্বরূপ, পরবর্তী সেশনটি স্ট্রিং ভেরিয়েবল s তৈরি করে এবং এটিকে স্ট্রিং অবজেক্ট 'Hello' বরাদ্দ করে। তারপর সেশনটি অবজেক্টের **লোয়ার** এবং **আপার** মেথডগুলিকে কল করে, যা মূল স্ট্রিংয়ের alllowercase এবং alluppercase সংস্করণ ধারণকারী নতুন স্ট্রিং তৈরি করে, s অপরিবর্তিত রেখে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: s = 'হ্যালো'

[2]: s.lower() # স্ট্রিং অবজেক্টে lower মেথড কল করুন s Out[2]: 'hello'

[3] তে: s.upper()

আউট[3]: 'হ্যালো'

[4] তে: s

আউট[4]: 'হ্যালো'

পাইথন স্ট্যান্ডার্ড লাইব্রেরির রেফারেন্স

<https://docs.python.org/3/library/index.html>

পাইথন স্ট্যান্ডার্ড লাইব্রেরিতে বিল্টইন প্রকারের পদ্ধতি এবং প্রকারগুলি বর্ণনা করে।

"অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং" অধ্যায়ে, আপনি ক্লাস নামক কাস্টম টাইপ তৈরি করবেন এবং সেই ক্লাসের অবজেক্টগুলিতে কল করার জন্য কাস্টম পদ্ধতিগুলি সংজ্ঞায়িত করবেন।

৪.১৩ স্কোপ নিয়ম

প্রতিটি শনাক্তকারীর একটি **সুযোগ** থাকে যা নির্ধারণ করে যে আপনি আপনার প্রোগ্রামে এটি কোথায় ব্যবহার করতে পারবেন। প্রোগ্রামের সেই অংশের জন্য, শনাক্তকারীকে "in scope" বলা হয়।

স্থানীয় ব্যাপ্তি

একটি স্থানীয় চলকের শনাক্তকারীর **স্থানীয় স্কোপ থাকে**। এটি কেবল তার সংজ্ঞা থেকে ফাংশনের ইন্টার্ন শেষ পর্যন্ত "in scope" থাকে। ফাংশনটি যখন তার কলারে ফিরে আসে তখন এটি "scope" এর বাইরে চলে যায়। সুতরাং, একটি স্থানীয় চলক শুধুমাত্র সেই ফাংশনের ভিতরে ব্যবহার করা যেতে পারে যা এটিকে সংজ্ঞায়িত করে।

বিশ্বব্যাপী সুযোগ

যেকোনো ফাংশন (বা ক্লাস) এর বাইরে সংজ্ঞায়িত আইডেন্টিফায়ারগুলির **গ্লোবাল স্কোপ থাকে** - এর মধ্যে ফাংশন, ভেরিয়েবল এবং ক্লাস অন্তর্ভুক্ত থাকতে পারে। গ্লোবাল স্কোপযুক্ত ভেরিয়েবলগুলিকে **গ্লোবাল ভেরিয়েবল** বলা হয়। গ্লোবাল স্কোপযুক্ত আইডেন্টিফায়ারগুলি সংজ্ঞায়িত হওয়ার পরে যেকোনো জায়গায় .py ফাইল বা ইন্টারেক্শন সেশনে ব্যবহার করা যেতে পারে।

একটি ফাংশন থেকে একটি গ্লোবাল ভেরিয়েবল অ্যাক্সেস করা

আপনি একটি ফাংশনের ভিতরে একটি গ্লোবাল ভ্যারিয়েবলের মান অ্যাক্সেস করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: x = 7
```

```
[2] তে: def access_global():
...:     মুদ্রণ (' এক্স_গ্লোবাল থেকে মুদ্রিত:', x)
...:
```

```
[3] তে: access_global() x access_global
থেকে মুদ্রিত: 7
```

তবে, ডিফল্টরূপে, আপনি কোনও ফাংশনে কোনও গ্লোবাল ভেরিয়েবল পরিবর্তন করতে পারবেন না - যখন আপনি প্রথমে কোনও ফাংশনের ইন্টার্ন কোনও ভেরিয়েবলের জন্য একটি মান নির্ধারণ করেন, তখন পাইথন একটি নতুন স্থানীয় ভেরিয়েবল তৈরি করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[4] তে: def try_to_modify_global():
...:     x = 3.5
...:     মুদ্রণ (' try_to_modify_global থেকে x মুদ্রিত:', x)
...:
```

```
[5] তে: try_to_modify_global() x try_to_modify_global
থেকে মুদ্রিত: 3.5
```

[6] তে: x

আউট[6]: 7

ফাংশন `try_to_modify_global` এর ব্লকে, স্থানীয় x প্লোবাল x কে **ছায়া দেয়**, যার ফলে এটি ফাংশনের ব্লকের স্কোপে অ্যাক্সেসযোগ্য হয় না। স্লিপেট [6] দেখায় যে প্লোবাল ভ্যারিয়েবল x এখনও বিদ্যমান এবং ফাংশনের পরে এর মূল মান (7) রয়েছে।

বিশ্বব্যাপী এক্সিকিউটগুলিকে `_পরিবর্তন_করার_চেষ্টা` করুন।

একটি ফাংশনের ব্লকে একটি প্লোবাল ভ্যারিয়েবল পরিবর্তন করতে, আপনাকে একটি **প্লোবাল স্টেটমেন্ট** ব্যবহার করে ঘোষণা করতে হবে যে ভ্যারিয়েবলটি প্লোবাল স্কোপে সংজ্ঞায়িত করা হয়েছে:

`কোড ইমেজ দেখতে এখানে ক্লিক করুন`

```
[7] তে: def modify_global(): global x
...
...
x = 'হ্যালো'
...
    মুদ্রণ ('x modify_global থেকে মুদ্রিত:', x)
...
```

```
[8] তে: modify_global() x modify_global
থেকে মুদ্রিত: hello
```

```
[9] তে: x
আউট[9]: 'হ্যালো'
```

ব্লক বনাম স্যুট

আপনি এখন ফাংশন ব্লক এবং কন্ট্রোল স্টেটমেন্ট স্যুট সংজ্ঞায়িত করেছেন। যখন আপনি একটি ব্লকে একটি ভেরিয়েবল তৈরি করেন, তখন এটি সেই ব্লকের স্থানীয় হয়। যাইহোক, যখন আপনি একটি কন্ট্রোল স্টেটমেন্টের স্যুটে একটি ভেরিয়েবল তৈরি করেন, তখন ভেরিয়েবলের স্কোপ নিয়ন্ত্রণ স্টেটমেন্টটি কোথায় অবস্থিত তার উপর নির্ভর করে।

সংজ্ঞায়িত করা হয়েছে:

- যদি কন্ট্রোল স্টেটমেন্টটি প্লোবাল স্কোপে থাকে, তাহলে কন্ট্রোল স্টেটমেন্টে সংজ্ঞায়িত যেকোনো ভেরিয়েবলের প্লোবাল স্কোপে থাকবে।
- যদি কন্ট্রোল স্টেটমেন্টটি একটি ফাংশনের ব্লকে থাকে, তাহলে কন্ট্রোল স্টেটমেন্টে সংজ্ঞায়িত যেকোনো ভেরিয়েবলের স্থানীয় সুযোগ থাকবে।

আমরা যখন কাস্টম ক্লাস চালু করব, তখন "অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং" অধ্যায়ে আমাদের স্কোপ আলোচনা চালিয়ে যাব।

ছায়াকরণ ফাংশন

পূর্ববর্তী অধ্যায়গুলিতে, মানগুলি সংকলন করার সময়, আমরা sum নামক একটি চলকটিতে sum সংরক্ষণ করেছি। আমরা এটি করার কারণ হল sum হল একটি বিল্টইন ফাংশন। আপনি যদি sum নামক একটি চলক সংজ্ঞায়িত করেন, তবে এটি বিল্টইন ফাংশনকে ছায়া দেয়, যা আপনার কোডে এটি অ্যাক্সেসযোগ্য করে তোলে। যখন আপনি নিম্নলিখিত অ্যাসাইনমেন্টটি কার্যকর করেন, তখন পাইথন 15 ধারণকারী int অবজেক্টের সাথে identifier sum কে আবদ্ধ করে। এই মুহূর্তে, identifier sum আর builtin ফাংশনকে উল্লেখ করে না। সুতরাং, যখন আপনি sum কে একটি ফাংশন হিসাবে ব্যবহার করার চেষ্টা করেন, তখন একটি TypeError ঘটে:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[10] তে: যোগফল = 10 + 5
[11] তে: যোগফল
আউট[11]: 15
[12] তে: sum([10, 5])

টাইপ এরর
<module>() এ ipythoninpu121237d97a65fb>
> ১ যোগফল([১০, ৫])

TypeError: 'int' অবজেক্টটি কলযোগ্য নয়
```

গ্লোবাল স্কোপে বিবৃতি

এখন পর্যন্ত আপনি যে স্ক্রিপ্টগুলি দেখেছেন, তাতে আমরা গ্লোবাল স্কোপে ফাংশনের বাইরে কিছু স্টেটমেন্ট এবং ফাংশন ব্লকের ভিতরে কিছু স্টেটমেন্ট লিখেছি। গ্লোবাল স্কোপে স্ক্রিপ্ট স্টেটমেন্টগুলি ইন্টারপ্রেটার দ্বারা দেখা মাত্রাই কার্যকর হয়, যেখানে ব্লকের স্টেটমেন্টগুলি কেবল তখনই কার্যকর হয় যখন ফাংশনটি কল করা হয়।

৪.১৪ আমদানি: আরও গভীর চেহারা

আপনি মডিউলগুলি (যেমন math এবং random) আমদানি করেছেন যার একটি বিবৃতি রয়েছে:

মডিউল_নাম আমদানি করুন

তারপর প্রতিটি মডিউলের নাম এবং একটি বিল্ড(.) এর মাধ্যমে তাদের বৈশিষ্ট্যগুলি অ্যাক্সেস করা হয়েছে। এছাড়াও, আপনি একটি মডিউল থেকে একটি নির্দিষ্ট শনাক্তকারী আমদানি করেছেন (যেমন দশমিক মডিউলের দশমিক ধরণ) যার একটি বিবৃতি রয়েছে:

মডিউল_নাম আমদানি শনাক্তকারী থেকে

তারপর মডিউলের নাম এবং একটি বিল্ড(.) না দিয়েই সেই শনাক্তকারী ব্যবহার করেছি।

একটি মডিউল থেকে একাধিক শনাক্তকারী আমদানি করা

from import স্টেটমেন্ট ব্যবহার করে আপনি একটি মডিউল থেকে কমা দ্বারা পৃথক করা শনাক্তকারীর তালিকা আমদানি করতে পারেন এবং তারপর আপনার কোডে মডিউলের নাম এবং একটি বিল্ড(.) না দিয়ে ব্যবহার করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: গণিত আমদানি সিল থেকে , মেঝে

[2] তে: ceil(10.3)

আউট[2]: ১১

[3] তে: মেঝে (10.7)

আউট[3]: ১০

আমদানি করা হয়নি এমন একটি ফাংশন ব্যবহার করার চেষ্টা করলে একটি NameError দেখা দেয়, যা নির্দেশ করে যে নাম সংজ্ঞায়িত করা হয়নি।

সর্বক্ষণ: ওয়াইল্ডকার্ড আমদানি এড়িয়ে চলুন

আপনি ফর্মের ওয়াইল্ডকার্ড আমদানির মাধ্যমে একটি মডিউলে সংজ্ঞায়িত সমস্ত শনাক্তকারী আমদানি করতে পারেন।

মডিউলনাম আমদানি থেকে *

এর ফলে আপনার কোডে ব্যবহারের জন্য মডিউলের সমস্ত শনাক্তকারী উপলব্ধ হবে। ওয়াইল্ডকার্ড ইস্পোর্টের মাধ্যমে মডিউলের শনাক্তকারী আমদানি করলে সূক্ষ্ম ত্রুটি হতে পারে—এটি একটি বিপজ্জনক অভ্যাস হিসেবে বিবেচিত যা আপনার এড়ানো উচিত। নিম্নলিখিত স্লিপেটগুলি বিবেচনা করুন:

[4] তে: e = 'হ্যালো'

[5] তে: গণিত আমদানি থেকে *

[6] তে: এবং

আউট[6]: 2.718281828459045

প্রাথমিকভাবে, আমরা e নামক একটি ভেরিয়েবলের জন্য 'hello' স্ট্রিংটি বরাদ্দ করি। যদিও স্লিপেট [5] কার্যকর করার পরে, ভেরিয়েবল e, সম্ভবত দুর্ঘটনাক্রমে, গণিত মডিউলের ধ্রবক e দিয়ে প্রতিস্থাপিত হয়, যা গাণিতিক ফ্লোটিংপয়েন্ট মান e প্রতিনিধিত্ব করে।

মডিউল এবং মডিউল শনাক্তকারীর জন্য বাইন্ডিং নাম

কখনও কখনও একটি মডিউল আমদানি করা এবং আপনার কোড সহজ করার জন্য এর সংক্ষিপ্ত রূপ ব্যবহার করা সহায়ক। import statement এর as clause আপনাকে মডিউলের শনাক্তকারীর উল্লেখ করার জন্য ব্যবহৃত নাম নির্দিষ্ট করতে দেয়। উদাহরণস্বরূপ, 3.14 অনুচ্ছেদে আমরা আমদানি করতে পারতাম পরিসংখ্যান মডিউলটি ব্যবহার করেছি এবং এর গড় ফাংশনটি নিম্নরূপ অ্যাক্সেস করেছি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[7] তে: পরিসংখ্যান পরিসংখ্যান হিসেবে আমদানি করুন

[8] তে: গ্রেড = [85, 93, 45, 87, 93]

[9] তে: stats.mean(grades)

আউট[9]: 80.6

পরবর্তী অধ্যায়গুলিতে আপনি দেখতে পাবেন, import as প্রায়শই Python লাইব্রেরি আমদানি করতে ব্যবহৃত হয়, যার সংক্ষিপ্ত রূপগুলি সুবিধাজনক, যেমন পরিসংখ্যান মডিউলের জন্য stats। অন্য একটি হিসাবে উদাহরণস্বরূপ, আমরা numpy মডিউল ব্যবহার করব যা সাধারণত আমদানি করা হয়

np হিসেবে numpy আমদানি করুন

লাইব্রেরির ডকুমেন্টেশনে প্রায়শই জনপ্রিয় শর্টহ্যান্ড নাম উল্লেখ করা হয়।

সাধারণত, একটি মডিউল আমদানি করার সময়, আপনার import অথবা import as স্টেটমেন্ট ব্যবহার করা উচিত, তারপর যথাক্রমে মডিউলের নাম অথবা as কীওয়ার্ড অনুসরণ করে সংক্ষেপণ ব্যবহার করে মডিউলটি অ্যাক্সেস করা উচিত। এটি নিশ্চিত করে যে আপনি ভুলবশত এমন কোনও শনাক্তকারী আমদানি করবেন না যা আপনার কোডের একটির সাথে সাংঘর্ষিক।

৪.১৫ ফাংশনগুলিতে ঘুঙ্গুলি পাস করা: একটি গভীর দৃষ্টিভঙ্গি

আসুন আমরা আরও বিস্তারিতভাবে দেখে নিই কিভাবে ফাংশনে আর্গুমেন্ট পাস করা হয়। অনেক প্রোগ্রামিং ল্যাঙ্গুয়েজে, আর্গুমেন্ট পাস করার দুটি উপায় রয়েছে - **passbyvalue** এবং **passbyreference** (কখনও কখনও যথাক্রমে **callbyvalue** এবং **callbyreference** বলা হয়):

- **passbyvalue** এর মাধ্যমে, কল করা ফাংশনটি আর্গুমেন্টের মানের একটি কপি গ্রহণ করে এবং সেই কপির সাথে একচেটিয়াভাবে কাজ করে। ফাংশনের কপিতে পরিবর্তন কলারের মূল ভেরিয়েবলের মানকে প্রভাবিত করে না।

- passbyreference এর সাহায্যে, কল করা ফাংশনটি সরাসরি কলারের আর্গুমেন্টের মান অ্যাক্সেস করতে পারে এবং যদি মানটি পরিবর্তনযোগ্য হয় তবে তা পরিবর্তন করতে পারে।

পাইথন আর্গুমেন্টগুলি সর্বদা রেফারেন্স দ্বারা পাস করা হয়। কেউ কেউ এটিকে **পাসবাই-অবজেক্টেরেফারেন্স** বলে, কারণ "পাইথনের সবকিছুই একটি অবজেক্ট।" যখন একটি ফাংশন কল একটি আর্গুমেন্ট প্রদান করে, তখন পাইথন আর্গুমেন্ট অবজেক্টের রেফারেন্সটি অনুলিপি করে - বস্তুটি নিজেই নয় - সংশ্লিষ্ট প্যারামিটার। এটি কর্মক্ষমতার জন্য গুরুত্বপূর্ণ। ফাংশনগুলি প্রায়শই বড় বস্তুগুলিকে ম্যানিপুলেট করে - ঘন ঘন সেগুলি অনুলিপি করলে প্রচুর পরিমাণে কম্পিউটার মেমোরি খরচ হয় এবং প্রোগ্রামের কর্মক্ষমতা উল্লেখযোগ্যভাবে ধীর হয়ে যায়।

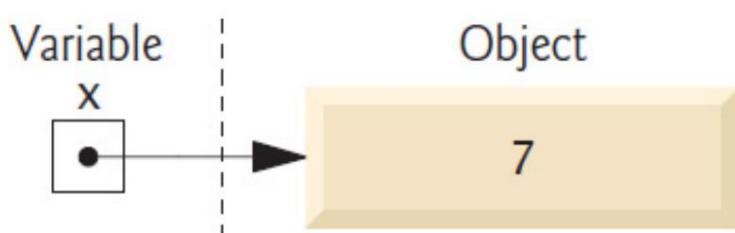
^৩ এমনকি এই অধ্যায়ে আপনি যে ফাংশনগুলি সংজ্ঞায়িত করেছেন এবং পরবর্তী অধ্যায়গুলিতে আপনি যে ক্লাসগুলি (কাস্টম টাইপ) সংজ্ঞায়িত করবেন সেগুলিও পাইথনের অবজেক্ট।

মেমোরি অ্যাড্রেস, রেফারেন্স এবং "পয়েন্টার"

আপনি একটি বস্তুর সাথে একটি রেফারেন্সের মাধ্যমে যোগাযোগ করেন, যা পর্দার আড়ালে কম্পিউটারের মেমরিতে সেই বস্তুর ঠিকানা (বা অবস্থান) থাকে—অন্যান্য ভাষায় যাকে কখনও কখনও "পয়েন্টার" বলা হয়। যেমন একটি অ্যাসাইনমেন্টের পরে

```
x = ৭
```

x চলকটিতে আসলে 7 মান থাকে না। বরং, এটি মেমরির অন্য কোথাও সংরক্ষিত 7 ধারণকারী একটি বস্তুর রেফারেন্স ধারণ করে। আপনি বলতে পারেন যে x "7 ধারণকারী বস্তুর দিকে নির্দেশ করে" (অর্থাৎ, রেফারেন্স), যেমনটি নীচের চিত্রে দেখানো হয়েছে:



বিল্ট-ইন ফাংশন আইডি এবং অবজেক্ট আইডেন্টিটি

আসুন দেখি কিভাবে আমরা ফাংশনে আর্গুমেন্ট পাস করি। প্রথমে, উপরে উল্লিখিত পূর্ণসংখ্যা ভেরিয়েবল x তৈরি করি—শীত্বার আমরা ফাংশন আর্গুমেন্ট হিসেবে x ব্যবহার করব:

```
[1] তে: x = ৭
```

এখন x বলতে 7 ধারণকারী পূর্ণসংখ্যার বস্তুকে বোঝায় (অথবা "নির্দেশ করে")। মেমোরিতে একই ঠিকানায় দুটি পৃথক বস্তু থাকতে পারে না, তাই মেমোরিতে প্রতিটি বস্তুর একটি অন্য ঠিকানা থাকে। যদিও আমরা কোনও বস্তুর ঠিকানা দেখতে পাই না, আমরা বিল্টইন আইডি ফাংশন ব্যবহার করতে পারি।

একটি অনন্য int মান পেতে যা মেমরিতে থাকাকালীন শুধুমাত্র সেই বস্তুটিকে সনাত্ত করে (আপনার কম্পিউটারে এটি চালানোর সময় আপনি সম্ভবত একটি ভিন্ন মান পাবেন):

```
[2] তে: id(x)
আউট[2]: 4350477840
```

আইডি কল করার পূর্ণসংখ্যার ফলাফলকে অবজেক্টের **আইডেন্টিটি বলা হয়।** মেমরিতে থাকা কোনও দুটি অবজেক্টের একই পরিচয় থাকতে পারে না। আমরা অবজেক্ট আইডেন্টিটি ব্যবহার করে দেখাবো যে অবজেক্টগুলি রেফারেন্স দ্বারা পাস করা হয়েছে।

⁸ পাইথন ডকুমেন্টেশন অনুসারে, আপনি যে পাইথন বাস্তবায়ন ব্যবহার করছেন তার উপর নির্ভর করে, একটি অবজেক্ট আইডেন্টিটি অবজেক্টের প্রকৃত মেমরি ঠিকানা হতে পারে, তবে এটি প্রয়োজনীয় নয়।

একটি ফাংশনে একটি অবজেক্ট পাস করা

চলুন একটি কিউব ফাংশন সংজ্ঞায়িত করি যা তার প্যারামিটারের পরিচয় প্রদর্শন করে, তারপর প্যারামিটারের মান কিউব করে ফেরত দেয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[3] তে: def cube(number):
...:     মুদ্রণ করন ('আইডি (নম্বর):', আইডি (নম্বর))
...:     রিটার্ন নম্বর ** ৩
...:
```

এরপর, আসুন x আর্গুমেন্ট দিয়ে কিউব কল করি, যা পূর্ণসংখ্যার বস্তুকে বোঝায় যার মধ্যে রয়েছে ৭:

```
[4] তে: ঘনক(x)
আইডি(নম্বর): ৪৩৫০৪৭৭৮৪০
আউট[4]: 343
```

কিউবের প্যারামিটার নম্বর—৪৩৫০৪৭৭৮৪০—এর জন্য প্রদর্শিত পরিচয়টি পূর্বে x-এর জন্য প্রদর্শিত পরিচয়ের মতোই। যেহেতু প্রতিটি বস্তুর একটি অনন্য পরিচয় থাকে, তাই কিউব কার্যকর করার সময় আর্গুমেন্ট x এবং প্যারামিটার নম্বর উভয়ই একই বস্তুকে নির্দেশ করে।

সুতরাং যখন ফাংশন কিউব তার গণনায় তার প্যারামিটার নম্বর ব্যবহার করে, তখন এটি কলারের মূল বস্তু থেকে সংখ্যার মান পায়।

is অপারেটর দিয়ে অবজেক্ট আইডেন্টিটি পরীক্ষা করা হচ্ছে

আপনি প্রমাণ করতে পারেন যে যুক্তি এবং প্যারামিটার একই বস্তুর সাথে সম্পর্কিত
পাইথন হল অপারেটর, যা True প্রদান করে যদি এর দুটি অপারেন্ডের পরিচয় একই থাকে:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

```
[5] তে: def cube(number):
...:     print('সংখ্যা হল x:', সংখ্যা হল x) # x হল একটি বিশ্বব্যাচী ভেরিয়েব e
...:     রিটার্ন নম্বর ** ৩
...:
```

```
[6] তে: ঘনক(x)
সংখ্যাটি x: সত্য
আউট[6]: 343
```

আর্গুমেন্ট হিসেবে অপরিবর্তনীয় বস্তু

যখন একটি ফাংশন আর্গুমেন্ট হিসেবে একটি অপরিবর্তনীয় (অপরিবর্তনীয়) বস্তুর রেফারেন্স প্রহণ করে—যেমন একটি int, float, string বা tuple—যদিও আপনার কলারে মূল বস্তুতে সরাসরি অ্যাক্সেস আছে, আপনি মূল অপরিবর্তনীয় বস্তুর মান পরিবর্তন করতে পারবেন না। এটি প্রমাণ করার জন্য, প্রথমে একটি বর্ধিত অ্যাসাইনমেন্টের মাধ্যমে প্যারামিটার নম্বরে একটি নতুন বস্তু নির্ধারণের আগে এবং পরে কিউব ডিসপ্লে id(number) দেখি:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

```
[7] তে: def cube(number):
...:     print('id(number) before modify number:', id(number))
...:     সংখ্যা **= ৩
...:     print('id(number) after modify number:', id(number))
...:     রিটার্ন নম্বর
...:
```

```
[8] তে: ঘনক(x)
নম্বর পরিবর্তন করার আগে আইডি(নম্বর): 4350477840
নম্বর পরিবর্তনের পর আইডি(নম্বর): 4396653744
আউট[8]: 343
```

যখন আমরা cube(x) বলি, তখন প্রথম মুদ্রিত বিবৃতিটি দেখায় যে id(number) প্রাথমিকভাবে স্লিপেট [2] তে id(x) এর মতো।
সংখ্যাসূচক মানগুলি অপরিবর্তনীয়, তাই বিবৃতিটি

সংখ্যা **= ৩

আসলে ঘনকযুক্ত মান ধারণকারী একটি নতুন বস্তু তৈরি করে, তারপর সেই বস্তুর রেফারেন্স প্যারামিটার নম্বরে বরাদ্দ করে।

মনে রাখবেন যদি আর কোন রেফারেন্স না থাকে

মূল বস্তু, এটি আবর্জনা সংগ্রহ করা হবে। ফাংশন কিউবের দ্বিতীয় মুদ্রণ বিবৃতিটি নতুন বস্তুর পরিচয় দেখায়। বস্তুর পরিচয় অবশ্যই অনন্য হতে হবে, তাই সংখ্যাটি অবশ্যই একটি ভিন্ন বস্তুর উল্লেখ করবে। x পরিবর্তন করা হয়নি তা দেখানোর জন্য, আমরা আবার এর মান এবং পরিচয় প্রদর্শন করব:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[9] তে: print(f'x = {x}; id(x) = {id(x)}')
x = ৭; আইডি(x) = ৪৩৫০৪৭৭৮৪০
```

আর্গুমেন্ট হিসেবে পরিবর্তনযোগ্য বস্তু

পরবর্তী অধ্যায়ে, আমরা দেখাবো যে যখন একটি পরিবর্তনযোগ্য বস্তুর রেফারেন্স, যেমন একটি তালিকা, একটি ফাংশনে পাস করা হয়, তখন ফাংশনটি কলারের মূল বস্তুটিকে পরিবর্তন করতে পারে।

৪.১৬ পুনরাবৃত্তি

চলুন একটি বিখ্যাত গাণিতিক গণনা করার জন্য একটি প্রোগ্রাম লিখি। একটি ধনাত্মক পূর্ণসংখ্যা n এর ফ্যাক্টোরিয়াল বিবেচনা করুন, যা $n!$ লেখা হয় এবং " n ফ্যাক্টোরিয়াল" উচ্চারণ করা হয়। এটি হল গুণফল

$$n \cdot (n - 1) \cdot (n - 2) \cdots 1$$

$1!$ এর সমান 1 এবং $0!$ এর সাথে 1 হিসেবে সংজ্ঞায়িত করা হয়েছে। উদাহরণস্বরূপ, $5!$ হল $5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$ এর গুণফল, যা 120 এর সমান।

পুনরাবৃত্ত ফ্যাক্টোরিয়াল পদ্ধতি

আপনি for স্টেটমেন্ট ব্যবহার করে পুনরাবৃত্তিমূলকভাবে $5!$ গণনা করতে পারেন, যেমন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: ফ্যাক্টোরিয়াল = ১
```

```
[2] তে: পরিসরে সংখ্যার জন্য (৫, ০, ১):
```

```
...:           ফ্যাক্টোরিয়াল *= সংখ্যা
...:
```

```
[3] তে: ফ্যাক্টোরিয়াল
```

```
আউট[3]: 120
```

রিকার্সিভ সমস্যা সমাধানের পদ্ধতির মধ্যে বেশ কিছু উপাদান মিল রয়েছে। যখন আপনি কোনও সমস্যা সমাধানের জন্য একটি রিকার্সিভ ফাংশন কল করেন, তখন এটি আসলে কেবল সবচেয়ে সহজ কেস(গুলি) বা [বেস কেস\(গুলি\)](#) সমাধান করতে সক্ষম। যদি আপনি ফাংশনটিকে একটি বেস কেস দিয়ে কল করেন, তবে এটি তাৎক্ষণিকভাবে একটি ফলাফল প্রদান করে। যদি আপনি ফাংশনটিকে আরও জটিল সমস্যাযুক্ত কল করেন, তবে এটি সাধারণত সমস্যাটিকে দুটি ভাগে বিভক্ত করে - একটি যা ফাংশনটি কীভাবে করতে হয় তা জানে এবং অন্যটি যা এটি কীভাবে করতে হয় তা জানে না। রিকার্সনকে সম্ভব করে তুলতে, এই শেষ অংশটি মূল সমস্যার কিছুটা সহজ বা ছোট সংস্করণ হতে হবে। যেহেতু এই নতুন সমস্যাটি মূল সমস্যার সাথে সাদৃশ্যপূর্ণ, তাই ছোট সমস্যাটি নিয়ে কাজ করার জন্য ফাংশনটি নিজের একটি নতুন কপি কল করে - এটিকে রিকার্সিভ কল বলা হয় এবং এটিকে [রিকার্সন ধাপও বলা হয়।](#) সমস্যাটিকে দুটি ছোট অংশে বিভক্ত করার এই ধারণাটি বইটিতে আগে প্রবর্তিত divideandconquer পদ্ধতির একটি রূপ।

মূল ফাংশন কলটি সক্রিয় থাকাকালীন (অর্থাৎ, এটি কার্যকর করা শেষ না হওয়া পর্যন্ত) রিকার্সন ধাপটি কার্যকর হয়। ফাংশনটি প্রতিটি নতুন উপ-সমস্যাকে দুটি ধারণাগত অংশে বিভক্ত করার ফলে এর ফলে আরও অনেক রিকার্সিভ কল আসতে পারে। রিকার্সনটি শেষ পর্যন্ত শেষ হওয়ার জন্য, প্রতিবার যখন ফাংশনটি মূল সমস্যার একটি সহজ সংস্করণের সাথে নিজেকে কল করে, তখন ছোট এবং ছোট সমস্যার ক্রম একটি বেস কেসের উপর একত্রিত হতে হবে।

যখন ফাংশনটি বেস কেস সনাক্ত করে, তখন এটি ফাংশনের পূর্ববর্তী কপিতে একটি ফলাফল ফেরত দেয়। রিটার্নের একটি ক্রম চলতে থাকে যতক্ষণ না মূল ফাংশন কলটি ফেরত দেয়।
কলকারীর কাছে চূড়ান্ত ফলাফল।

রিকার্সিভ ফ্যাক্টোরিয়াল অ্যাপ্রোচ

n! লেখা সম্ভব তা পর্যবেক্ষণ করে আপনি একটি পুনরাবৃত্ত ফ্যাক্টোরিয়াল উপস্থাপনায় পৌঁছাতে পারেন

যেমন:

$$n! = n \cdot (n - 1)!$$

উদাহরণস্বরূপ, 5! হল $5 \cdot 4!$ এর সমান, যেমন:

$$5! = 5 \cdot 4 \cdot 3 \cdot 2 \cdot 1$$

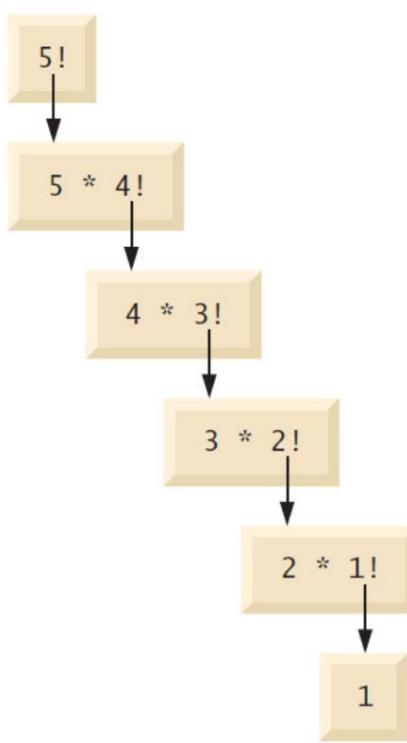
$$5! = 5 \cdot (4 \cdot 3 \cdot 2 \cdot 1)$$

$$5! = 5 \cdot (4!)$$

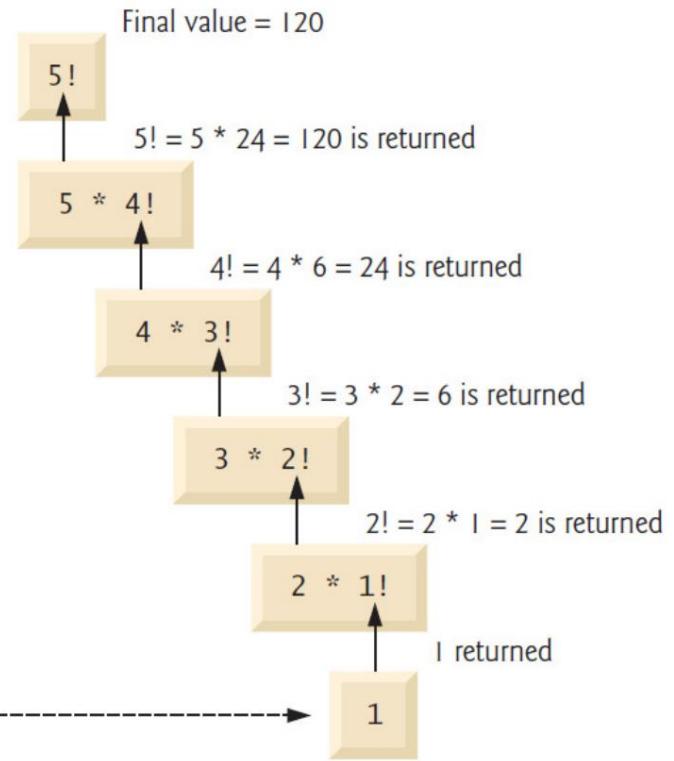
ডিজুয়ালাইজিং রিকার্শন

5! এর মূল্যায়ন নীচে দেখানো পদ্ধতি অনুসারে হবে। বাম কলামে দেখানো হয়েছে কিভাবে রিকার্সিভ কলের ধারাবাহিকতা 1! (বেস কেস) 1 হিসাবে মূল্যায়ন না করা পর্যন্ত এগিয়ে যায়, যা রিকার্সন শেষ করে। ডান কলামে নীচ থেকে উপরে পর্যন্ত মানগুলি দেখানো হয়েছে।

প্রতিটি পুনরাবৃত্ত কল থেকে তার কলারে ফিরে আসে যতক্ষণ না চূড়ান্ত মান গণনা করা হয় এবং



(a) Sequence of recursive calls



(b) Values returned from each recursive call

একটি রিকার্সিভ ফ্যাক্টোরিয়াল ফাংশন বাস্তবায়ন

নিম্নলিখিত সেশনটি 0 থেকে 10 পর্যন্ত পূর্ণসংখ্যার ফ্যাক্টোরিয়াল গণনা এবং প্রদর্শনের জন্য পুনরাবৃত্তি ব্যবহার করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[4] তে: def factorial(number):
    ...
        """সংখ্যার ফ্যাক্টোরিয়াল ফেরত দিন।"""
    ...
        যদি সংখ্যা <= 1:
            ...
                ১ ফেরত দিন
        ...
            রিটার্ন নম্বর * ফ্যাক্টোরিয়াল (নম্বর ১) # রিকার্সিভ কল
    ...

```

```
[5] তে: range(11) তে i এর জন্য : print(f'{i}! =
```

```
    ...
        {factorial(i)}')
```

```
    ...
```

```
0! = ১
```

```
১! = ১
```

```
২! = ২
```

```
৩! = ৬
```

```
৪! = ২৪
```

```
৫! = ১২০
```

```
৬! = ৭২০
```

```
৭! = ৫০৪০
```

```
৮! = ৪০৩২০
```

```
৯! = ৩৬২৮৮০
```

```
১০! = ৩৬২৮৮০
```

স্লিপেট [4] এর রিকার্সিভ ফাংশন ফ্যাক্টরিয়াল প্রথমে নির্ধারণ করে যে সমাপ্তিকারী শর্ত সংখ্যা ≤ 1 সত্য কিনা। যদি এই শর্তটি সত্য হয় (বেস কেস), ফ্যাক্টরিয়াল

১ রিটার্ন করে এবং আর কোন রিকারশনের প্রয়োজন হয় না। যদি সংখ্যা ১ এর চেয়ে বড় হয়, তাহলে দ্বিতীয় রিটার্ন স্টেটমেন্টটি সমস্যাটিকে সংখ্যার গুণফল এবং ফ্যাক্টরিয়ালকে একটি রিকারসিভ কল হিসেবে প্রকাশ করে যা ফ্যাক্টরিয়াল (সংখ্যা ১) মূল্যায়ন করে। এটি মূল গণনা, ফ্যাক্টরিয়াল (সংখ্যা) এর তুলনায় সামান্য ছোট সমস্যা। মনে রাখবেন যে ফাংশন ফ্যাক্টরিয়াল অবশ্যই একটি অ-খাণ্ডক যুক্তি গ্রহণ করবে। আমরা এর জন্য পরীক্ষা করি না।

মামলা।

স্লিপেট [5] এর লুপটি ০ থেকে 10 পর্যন্ত মানের জন্য ফ্যাক্টরিয়াল ফাংশন কল করে। আউটপুট দেখায় যে ফ্যাক্টরিয়াল মানগুলি দ্রুত বৃদ্ধি পায়। পাইথন অন্যান্য অনেক প্রোগ্রামিং ভাষার মতো পূর্ণসংখ্যার আকার সীমাবদ্ধ করে না।

পরোক্ষ পুনরাবৃত্তি

একটি রিকার্সিভ ফাংশন অন্য একটি ফাংশন কল করতে পারে, যা পরবর্তীতে রিকার্সিভ ফাংশনে কল ব্যাক করতে পারে। এটি একটি [ইনডাইরেক্ট](#) রিকার্সিভ কল বা [ইনডাইরেক্ট](#) নামে পরিচিত।

পুনরাবৃত্তি। উদাহরণস্বরূপ, ফাংশন A ফাংশন B কে কল করে, যা কল ব্যাক করে ফাংশন A। এটি এখনও পুনরাবৃত্তি কারণ ফাংশন A-তে দ্বিতীয় কলটি করা হয় যখন ফাংশন A-তে প্রথম কলটি সক্রিয়। অর্থাৎ, ফাংশন A-তে প্রথম কলটি এখনও কার্যকর করা শেষ হয়নি (কারণ এটি ফাংশন B-তে ফলাফল ফেরত দেওয়ার জন্য অপেক্ষা করছে) এবং ফাংশন A-এর মূল কলারে ফিরে আসেন।

স্ট্যাক ওভারফ্লো এবং অসীম পুনরাবৃত্তি

অবশ্যই, একটি কম্পিউটারে মেমোরির পরিমাণ সীমিত, তাই ফাংশনকল স্ট্যাকে অ্যাক্টিভেশন রেকর্ড সংরক্ষণের জন্য শুধুমাত্র একটি নির্দিষ্ট পরিমাণ মেমোরি ব্যবহার করা যেতে পারে। যদি স্ট্যাকে অ্যাক্টিভেশন রেকর্ড সংরক্ষণের চেয়ে বেশি রিকার্সিভ ফাংশন কল ঘটে, তাহলে স্ট্যাক ওভারফ্লো নামে পরিচিত একটি মারাত্মক ত্রুটি ঘটে। এটি সাধারণত [অসীম রিকারশনের](#) ফলাফল, যা বেস কেস বাদ দেওয়ার কারণে বা রিকার্সন ধাপটি ভুলভাবে লেখার ফলে হতে পারে যাতে এটি বেস কেসে একত্রিত না হয়। এই ত্রুটিটি একটি পুনরাবৃত্ত (অ-রিকার্সিভ) সমাধানে একটি অসীম লুপের সমস্যার অনুরূপ।

৪.১৭ কার্যকরী-শেলী প্রোগ্রামিং

জাভা এবং সি# এর মতো অন্যান্য জনপ্রিয় ভাষার মতো, পাইথন সম্পূর্ণরূপে কার্যকরী ভাষা নয়। বরং, এটি "ফাংশনালস্টাইল" বৈশিষ্ট্যগুলি অফার করে যা আপনাকে কোড লিখতে সাহায্য করে যাতে ত্রুটি থাকার সন্তুষ্টি কম, আরও সংক্ষিপ্ত এবং পড়া, ডিবাগ এবং পরিবর্তন করা সহজ।

ফাংশনালস্টাইল প্রোগ্রামগুলিকে আরও ভালো পারফরম্যান্স পেতে সমান্তরাল করা সহজ হতে পারে।

আজকের মাল্টিকোর প্রসেসর। নীচের চার্টে পাইথনের বেশিরভাগ গুরুত্বপূর্ণ কার্যকরী-শিলীল প্রোগ্রামিং ক্ষমতা তালিকাভুক্ত করা হয়েছে এবং বন্ধনীতে সেই অধ্যায়গুলি দেখানো হয়েছে যেখানে আমরা প্রাথমিকভাবে তাদের অনেকগুলি নিয়ে আলোচনা করব।

কার্যকরী স্টাইল প্রোগ্রামিং বিষয়গুলি		
পার্শ্ব প্রতিক্রিয়া এড়ানো (4)	জেনারেটর ফাংশন	
বন্ধ	উচ্চতর ক্রম	অলস মূল্যায়ন (5)
ঘোষণামূলক প্রোগ্রামিং (4)	ফাংশন (5)	তালিকার বোধগম্যতা (5)
	অপরিবর্তনীয়তা (4)	অপারেটর মডিউল (5, 11, 16)
সাজসজ্জাকারী (১০)	অভ্যন্তরীণ পুনরাবৃত্তি (4)	
অভিধানের বোধগম্যতা (6)	পুনরাবৃত্তিকারী (3)	বিশুদ্ধ ফাংশন (4)
ফিল্টার/ম্যাপ/হ্রাস (5)	itertools মডিউল	রেঞ্জ ফাংশন (3, 4)
functools মডিউল	ল্যান্ডডা এক্সপ্রেশন	হ্রাস (৩, ৫)
জেনারেটর এক্সপ্রেশন (5)	(৫)	সেট কম্প্রেশন (6)

আমরা বই জুড়ে এই বৈশিষ্ট্যগুলির বেশিরভাগই কভার করেছি—অনেকগুলি কোড উদাহরণ সহ এবং অন্যান্যগুলি সাক্ষরতার দৃষ্টিকোণ থেকে। আপনি ইতিমধ্যেই for স্টেটমেন্ট সহ তালিকা, স্ট্রিং এবং বিল্টইন ফাংশন রেঞ্জ ইটারেটর এবং বেশ কয়েকটি হ্রাস (ফাংশন sum, len, min এবং max) ব্যবহার করেছেন। আমরা ঘোষণামূলক প্রোগ্রামিং, অপরিবর্তনীয়তা এবং অভ্যন্তরীণ নিচে পুনরাবৃত্তি।

কী বনাম কীভাবে

আপনার সম্পাদিত কাজগুলি যত জটিল হয়ে উঠবে, আপনার কোডটি পড়া, ডিবাগ করা এবং সংশোধন করা তত কঠিন হয়ে পড়বে এবং ক্রটি থাকার সম্ভাবনাও তত বেশি থাকবে। কোডটি কীভাবে কাজ করে তা নির্দিষ্ট করা জটিল হয়ে উঠতে পারে।

ফাংশনালস্টাইল প্রোগ্রামিং আপনাকে কেবল বলতে দেয় যে আপনি কী করতে চান। এটি অনেক কিছু লুকিয়ে রাখে

প্রতিটি কাজ কীভাবে সম্পাদন করতে হবে তার বিশদ বিবরণ। সাধারণত, লাইব্রেরি কোড আপনার জন্য কীভাবে পরিচালনা করে। যেমন দেখবেন, এটি অনেক ক্রটি দূর করতে পারে।

অন্যান্য অনেক প্রোগ্রামিং ভাষার for স্টেটমেন্ট বিবেচনা করুন। সাধারণত, আপনাকে কাউন্টারকন্ট্রোলড ইটারেশনের সমস্ত বিবরণ উল্লেখ করতে হবে: একটি কন্ট্রোল ভেরিয়েবল, এর প্রাথমিক মান, এটি কীভাবে বৃদ্ধি করা যায় এবং একটি লুপকন্টিনিউয়েশন শর্ত যা পুনরাবৃত্তি চালিয়ে যেতে হবে কিনা তা নির্ধারণ করতে নিয়ন্ত্রণ ভেরিয়েবল ব্যবহার করে। পুনরাবৃত্তির এই ধরণটি **বাহ্যিক পুনরাবৃত্তি** নামে পরিচিত এবং এটি ক্রটিপ্রবণ। উদাহরণস্বরূপ, আপনি একটি ভুল ইনিশিয়ালাইজার, ইনক্রিমেন্ট বা লুপকন্টিনিউয়েশন শর্ত প্রদান করতে পারেন। বাহ্যিক পুনরাবৃত্তি নিয়ন্ত্রণ ভেরিয়েবলকে **পরিবর্তন করে** (অর্থাৎ, পরিবর্তন করে) এবং for স্টেটমেন্টের সুষ্ঠু প্রায়শই অন্যান্য পরিবর্তন করে

ভেরিয়েবলও। প্রতিবার যখন আপনি ভেরিয়েবল পরিবর্তন করবেন তখন আপনি ক্রটিগুলি প্রবর্তন করতে পারেন।

ফাংশনালস্টাইল প্রোগ্রামিং অপরিবর্তনীয়তার উপর জোর দেয়। অর্থাৎ, এটি এমন ক্রিয়াকলাপ এড়িয়ে চলে যা ভেরিয়েবলের মান পরিবর্তন করে। আমরা পরবর্তী অধ্যায়ে আরও বিস্তারিত আলোচনা করব।

পাইথনের for statement এবং range ফাংশন বেশিরভাগ কাউন্টার-কন্ট্রোলড পুনরাবৃত্তির বিবরণ লুকায়। আপনি নির্দিষ্ট করেন যে range কোন মান তৈরি করবে এবং কোন ভেরিয়েবল প্রতিটি মান তৈরি হওয়ার সাথে সাথে গ্রহণ করবে। Function range জানে কীভাবে সেই মানগুলি তৈরি করতে হয়। একইভাবে, for statement জানে কীভাবে range থেকে প্রতিটি মান পেতে হয় এবং যখন আর কোন মান থাকে না তখন কীভাবে পুনরাবৃত্তি বন্ধ করতে হয়। অভ্যন্তরীণ পুনরাবৃত্তির একটি গুরুত্বপূর্ণ দিক হল কী, কিন্তু কীভাবে নয় তা নির্দিষ্ট করা - একটি গুরুত্বপূর্ণ ফাংশনাল স্টাইল প্রোগ্রামিং ধারণা।

পাইথনের অন্তর্নির্মিত ফাংশন sum, min এবং max প্রতিটি অভ্যন্তরীণ পুনরাবৃত্তি ব্যবহার করে। তালিকার গ্রেডের উপাদানগুলিকে মোট করার জন্য, আপনাকে কেবল আপনি কী করতে চান তা ঘোষণা করতে হবে - অর্থাৎ, sum(grades)। ফাংশন sum জানে কীভাবে তালিকার মধ্য দিয়ে পুনরাবৃত্তি করতে হয় এবং প্রতিটি উপাদানকে চলমান মোটে যোগ করতে হয়। প্রোগ্রামিং কীভাবে করতে হয় তা বলার পরিবর্তে আপনি কী করতে চান তা বলাকে ঘোষণামূলক প্রোগ্রামিং বলা হয়।

বিশুদ্ধ ফাংশন

বিশুদ্ধ ফাংশনাল প্রোগ্রামিং ল্যাঙ্গুয়েজে আপনি বিশুদ্ধ ফাংশন লেখার উপর জোর দেন। একটি **বিশুদ্ধ ফাংশনের** ফলাফল কেবলমাত্র আপনি যে আর্গুমেন্ট(গুলি) এতে পাঠান তার উপর নির্ভর করে। এছাড়াও, একটি নির্দিষ্ট আর্গুমেন্ট (বা আর্গুমেন্ট) দেওয়া হলে, একটি বিশুদ্ধ ফাংশন সর্বদা একই ফলাফল দেয়।

উদাহরণস্বরূপ, বিল্টইন ফাংশন sum এর রিটার্ন মান শুধুমাত্র আপনি যে ইটারেবলটি পাস করেন তার উপর নির্ভর করে। একটি তালিকা [1, 2, 3] দেওয়া হলে, sum সর্বদা 6 প্রদান করে, আপনি যতবারই এটিকে কল করুন না কেন। এছাড়াও, একটি pure ফাংশনের কোনও পার্শ্ব প্রতিক্রিয়া নেই। উদাহরণস্বরূপ, আপনি যদি একটি পরিবর্তনযোগ্য তালিকা একটি pure ফাংশনে পাস করেন, তবুও তালিকাটিতে ফাংশন কলের আগে এবং পরে একই মান থাকবে। আপনি যখন pure ফাংশন sum কল করেন, তখন এটি তার আর্গুমেন্ট পরিবর্তন করে না।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: মান = [1, 2, 3]

[2] তে: sum(মান)

আউট[2]: 6

[3] তে: sum(values) # একই কল সর্বদা একই ফলাফল প্রদান করে

আউট[3]: 6

[4] তে: মান

আউট[5]: [1, 2, 3]

পরবর্তী অধ্যায়ে, আমরা ফাংশনাল স্টাইল প্রোগ্রামিং ধারণাগুলি ব্যবহার চালিয়ে যাব। এছাড়াও, আপনি দেখতে পাবেন যে ফাংশনগুলি হল এমন বস্তু যা আপনি অন্যান্য ফাংশনে ডেটা হিসাবে প্রেরণ করতে পারেন।

৪.১৮ তথ্য বিজ্ঞানের ভূমিকা: বিচ্ছুরণের পরিমাপ

বর্ণনামূলক পরিসংখ্যানের আলোচনায়, আমরা কেন্দ্রীয় প্রবণতার পরিমাপ বিবেচনা করেছি - গড়, মধ্যমা এবং মোড়।

এগুলি আমাদের একটি গোষ্ঠীতে সাধারণ মানগুলিকে শ্রেণীবদ্ধ করতে সাহায্য করে - যেমন আপনার সহপাঠীদের গড় উচ্চতা বা একটি নির্দিষ্ট দেশে সর্বাধিক কেনা গাড়ির ব্র্যান্ড (মোড়)।

যখন আমরা একটি গোষ্ঠীর কথা বলি, তখন সমগ্র গোষ্ঠীকে **জনসংখ্যা** বলা হয়।

কখনও কখনও জনসংখ্যা বেশ বড় হয়, যেমন পরবর্তী মার্কিন রাষ্ট্রপতি নির্বাচনে ভোট দেওয়ার সম্ভাবনা বেশি, যা ১০ কোটিরও বেশি লোকের। বাস্তবিক কারণে, পরবর্তী রাষ্ট্রপতি কে হবেন তা ভবিষ্যত্বাণী করার চেষ্টাকারী পোলিং সংস্থাগুলি জনসংখ্যার সাবধানে নির্বাচিত ছোট উপ-উপাংশগুলিকে নিয়ে কাজ করে যাকে **নমুনা** বলা হয়। ২০১৬ সালের নির্বাচনের অনেক জরিপে প্রায় ১০০০ জনের নমুনা আকার ছিল।

এই বিভাগে, আমরা মৌলিক বর্ণনামূলক পরিসংখ্যান নিয়ে আলোচনা চালিয়ে যাচ্ছি। আমরা **বিচ্ছুরণের পরিমাপ** (যাকে **পরিবর্তনশীলতার পরিমাপও বলা হয়**) পরিচয় করিয়ে দেব যা আপনাকে মানগুলি কতটা বিস্তৃত তা বুঝতে সাহায্য করবে। উদাহরণস্বরূপ, শিক্ষার্থীদের একটি শ্রেণিতে, এমন একদল শিক্ষার্থী থাকতে পারে যাদের উচ্চতা গড়ের কাছাকাছি, এবং অন্য সংখ্যক শিক্ষার্থী থাকতে পারে যারা যথেষ্ট খাটো বা লম্বা।

আমাদের উদ্দেশ্যে, আমরা প্রতিটি বিচ্ছুরণের পরিমাপ হাতে এবং মডিউল পরিসংখ্যান থেকে ফাংশন ব্যবহার করে গণনা করব, নিম্নলিখিত 10 টি ছয়-পার্শ্বযুক্ত জনসংখ্যা ব্যবহার করে

ডাই রোলস:

ভ্যারিয়েন্স নির্ধারণ করতে, আমরা এই মানগুলির গড় দিয়ে শুরু করি—3.5। আপনি এই ফলাফলটি পেতে পারেন মুখ মানের যোগফল, 35, কে রোলের সংখ্যা, 10 দিয়ে ভাগ করে।

এরপর, আমরা প্রতিটি ডাই মান থেকে গড় বিয়োগ করব (এটি কিছু নেতিবাচক ফলাফল তৈরি করে):

“সহজ করার জন্য, জনসংখ্যার তারতম্য গণনা করা হচ্ছিল। একটি সুস্থ পার্থক্য আছে জনসংখ্যার ভ্যারিয়েন্স এবং নমুনা ভ্যারিয়েন্সের মধ্যে। n দিয়ে ভাগ করার পরিবর্তে (আমাদের উদাহরণে ডাই রোলের সংখ্যা), নমুনা ভ্যারিয়েন্স n 1 দিয়ে ভাগ করা হয়। ছোট নমুনার ক্ষেত্রে পার্থক্যটি স্পষ্ট হয় এবং নমুনার আকার বৃদ্ধির সাথে তা তুচ্ছ হয়ে যায়। পরিসংখ্যান মডিউলটি যথাক্রমে জনসংখ্যার ভ্যারিয়েন্স এবং নমুনা ভ্যারিয়েন্স গণনা করার জন্য pvariance এবং variance ফাংশন প্রদান করে।

একইভাবে, পরিসংখ্যান মডিউলটি যথাক্রমে জনসংখ্যার মান বিচুর্ণি এবং নমুনা মান বিচুর্ণি গণনা করার জন্য pstdev এবং stdev ফাংশনগুলি প্রদান করে।

2.5, 0.5, 0.5, 1.5, 2.5, 1.5, 0.5, 0.5, 1.5, 1.5

তারপর, আমরা এই ফলাফলগুলির প্রতিটিকে বর্গ করি (শুধুমাত্র ইতিবাচক ফলাফল প্রদান করে):

6.25, 0.25, 0.25, 2.25, 6.25, 2.25, 0.25, 0.25, 2.25, 2.25

অবশেষে, আমরা এই বর্গগুলির গড় গণনা করি, যা হল 2.25 (22.5 / 10)-এটি হল জনসংখ্যার বৈচিত্র্য। প্রতিটি ডাই মান এবং সমস্ত ডাই মানের গড়ের মধ্যে পার্থক্যকে বর্গ করার মাধ্যমে আউটলায়ারগুলিকে জোর দেওয়া হয় — যে মানগুলি গড় থেকে সবচেয়ে দূরে। আমরা যখন ডেটা বিশ্লেষণের গভীরে প্রবেশ করি, তখন কখনও কখনও আমরা আউটলায়ারগুলির প্রতি যত্নবান মনোযোগ দিতে চাই, এবং কখনও কখনও আমরা সেগুলিকে উপেক্ষা করতে চাই। নিম্নলিখিত কোডটি আমাদের ম্যানুয়াল ফলাফল নিশ্চিত করার জন্য পরিসংখ্যান মডিউলের **পিভেরিয়েন্স** ফাংশন ব্যবহার করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: আমদানি পরিসংখ্যান

[2] তে: statistics.pvariance([1, 3, 4, 2, 6, 5, 3, 4, 5, 2])
আউট[2]: 2.25

স্ট্যান্ডার্ড বিচুর্ণি

আদর্শ **বিচুর্ণি** হল প্রকরণের বর্গমূল (এই ক্ষেত্রে, 1.5), যা বহিমুখী প্রভাব কমিয়ে দেয়। ভ্যারিয়েন্স এবং স্ট্যান্ডার্ড ডেভিয়েশন যত কম হবে

are, ডেটা মানগুলি গড়ের যত কাছাকাছি থাকবে এবং মান এবং গড়ের মধ্যে সামগ্রিক **বিচ্ছুরণ** (অর্থাৎ, **স্প্রেড**) তত কম হবে। নিম্নলিখিত কোডটি পরিসংখ্যান মডিউলের **pstdev** ফাংশন ব্যবহার করে জনসংখ্যার মান **বিচ্যুতি** গণনা করে, আমাদের ম্যানুয়াল ফলাফল নিশ্চিত করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3] তে: `statistics.pstdev([1, 3, 4, 2, 6, 5, 3, 4, 5, 2])`

আউট[3]: 1.5

`pvariance` ফাংশনের ফলাফল গণিত মডিউলের `sqrt` ফাংশনে পাস করলে নিশ্চিত হয়

আমাদের ফলাফল ১.৫:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: **আমদানি** গণিত

[5] তে: `math.sqrt(statistics.pvariance([1, 3, 4, 2, 6, 5, 3, 4, 5, 2]))`

আউট[5]: 1.5

জনসংখ্যার মান বিচ্যুতি বনাম জনসংখ্যার বৈচিত্র্যের সুবিধা

ধরুন আপনি আপনার এলাকার মার্চ মাসের ফারেনহাইট তাপমাত্রা রেকর্ড করেছেন। আপনার কাছে 19, 32, 28 এবং 35 এর মতো 31টি সংখ্যা থাকতে পারে। এই সংখ্যাগুলির একক হল ডিগ্রি।

যখন আপনি জনসংখ্যার প্রকরণ গণনা করার জন্য আপনার তাপমাত্রার বর্গ করেন, তখন জনসংখ্যার প্রকরণের এককগুলি "ডিগ্রি বর্গ" হয়ে যায়। যখন আপনি জনসংখ্যার প্রকরণের বর্গমূল নিয়ে জনসংখ্যার মান বিচ্যুতি গণনা করেন, তখন ইউনিটগুলি আবার ডিগ্রিতে পরিণত হয়, যা আপনার তাপমাত্রার সমান একক।

৪.১৯ র্যাপ-আপ

এই অধ্যায়ে, আমরা কাস্টম ফাংশন তৈরি করেছি। আমরা র্যান্ডম এবং ম্যাথ মডিউল থেকে ক্ষমতা আমদানি করেছি। আমরা র্যান্ডমস্বার জেনারেশন চালু করেছি এবং এটি ব্যবহার করে একটি ছয়-পার্শ্বযুক্ত ডাই রোলিং সিমুলেট করেছি। একটি ফাংশন থেকে একাধিক মান ফেরত দেওয়ার জন্য আমরা একাধিক মান টিপলে প্যাক করেছি। এর মানগুলি অ্যাক্সেস করার জন্য আমরা একটি টিপলও আনপ্যাক করেছি। "পুনর্ড্রাবন" এড়াতে পাইথন স্ট্যান্ডার্ড লাইব্রেরির মডিউলগুলি ব্যবহার করে আমরা আলোচনা করেছি।

চাকা।"

আমরা ডিফল্ট প্যারামিটার মান সহ ফাংশন তৈরি করেছি এবং কীওয়ার্ড আর্গমেন্ট সহ ফাংশন নামকরণ করেছি।

আমরা ইচ্ছাকৃত আর্গমেন্ট তালিকা সহ ফাংশনগুলি ও সংজ্ঞায়িত করেছি। আমরা অবজেক্টের পদ্ধতি নামকরণ করেছি।

আমরা আলোচনা করেছি যে কীভাবে একটি শনাক্তকারীর স্কোপ আপনার প্রোগ্রামের কোথায় তা নির্ধারণ করে।

তুমি এটা ব্যবহার করতে পারো।

আমরা মডিউল আমদানি সম্পর্কে আরও উপস্থাপন করেছি। আপনি দেখেছেন যে আর্গমেন্টগুলি ফাংশনের সাথে সম্পর্কিত, এবং ফাংশনকল স্ট্যাক এবং স্ট্যাক ফ্রেমগুলি কীভাবে ফাংশনক্যাল্যান্ডর্টার্ম প্রক্রিয়া সমর্থন করে। আমরা একটি [রিকার্সিভ ফাংশন](#)ও উপস্থাপন করেছি এবং পাইথনের ফাংশনালস্টাইল প্রোগ্রামিং ক্ষমতাগুলি প্রবর্তন শুরু করেছি। আমরা গত দুটি অধ্যায়ে মৌলিক তালিকা এবং টিপল ক্ষমতাগুলি প্রবর্তন করেছি - পরবর্তী অধ্যায়ে, আমরা

সেগুলো বিস্তারিত আলোচনা করো।

পরিশেষে, আমরা বর্ণনামূলক পরিসংখ্যান নিয়ে আমাদের আলোচনা অব্যাহত রেখেছি বিচ্ছুরণের পরিমাপ—প্রকরণ এবং মানক বিচ্যুতি—প্রবর্তন করে এবং পাইথন স্ট্যান্ডার্ড লাইব্রেরির পরিসংখ্যান মডিউল থেকে ফাংশন ব্যবহার করে সেগুলি গণনা করে।

কিছু ধরণের সমস্যার জন্য, ফাংশনগুলিকে নিজেদের কল করা কার্যকর। একটি [রিকার্সিভ ফাংশন](#) সরাসরি বা পরোক্ষভাবে অন্য ফাংশনের মাধ্যমে নিজেকে কল করে।

ইলিস্ট

ক্ষে. ক্রম: তালিকা এবং টুপল

উদ্দেশ্যমূলক ছবি

এই অধ্যায়ে, আপনি পাবেন: আর্নিং
পাথ

■ তালিকা এবং টিপল তৈরি এবং আরম্ভ করুন।
ফার্স্ট এবং ডিল

■ তালিকা, টিপল এবং স্ট্রিং এর উপাদানগুলি দেখুন।

■ তালিকা সাজান এবং অনুসন্ধান করুন, এবং টিপল অনুসন্ধান করুন।

ফাংশন

■ এবং পদ্ধতিতে তালিকা এবং টিপল পাস করুন।

সমর্থন

■ সাধারণ কারসাজি সম্পাদনের জন্য তালিকা পদ্ধতি ব্যবহার করুন, যেমন আইটেম অনুসন্ধান করা,
সাইনআলজিকা সাজানো, আইটেম সন্নিবেশ করা এবং আইটেম অপসারণ করা।

■ ল্যাম্বডাস সহ অতিরিক্ত পাইথন ফাংশনালস্টাইল প্রোগ্রামিং ক্ষমতা ব্যবহার করুন
এবং ফাংশনাল স্টাইল প্রোগ্রামিং অপারেশন ফিল্টার, ম্যাপ এবং রিডুস করে।

■ দ্রুত এবং সহজে তালিকা তৈরি করতে কার্যকরী শৈলীর তালিকা বোঝার পদ্ধতি ব্যবহার করুন এবং ব্যবহার করুন
চাহিদা অনুযায়ী মান তৈরি করতে এক্সপ্রেশন তৈরি করে।

■ দ্বিমাত্রিক তালিকা ব্যবহার করুন।

■ Seaborn এবং Matplotlib এর মাধ্যমে আপনার বিশ্লেষণ এবং উপস্থাপনা দক্ষতা বৃদ্ধি করুন।
ভিজুয়ালাইজেশন লাইব্রেরি।

রূপরেখা

.1 ভূমিকা

.2 তালিকা

.3 টিপল

.৪ আনপ্যাকিং সিকোয়েল

.৫ সিকোয়েল স্লাইসিং

বিবৃতির .৬

.৭ তালিকাগুলিকে ফাংশনে স্থানান্তর করা

.৮ বাছাই তালিকা

.৯ অনুসন্ধানের ক্রম

.১০ অন্যান্য তালিকা পদ্ধতি

.১১ তালিকা সহ স্ট্যাক সিমুলেশন

.১২ তালিকা বোধগম্যতা

.১৩ জেনারেটর এক্সপ্রেশন

.১৪ ফিল্টার, ম্যাপ এবং রিডুস

.১৫ অন্যান্য সিকোয়েল প্রসেসিং ফাংশন

.১৬ দ্বিমাত্রিক তালিকা

.১৭ ডেটা সায়েন্সের ভূমিকা: সিমুলেশন এবং স্ট্যাটিক ভিজুয়ালাইজেশন

.১৭.১ ৬০০, ৬০,০০০ এবং ৬,০০০,০০০ ডাই রোলের নমুনা গ্রাফ

.১৭.২ ডাইরোল ফ্রিকোয়েন্সি এবং শতাংশের ভিজুয়ালাইজেশন

.১৮ সারসংক্ষেপ

৫.১ ভূমিকা

গত দুটি অধ্যায়ে, আমরা আইটেমের ক্রমানুসারে সংগ্রহের প্রতিনিধিত্ব করার জন্য তালিকা এবং টুপল সিকোয়েলের ধরণগুলি সংক্ষেপে পরিচয় করিয়ে দিয়েছি। **সংগ্রহগুলি** হল প্রাক-প্যাকেজ করা ডেটা স্ট্রাকচার যা সম্পর্কিত ডেটা আইটেমগুলি নিয়ে গঠিত। সংগ্রহের উদাহরণগুলির মধ্যে রয়েছে আপনার স্মার্টফোনে আপনার প্রিয় গান, আপনার পরিচিতি তালিকা, একটি লাইব্রেরির বই, একটি কার্ড গেমে আপনার কার্ড, আপনার প্রিয় ক্রীড়া দলের খেলোয়াড়, একটি বিনিয়োগ পোর্টফোলিওতে থাকা স্টক, একটি ক্যাল্নার গবেষণায় থাকা রোগী এবং একটি শপিং তালিকা। পাইথনের অন্তর্নির্মিত সংগ্রহগুলি আপনাকে সঞ্চয় এবং অ্যাক্রেস করতে সক্ষম করে

সুবিধাজনক এবং দক্ষতার সাথে ডেটা ব্যবহার করা। এই অধ্যায়ে, আমরা তালিকা এবং টিপলগুলি আরও বিশদে আলোচনা করব।

আমরা সাধারণ তালিকা এবং টিপল ম্যানিপুলেশনগুলি প্রদর্শন করব। আপনি দেখতে পাবেন যে তালিকাগুলি (যা পরিবর্তনযোগ্য) এবং টিপলগুলি (যা নয়) এর অনেকগুলি সাধারণ ক্ষমতা রয়েছে। প্রতিটি একই বা বিভিন্ন ধরণের আইটেম ধারণ করতে পারে। তালিকাগুলি প্রয়োজন অনুসারে গতিশীলভাবে আকার পরিবর্তন করতে পারে, কার্যকর করার সময় বৃদ্ধি এবং সংকোচন করতে পারে। আমরা এক-মাত্রিক এবং দ্বি-মাত্রিক নিয়ে আলোচনা করব-

মাত্রিক তালিকা।

পূর্ববর্তী অধ্যায়ে, আমরা র্যান্ডম নাম্বার জেনারেশন এবং সিঙ্ক্র-সাইডেড ডাই রোলিং সিমুলেটেড করে দেখিয়েছি। আমরা এই অধ্যায়টি আমাদের পরবর্তী ডেটা সায়েন্স ভূমিকা বিভাগের মাধ্যমে শেষ করছি, যা ডিজ্যুয়ালাইজেশন লাইব্রেরি সিবর্ন এবং ম্যাটপ্লটলিব ব্যবহার করে ডাই ফ্রিকোয়েন্সি ধারণকারী স্ট্যাটিক বার চার্টগুলি ইন্টারেক্টিভভাবে বিকাশ করে। পরবর্তী অধ্যায়ের ডেটা সায়েন্স ভূমিকা বিভাগে, আমরা একটি অ্যানিমেটেড ডিজ্যুয়ালাইজেশন উপস্থাপন করব যেখানে ডাই রোলের সংখ্যা বৃদ্ধির সাথে সাথে বার চার্টটি গতিশীলভাবে পরিবর্তিত হয় - আপনি "কার্যক্ষম" বৃহৎ সংখ্যার আইন দেখতে পাবেন।

৫.২ তালিকা

এখানে, আমরা তালিকাগুলি আরও বিশদে আলোচনা করব এবং নির্দিষ্ট তালিকার **উপাদানগুলিকে** কীভাবে উল্লেখ করতে হবে তা ব্যাখ্যা করব। এই বিভাগে দেখানো অনেক ক্ষমতা সমন্বয় ধরণের ক্রম অনুসারে প্রয়োজ্য।

একটি তালিকা তৈরি করা

তালিকাগুলি সাধারণত **সমজাতীয় ডেটা সংরক্ষণ করে**, অর্থাৎ একই ডেটা টাইপের মান।

তালিকা c বিবেচনা করুন, যেখানে পাঁচটি পূর্ণসংখ্যা উপাদান রয়েছে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: $c = [45, 6, 0, 72, 1543]$

[2] তে: g

আউট[2]: [45, 6, 0, 72, 1543]

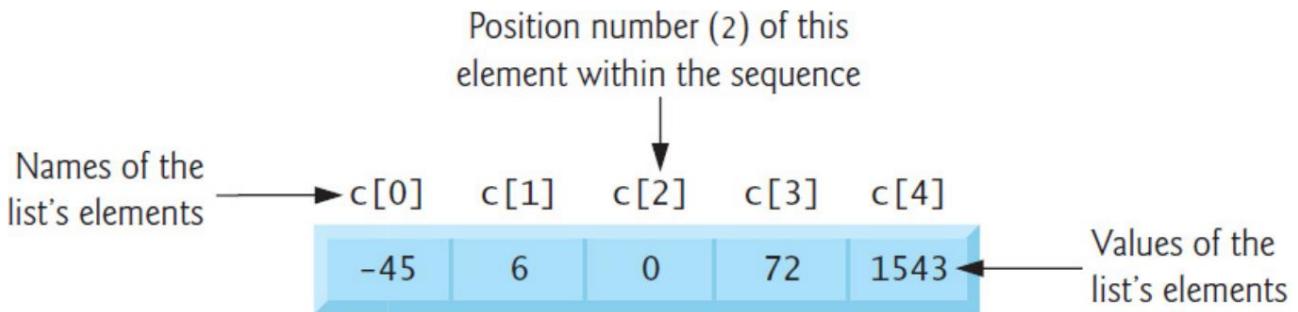
তারা **ভিন্নধর্মী** তথ্যও সংরক্ষণ করতে পারে, অর্থাৎ, বিভিন্ন ধরণের তথ্য। উদাহরণস্বরূপ, নিম্নলিখিত তালিকায় একজন শিক্ষার্থীর প্রথম নাম (একটি স্ট্রিং), শেষ নাম (একটি স্ট্রিং), গ্রেড পয়েন্ট গড় (একটি ফ্লোট) এবং স্নাতক বছর (একটি int) রয়েছে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[‘মেরি’, ‘স্মিথ’, ৩.৫৭, ২০২২]

তালিকার উপাদানগুলিতে অ্যাক্সেস করা

আপনি একটি তালিকার উপাদানের রেফারেন্স হিসেবে তালিকার নাম লিখে তার পরে উপাদানের **সূচক** (অর্থাৎ, এর অবস্থান নম্বর) বর্গাকার বন্ধনীতে ([]), যা **সাবস্ক্রিপশন অপারেটর** নামে পরিচিত) লিখে রাখতে পারেন।
নিম্নলিখিত চিত্রটিতে তালিকাটি C লেবেলযুক্ত দেখানো হয়েছে।
উপাদানের নাম:



তালিকার প্রথম উপাদানটির সূচক 0 থাকে। সুতরাং, পাঁচটি উপাদানের তালিকা C-তে, প্রথম উপাদানটি নামকরণ করা হয়েছে C[0] এবং শেষেরটি হল C[4]:

[3] তে: c[0]

আউট[3]: ৪৫

[4] তে: c[4]

আউট[4]: ১৫৪৩

তালিকার দৈর্ঘ্য নির্ধারণ করা

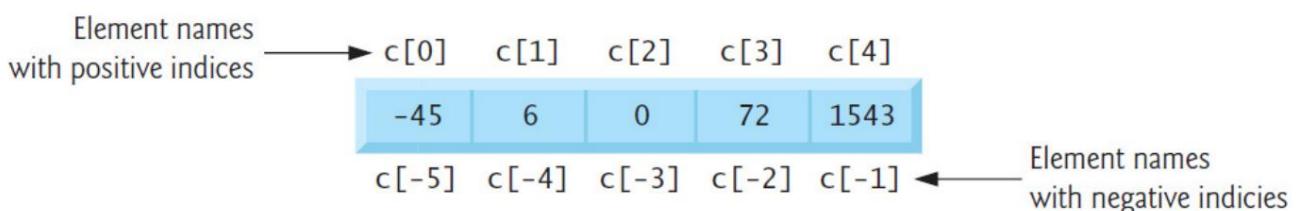
তালিকার দৈর্ঘ্য পেতে, বিল্টইন **len** ফাংশনটি ব্যবহার করুন:

[5] তে: len(c)

আউট[5]: ৫

নেতিবাচক সূচক সহ তালিকার শেষ থেকে উপাদানগুলি অ্যাক্সেস করা

নেতিবাচক সূচক ব্যবহার করে তালিকাগুলি শেষ থেকেও অ্যাক্সেস করা যেতে পারে:



সুতরাং, তালিকা C এর শেষ উপাদান (C[4]), C[1] দিয়ে এবং এর প্রথম উপাদান দিয়ে অ্যাক্সেস করা যেতে পারে গ[ডেফল্ট]:

[6] তে: c[1]

আউট[6]: ১৫৪৩

[7] তে: c[5]

আউট[7]: ৮৫

সূচকগুলি অবশ্যই পূর্ণসংখ্যা অথবা পূর্ণসংখ্যার রাশি হতে হবে

একটি সূচক অবশ্যই একটি পূর্ণসংখ্যা বা পূর্ণসংখ্যার রাশি (অথবা একটি স্লাইস, যা আমরা শীঘ্রই দেখব) হতে হবে:

[8] তে: a = 1

[9] তে: b = 2

[10] তে: c[a + b]

আউট[10]: 72

একটি নন-ইন্টিজার ইনডেক্স মান ব্যবহার করলে একটি TypeError দেখা দেয়।

তালিকা পরিবর্তনযোগ্য

তালিকাগুলি পরিবর্তনযোগ্য - তাদের উপাদানগুলি পরিবর্তন করা যেতে পারে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[11] তে: c[4] = 17

[12] তে: গ

আউট[12]: [৮৫, ৬, ০, ৭২, ১৭]

শীঘ্রই আপনি দেখতে পাবেন যে আপনি তালিকার দৈর্ঘ্য পরিবর্তন করে উপাদানগুলি সন্নিবেশ এবং মুছতেও পারেন।

কিছু সিকোয়েল অপরিবর্তনীয়

পাইথনের স্ট্রিং এবং টিপল সিকোয়েলগুলি অপরিবর্তনীয় - এগুলি পরিবর্তন করা যায় না। আপনি একটি স্ট্রিংয়ে পৃথক অক্ষর পেতে পারেন, কিন্তু একটি অক্ষরের জন্য একটি নতুন মান নির্ধারণ করার চেষ্টা করলে TypeError দেখা দেয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

n [13]: s = 'হ্যালো'

[14] তে: s[0]

আউট[14]: 'h'

[15] তে: s[0] = 'H'

<মডিউল>() এ টাইপ

ট্রেসব্যাক (সর্বশেষ কল শেষ)

কৃটি <ipythoninput15812ef2514689>

> ১ সেকেন্ড[০] = 'এইচ'

TypeError: 'str' অবজেক্ট আইটেম অ্যাসাইনমেন্ট সমর্থন করে না

অস্তিত্বহীন একটি উপাদান অ্যাক্সেস করার চেষ্টা করা

একটি outofrange তালিকা, tuple বা string index ব্যবহার করলে একটি IndexError দেখা দেয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[16] তে: c[100]

সূচক কৃটি

ট্রেসব্যাক (সর্বশেষ কলটি শেষ)

ipythoninput169a31ea1e1a13> <module>() এ

> ১ গ[১০০]

IndexError: তালিকার সূচী সীমার বাইরে

এক্সপ্রেশনে তালিকা উপাদান ব্যবহার করা

তালিকার উপাদানগুলি এক্সপ্রেশনে ভেরিয়েবল হিসেবে ব্যবহার করা যেতে পারে:

[17] তে: c[0] + c[1] + c[2]

আউট[17]: 39

+= দিয়ে তালিকায় যোগ করা হচ্ছে

একটি খালি তালিকা [] দিয়ে শুরু করা যাক, তারপর a for স্টেটমেন্ট এবং += ব্যবহার করে তালিকায় ১ থেকে ৫ পর্যন্ত মান যোগ করুন —প্রতিটি আইটেমকে স্থান দেওয়ার জন্য তালিকাটি গতিশীলভাবে বৃদ্ধি পায়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[18] তে: a_list = []

[19] তে: পরিসরে সংখ্যার জন্য (1, 6):

```
...:           a_list += [সংখ্যা]
...:
```

[20] তে: a_list

আউট[20]: [1, 2, 3, 4, 5]

যখন += এর বাম অপারেন্টটি একটি তালিকা হয়, তখন ডান অপারেন্টটি অবশ্যই পুনরাবৃত্তিযোগ্য হতে হবে; অন্যথায়, একটি TypeError ঘটে। স্মিপেট [19] এর সূচটে, সংখ্যার চারপাশে বর্গাকার বন্ধনীগুলি একটি oneelement তালিকা তৈরি করে, যা আমরা a_list-এ যুক্ত করি। যদি ডান অপারেন্টে একাধিক উপাদান থাকে, তবে += তাদের সকলকে যুক্ত করে। নিম্নলিখিতটি 'Python' এর অক্ষরগুলিকে তালিকার অক্ষরগুলি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[21] তে: অক্ষর = []

[22] তে: অক্ষর += 'পাইথন'

[23] তে: অক্ষর

আউট[23]: ['P', 'y', 't', 'h', 'o', 'n']

যদি += এর ডান অপারেন্টটি একটি টুপল হয়, তাহলে এর উপাদানগুলিও তালিকায় যুক্ত করা হবে। অধ্যায়ের পরে, আমরা তালিকায় আইটেম যোগ করার জন্য list পদ্ধতি append ব্যবহার করব।

+ এর সাথে তালিকা সংযুক্ত করা

আপনি + অপারেটর ব্যবহার করে দুটি তালিকা, দুটি টিপল বা দুটি স্ট্রিং একত্রিত করতে পারেন। ফলাফলটি একই ধরণের একটি নতুন ক্রম তৈরি করে যেখানে বাম অপারেন্টের উপাদানগুলি থাকে এবং তারপরে ডান অপারেন্টের উপাদানগুলি থাকে। মূল ক্রমগুলি অপরিবর্তিত রয়েছে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[24] তে: তালিকা 1 = [10, 20, 30]

[25] তে: তালিকা 2 = [40, 50]

[26] তে: concatenated_list = list1 + list2

[27] তে: concatenated_list

আউট[27]: [10, 20, 30, 40, 50]

যদি + অপারেটরের অপারেন্টগুলি ডিফারেন্স সিকোয়েল টাইপের হয় তাহলে একটি TypeError ঘটে—উদাহরণস্বরূপ, একটি তালিকা এবং একটি টিপলকে সংযুক্ত করা একটি ত্রুটি।

তালিকা সূচক এবং মান অ্যাক্সেস করতে for এবং range ব্যবহার করা

তালিকার উপাদানগুলি তাদের সূচক এবং সার্কিটিপশন অপারেটর ([]) এর মাধ্যমেও অ্যাক্সেস করা যেতে পারে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[28] তে: for i in range(len(concatenated_list)):
...:     মুদ্রণ (f'{i}: {concatenated_list[i]}')
...:
0:১০
1:২০
2:৩০
3:৪০
4:৫০
```

ফাংশন কল range(len(concatenated_list)) concatenated_list এর সূচকগুলি উপস্থাপন করে এমন পূর্ণসংখ্যার একটি ক্রম তৈরি করে (এই ক্ষেত্রে, 0 থেকে 4 পর্যন্ত)। এই পদ্ধতিতে লুপ করার সময়, আপনাকে নিশ্চিত করতে হবে যে সূচকগুলি পরিসরে থাকে। শীত্বাই, আমরা বিল্টইন ফাংশন enumerate ব্যবহার করে উপাদান সূচক এবং মানগুলি অ্যাক্সেস করার একটি নিরাপদ উপায় দেখাব।

তুলনা অপারেটর

আপনি তুলনামূলক অপারেটর ব্যবহার করে elementbyelement এর সম্পূর্ণ তালিকা তুলনা করতে পারেন:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

[29] তে: a = [1, 2, 3]

[30] তে: b = [1, 2, 3]

[31] তে: c = [1, 2, 3, 4]

[32] তে: a == b # সত্য: উভয়ের সংশ্লিষ্ট উপাদান সমান

আউট[32]: সত্য

[33] তে: a == c # মিথ্যা: a এবং c এর উপাদান এবং দৈর্ঘ্য ডিই

আউট[33]: মিথ্যা

[34] তে: a < c # সত্য: a তে c এর চেয়ে কম উপাদান রয়েছে

আউট[34]: সত্য

[35] তে: c >= b # সত্য: 02 উপাদান সমান কিন্তু c তে আরও উপাদান রয়েছে

আউট[35]: সত্য

৫.৩ টিপলস

পূর্ববর্তী অধ্যায়ে আলোচনা করা হয়েছে, টিপলগুলি অপরিবর্তনীয় এবং সাধারণত ভিন্নধর্মী ডেটা সংক্ষয় করে, তবে ডেটা একজাত হতে পারে। একটি টিপলের দৈর্ঘ্য হল এর উপাদানের সংখ্যা এবং প্রোগ্রাম কার্যকর করার সময় এটি পরিবর্তন হতে পারে না।

একটি খালি টুপল তৈরি করা, খালি বন্ধনী ব্যবহার করুন:

[1] তে: student_tuple = ()

[2] তে: student_tuple
আউট[2]: ()

[3] তে: len(student_tuple)
আউট[3]: 0

মনে রাখবেন যে আপনি একটি টুপলকে কমা দিয়ে আলাদা করে প্যাক করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: student_tuple = 'John', ৩.০

[5] তে: student_tuple Out[5]: ('জন',
'সবুজ', 3.3)

[6] তে: len(student_tuple)
আউট[6]: 3

যখন আপনি একটি টুপল আউটপুট করেন, তখন পাইথন সর্বদা তার বিষয়বস্তু বন্ধনীতে প্রদর্শন করে। আপনি এইচিক বন্ধনী দিয়ে একটি টুপলের কমা দ্বারা পৃথক করা মানগুলির তালিকা ঘিরে রাখতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[7] তে: another_student_tuple = ('মেরি',
'লাল', ৩.০)

[8] তে: another_student_tuple Out[8]: ('মেরি', 'রেড',
3.3)

নিম্নলিখিত কোডটি একটি oneelement tuple তৈরি করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[9] তে: a_singleton_tuple = ('লাল',) # কমাটি লক্ষ্য করুন

[10] তে: a_singleton_tuple
আউট[10]: ('লাল',)

'লাল' স্ট্রিং এর পরে থাকা কমা (,) a_singleton_tuple কে tuple হিসেবে চিহ্নিত করে—বন্ধনীগুলি এইচিক। যদি কমাটি বাদ দেওয়া হয়, তাহলে বন্ধনীগুলি হবে

অপ্রয়োজনীয় হবে, এবং `a_singleton_tuple` কেবল `tuple`-এর পরিবর্তে 'red' স্ট্রিংটি উল্লেখ করবে।

টুপল এলিমেন্ট অ্যাক্সেস করা

একটি টুপলের উপাদানগুলি, যদিও সম্পর্কিত, প্রায়শই একাধিক ধরণের হয়। সাধারণত, আপনি সেগুলি পুনরাবৃত্তি করেন না। বরং, আপনি প্রতিটি পৃথকভাবে অ্যাক্সেস করেন। তালিকা সূচকের মতো, টুপল সূচকগুলি 0 থেকে শুরু হয়। নিম্নলিখিত কোডটি `time_tuple` তৈরি করে যা একটি ঘন্টা, মিনিট এবং সেকেন্ডকে প্রতিনিধিত্ব করে, টুপলটি প্রদর্শন করে, তারপর মধ্যরাত থেকে সেকেন্ডের সংখ্যা গণনা করার জন্য এর উপাদানগুলি ব্যবহার করে—মনে রাখবেন যে আমরা টুপলের প্রতিটি মানের সাথে একটি ভিন্ন অপারেশন করি:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

```
[11] তে: time_tuple = (9, 16, 1)
```

```
[12] তে: time_tuple  
আউট[12]: (9, 16, 1)
```

```
[13] তে: time_tuple[0] * 3600 + time_tuple[1] * 60 + time_tuple[2]  
আউট[13]: 33361
```

একটি `tuple` উপাদানে একটি মান নির্ধারণ করলে একটি `TypeError` দেখা দেয়।

একটি স্ট্রিং বা টুপলে আইটেম যোগ করা

তালিকার মতো, `+=` অগমেন্টেড অ্যাসাইনমেন্ট স্টেটমেন্টটি স্ট্রিং এবং টিপলের সাথে ব্যবহার করা যেতে পারে, যদিও এগুলি অপরিবর্তনীয়। নিম্নলিখিত কোডে, দুটি অ্যাসাইনমেন্টের পরে, `tuple1` এবং `tuple2` একই `tuple` অবজেক্টকে নির্দেশ করে:

```
[14] তে: tuple1 = (10, 20, 30)
```

```
[15] তে: tuple2 = tuple1
```

```
[16] তে: tuple2  
আউট[16]: (10, 20, 30)
```

টিউপল `(40, 50)` কে টিউপল`1` এর সাথে সংযুক্ত করলে একটি নতুন টিউপল তৈরি হয়, তারপর এটির জন্য `tuple1` তেরিয়েবলের একটি রেফারেন্স বরাদ্দ করা হয়—টিউপল`2` এখনও মূল টিউপলকে নির্দেশ করে:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

```
[17] তে: tuple1 += (40, 50) www.EBooksWorld.ir
```

[18] তে: টুপল1

আউট[18]: (10, 20, 30, 40, 50)

[19] তে: tuple2

আউট[19]: (10, 20, 30)

একটি স্ট্রিং বা টিপলের জন্য, += এর ডানদিকের আইটেমটি যথাক্রমে একটি স্ট্রিং বা টিপল হতে হবে— প্রকারণে মিশ্রিত করলে একটি TypeError হয়।

তালিকায় টুপল যোগ করা হচ্ছে

আপনি += ব্যবহার করে একটি তালিকায় একটি টুপল যোগ করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[20] তে: সংখ্যা = [1, 2, 3, 4, 5]

[21] তে: সংখ্যা += (6, 7)

[22] তে: সংখ্যা

আউট[22]: [1, 2, 3, 4, 5, 6, 7]

টুপলে পরিবর্তনযোগ্য বস্তু থাকতে পারে

চলুন, প্রথম নাম, পদবি এবং গ্রেডের তালিকা সহ একটি student_tuple তৈরি করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[23] তে: student_tuple = ('আমান্তা',

'নীল', [98, 75, 87])

যদিও টিপলটি অপরিবর্তনীয়, এর তালিকা উপাদানটি পরিবর্তনযোগ্য:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[24] তে: student_tuple[2][1] = 85

[25] তে: student_tuple

আউট[25]: ('আমান্তা', 'নীল', [98, 85, 87])

ডাবলসবক্সিপ্টেড নাম student_tuple[2][1]-এ, Python student_tuple[2] কে তালিকা [98, 75, 87]

ধারণকারী টিপলের উপাদান হিসেবে দেখে, তারপর [1] ব্যবহার করে 75 ধারণকারী তালিকা উপাদানটি অ্যাক্সেস করে। স্লিপেটে অ্যাসাইনমেন্ট [24]

সেই গ্রেডটিকে 85 দিয়ে প্রতিস্থাপন করে।

৫.৪ আনপ্যাকিংয়ের ক্রম

পূর্ববর্তী অধ্যায়ে টুপল আনপ্যাকিং চালু করা হয়েছিল। আপনি যেকোনো সিকোয়েলের উপাদানগুলিকে কমা দ্বারা পৃথক করা ভেরিয়েবলের তালিকায় সিকোয়েল বিবরণ করে আনপ্যাক করতে পারেন। যদি অ্যাসাইনমেন্ট প্রতীকের বাম দিকের ভেরিয়েবলের সংখ্যা ডানদিকের সিকোয়েলের উপাদানের সংখ্যার সাথে সমান না হয় তবে একটি ValueError ঘটে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: student_tuple = ('আমান্ত', [98, 85, 87])
[2] তে: প্রথম_নাম, গ্রেড = ছাত্র_টুপল
[3] তে: প্রথম_নাম
আউট[3]: 'আমান্ত'
[4]: গ্রেড
আউট[4]: [98, 85, 87]
```

নিম্নলিখিত কোডটি একটি স্ট্রিং, একটি তালিকা এবং পরিসর দ্বারা উত্পাদিত একটি ক্রম আনপ্যাক করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[5] তে: প্রথম, দ্বিতীয় = 'হাই'
[6] তে: print(f'{first} {second}')
হাই
[7] তে: সংখ্যা 1, সংখ্যা 2, সংখ্যা 3 = [2,
[8] তে: print(f'{number1} {number2} {number3}')
[9] তে: সংখ্যা 1, সংখ্যা 2, সংখ্যা 3 = পরিসর (10,
[10] তে: print(f'{number1} {number2} {number3}')
১০ ২০ ৩০
```

প্যাকিং এবং আনপ্যাকিংয়ের মাধ্যমে মান অদলবদল করা

সিকোয়েল প্যাকিং এবং আনপ্যাকিং ব্যবহার করে আপনি দুটি ভেরিয়েবলের মান অদলবদল করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[11] তে: সংখ্যা 1 = 99

[12] তে: সংখ্যা 2 = 22

[13] তে: সংখ্যা 1, সংখ্যা 2 = (সংখ্যা 2, সংখ্যা 1)

[14] তে: `print(f'number1 = {number1}; number2 = {number2}')` number1 = 22; number2 = 99

বিল্ট-ইন ফাংশন এনুমেরেটের সাহায্যে নিরাপদে সূচক এবং মান অ্যাক্সেস করা

আগে, আমরা সূচক মানের একটি ক্রম তৈরি করার জন্য range নামক একটি কোড ব্যবহার করেছিলাম, তারপর সূচক মান এবং সাবস্ক্রিপশন অপারেটর ([]) ব্যবহার করে একটি for লুপে তালিকা উপাদানগুলি অ্যাক্সেস করেছি। এটি ক্রটিপ্রবণ কারণ আপনি ভুল আর্গুমেন্টগুলি range-এ পাস করতে পারেন। যদি range দ্বারা উত্পাদিত কোনও মান একটি outofbounds সূচক হয়, তাহলে এটিকে সূচক হিসাবে ব্যবহার করলে একটি সূচক ক্রটি।

একটি এলিমেন্টের ইনডেক্স এবং ভ্যালু অ্যাক্সেস করার জন্য পছন্দের মেকানিজম হল বিল্ট-ইন ফাংশন **এনুমেরেট**। এই ফাংশনটি একটি ইটারেবল গ্রহণ করে এবং একটি ইটারেটর তৈরি করে যা প্রতিটি এলিমেন্টের জন্য, এলিমেন্টের ইনডেক্স এবং ভ্যালু ধারণকারী একটি তালিকা তৈরি করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[15] তে: `রঙ = ['লাল', 'কমলা', 'হলুদ']`

[16] তে: তালিকা (গণনা (রঙ))

আউট[16]: [(0, 'লাল'), (1, 'কমলা'), (2, 'হলুদ')]

একইভাবে বিল্ট-ইন ফাংশন **tuple** একটি ক্রম থেকে একটি tuple তৈরি করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[17] তে: `tuple(enumerate(colors))`

আউট[17]: [(0, 'লাল'), (1, 'কমলা'), (2, 'হলুদ'))

নিম্নলিখিত for লুপটি enumerate দ্বারা ফেরত আসা প্রতিটি টুপলকে ভেরিয়েবল সূচক এবং মানের মধ্যে আনপ্যাক করে এবং সেগুলি প্রদর্শন করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```

...:           মুদ্রণ (f'{সূচক}: {মান}')
...:
0: লাল
1: কমলা
2: হলুদ

```

একটি আদিম বার চার্ট তৈরি করা

নিম্নলিখিত স্ক্রিপ্টটি একটি আদিম **বার চার্ট** তৈরি করে যেখানে প্রতিটি বারের দৈর্ঘ্য তারকাচিহ্ন (*) দিয়ে তৈরি এবং তালিকার সংশ্লিষ্ট উপাদান মানের সমানুপাতিক। তালিকার সূচক এবং মানগুলি নিরাপদে অ্যাক্সেস করতে আমরা enumerate ফাংশনটি ব্যবহার করি। এই উদাহরণটি চালানোর জন্য, এই অধ্যায়ের ch05 উদাহরণ ফোল্ডারে পরিবর্তন করুন, তারপর লিখুন:

আইপাইথন fig05_01.py

অথবা, যদি আপনি ইতিমধ্যেই IPython-এ থাকেন, তাহলে কমান্ডটি ব্যবহার করুন:

fig05_01.py চালান

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```

1 # fig05_01.py ১ """একটি
2   বার চার্ট প্রদর্শন করা হচ্ছে"""
3   ৩টি সংখ্যা = [১৯, ৩, ১৫, ৭, ১১]
4
5   ৫টি প্রিন্ট("\nসংখ্যা থেকে একটি বার চার্ট তৈরি করা:")
6   ৬টি প্রিন্ট(f'Index{"Value":>৮} বার')
7
8
9
10  সূচকের জন্য ৮ , enumerate(numbers) এ মান:
11  ৯    মুদ্রণ(f'{সূচক:>৫}{মান:>৮} {"*"* * মান}')

```

কোড ইমেজ দেখতে এখানে ক্লিক করুন

সংখ্যা থেকে একটি বার চার্ট তৈরি করা:

সূচক মান বার

0	19 *****
1	3 ***
2	15 *****
3	7 *****
8	11 *****

for স্টেটমেন্টটি প্রতিটি উপাদানের সূচক এবং মান পেতে enumerate ব্যবহার করে, তারপর

“*” * মান

মান তারকাচিহ্ন দিয়ে গঠিত একটি স্ট্রিং তৈরি করে। যখন একটি সিকোয়েল্সের সাথে ব্যবহার করা হয়, তখন গুণন অপারেটর (*) সিকোয়েল্সটি পুনরাবৃত্তি করে—এই ক্ষেত্রে, “*” স্ট্রিং—মান বার। এই অধ্যায়ের পরে, আমরা একটি প্রকাশনার গুণমান বার চার্ট ভিজুয়ালাইজেশন প্রদর্শনের জন্য ওপেনসোর্স সিবর্ন এবং ম্যাটপ্লটলিব লাইব্রেরি ব্যবহার করব।

৫.৫ সিকোয়েল্স স্লাইসিং

আপনি মূল উপাদানগুলির উপসেট ধারণকারী একই ধরণের নতুন ক্রম তৈরি করতে ক্রমগুলি স্লাইস করতে পারেন। স্লাইস অপারেশনগুলি পরিবর্তনযোগ্য ক্রমগুলি পরিবর্তন করতে পারে - যেগুলি ক্রম পরিবর্তন করে না সেগুলি তালিকা, টিপল এবং স্ট্রিংয়ের জন্য একইভাবে কাজ করে।

শুরু এবং শেষ সূচক সহ একটি স্লাইস নির্দিষ্ট করা

চলুন তালিকার সূচক ২ থেকে ৫ পর্যন্ত উপাদানগুলি নিয়ে একটি স্লাইস তৈরি করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: সংখ্যা = [2, 3, 5, 7, 11, 13, 17, 19]

[2]: সংখ্যা [2:6]

আউট[2]: [5, 7, 11, 13]

স্লাইসটি কোলনের বাম দিকের শুরুর সূচক (2) থেকে কোলনের ডানদিকের শেষ সূচক (6) পর্যন্ত উপাদানগুলি অনুলিপি করে, কিন্তু অন্তর্ভুক্ত করে না। মূল তালিকাটি পরিবর্তিত হয় না।

শুধুমাত্র একটি শেষ সূচক সহ একটি স্লাইস নির্দিষ্ট করা

যদি আপনি শুরুর সূচক বাদ দেন, তাহলে ধরে নেওয়া হবে 0। সুতরাং, স্লাইস সংখ্যাগুলি [:6] এর সমতুল্য স্লাইস সংখ্যা [0:6]:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3]: সংখ্যা[:6]

আউট[3]: [2, 3, 5, 7, 11, 13]

[4] তে: সংখ্যা [0:6]

শুধুমাত্র একটি প্রারম্ভিক সূচক সহ একটি স্লাইস নির্দিষ্ট করা

যদি আপনি শেষের সূচীটি বাদ দেন, তাহলে পাইথন ক্রমের দৈর্ঘ্য ধরে নেবে (এখানে 8), তাই স্লিপেট [5] এর স্লাইসে সূচক 6 এবং 7-এ সংখ্যার উপাদান রয়েছে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[5]: সংখ্যা[6:]

আউট[5]: [17, 19]

[6]: সংখ্যা[6:len(সংখ্যা)]

আউট[6]: [17, 19]

কোণও সূচক ছাড়াই একটি স্লাইস নির্দিষ্ট করা

শুরু এবং শেষ উভয় সূচক বাদ দিলে পুরো ক্রমটি অনুলিপি হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[7]: সংখ্যা[:]

আউট[7]: [2, 3, 5, 7, 11, 13, 17, 19]

যদিও স্লাইসগুলি নতুন অবজেক্ট তৈরি করে, স্লাইসগুলি উপাদানগুলির **অগভীর অনুলিপি** তৈরি করে — অর্থাৎ, তারা উপাদানগুলির রেফারেন্সগুলি অনুলিপি করে কিন্তু তারা যে বস্তুগুলির দিকে নির্দেশ করে তা নয়। সুতরাং, উপরের স্লিপেটে, নতুন তালিকার উপাদানগুলি মূল তালিকার উপাদানগুলির মতো একই বস্তুগুলিকে উল্লেখ করে, আলাদা কপি করার পরিবর্তে। “ArrayOriented Programming with NumPy” অধ্যায়ে, আমরা deep copying ব্যাখ্যা করব, যা আসলে referenced objects গুলিকে নিজেরাই কপি করে, এবং আমরা নির্দেশ করব কখন deep copying পছন্দ করা হয়।

ধাপে ধাপে কাটা

নিচের কোডটি প্রতিটি অন্যান্য উপাদানের সাথে একটি স্লাইস তৈরি করতে 2 ধাপ ব্যবহার করে সংখ্যা:

[8]: সংখ্যা[::-2]

আউট[8]: [2, 5, 11, 17]

আমরা শুরু এবং শেষ সূচকগুলি বাদ দিয়েছি, তাই যথাক্রমে 0 এবং len(numbers) ধরে নেওয়া হয়েছে।

নেতিবাচক সূচক এবং ধাপগুলি দিয়ে কাটা

বিপরীত ক্রমে স্লাইস নির্বাচন করার জন্য আপনি একটি নেতিবাচক ধাপ ব্যবহার করতে পারেন। নিম্নলিখিত কোডটি সংক্ষেপে বিপরীত ক্রমে একটি নতুন তালিকা তৈরি করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[9]: সংখ্যা[::1]

আউট[9]: [19, 17, 13, 11, 7, 5, 3, 2]

এটি এর সমতুল্য:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[10] তে: সংখ্যা [1:9:1]

আউট[10]: [19, 17, 13, 11, 7, 5, 3, 2]

স্লাইস ব্যবহার করে তালিকা পরিবর্তন করা

আপনি একটি তালিকার একটি অংশ নির্দিষ্ট করে পরিবর্তন করতে পারেন—তালিকার বাকি অংশ অপরিবর্তিত থাকবে। নিম্নলিখিত কোডটি সংখ্যার প্রথম তিনটি উপাদান প্রতিস্থাপন করে, বাকিগুলি অপরিবর্তিত রাখে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[11] তে: সংখ্যা [0:3] = ['দুই', 'তিন', 'পাঁচ']

[12] তে: সংখ্যা

আউট[12]: ['দুই', 'তিন', 'পাঁচ', 7, 11, 13, 17, 19]

নিম্নলিখিতটি একটি খালি স্থান নির্ধারণ করে সংখ্যার শুধুমাত্র প্রথম তিনটি উপাদান মুছে ফেলে তিনটি উপাদানের স্লাইসের তালিকা:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[13] তে: সংখ্যা [0:3] = []

[14] তে: সংখ্যা

আউট[14]: [7, 11, 13, 17, 19]

নিম্নলিখিতটি একটি তালিকার উপাদানগুলিকে সংখ্যার অন্যান্য প্রতিটি উপাদানের একটি অংশে বরাদ্দ করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[15] তে: সংখ্যা = [2, 3, 5, 7, 11, 13, 17, 19]

[16] তে: সংখ্যা [:2] = [100, 100, 100, 100]

[17] তে: সংখ্যা

আউট[17]: [100, 3, 100, 7, 100, 13, 100, 19]

[18] তে: আইডি(সংখ্যা)

আউট[18]: 4434456648

বিদ্যমান তালিকাটি খালি রেখে সংখ্যার সমস্ত উপাদান মুছে ফেলা যাক:

কোড ইমেজ দেখতে এখানে [ক্লিক করুন](#)

[19] তে: সংখ্যা[:] = []

[20] তে: সংখ্যা

আউট[20]: []

[21] তে: আইডি(সংখ্যা)

আউট[21]: 4434456648

সংখ্যার বিষয়বস্তু মুছে ফেলা (স্লিপেট [19]) সংখ্যাগুলিকে একটি নতুন খালি তালিকা [] (স্লিপেট [22]) বরাদ্দ করার থেকে আলাদা। এটি প্রমাণ করার জন্য, আমরা প্রতিটি ক্রিয়াকলাপের পরে সংখ্যার পরিচয় প্রদর্শন করি। পরিচয়গুলি ভিন্ন, তাই তারা মেমরিতে পৃথক বস্তু উপস্থাপন করে:

[22] তে: সংখ্যা = []

[23] তে: সংখ্যা

আউট[23]: []

[24] তে: আইডি(সংখ্যা)

আউট[24]: 4406030920

যখন আপনি একটি ডেরিয়েবলে একটি নতুন অবজেক্ট বরাদ্দ করেন (যেমন স্লিপেট [21]), তখন অন্য কোনও ডেরিয়েবল যদি এটি উল্লেখ না করে তবে মূল অবজেক্টটি আবর্জনা সংগ্রহ করা হবে।

বিবৃতির ৫.৬

ডেল [স্টেটমেন্টটি](#) তালিকা থেকে উপাদানগুলি সরাতে এবং মুছে ফেলার জন্যও ব্যবহার করা যেতে পারে

ইন্টারেক্টিভ সেশন থেকে ডেরিয়েবল। আপনি যেকোনো বৈধ সূচকে উপাদানটি অথবা যেকোনো বৈধ স্লাইস থেকে উপাদান(গুলি) সরাতে পারেন।

তালিকা তৈরি করা, তারপর del ব্যবহার করে এর শেষ উপাদানটি সরানো:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: সংখ্যা = তালিকা (পরিসর (0, 10))

[2] তে: সংখ্যা

আউট[2]: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]

[3]: ডেল সংখ্যা[1]

[4] তে: সংখ্যা

আউট[4]: [0, 1, 2, 3, 4, 5, 6, 7, 8]

তালিকা থেকে একটি স্লাইস মুছে ফেলা হচ্ছে

নিম্নলিখিতটি তালিকার প্রথম দুটি উপাদান মুছে ফেলে:

[5]: ডেল সংখ্যা [0:2]

[6] তে: সংখ্যা

আউট[6]: [2, 3, 4, 5, 6, 7, 8]

সম্পূর্ণ তালিকা থেকে অন্যান্য প্রতিটি উপাদান মুছে ফেলার জন্য নিম্নলিখিতটি স্লাইসের একটি ধাপ ব্যবহার করে:

[7]: ডেল সংখ্যা[::2]

[8] তে: সংখ্যা

আউট[8]: [3, 5, 7]

সম্পূর্ণ তালিকার প্রতিনিধিত্বকারী একটি স্লাইস মুছে ফেলা হচ্ছে

নিম্নলিখিত কোডটি তালিকার সমস্ত উপাদান মুছে ফেলে:

[9]: ডেল সংখ্যা[:]

[10] তে: সংখ্যা

আউট[10]: []

বর্তমান সেশন থেকে একটি চলক মুছে ফেলা হচ্ছে

del স্টেটমেন্ট যেকোনো ভেরিয়েবল মুছে ফেলতে পারে। ইন্টারেক্ষিভ সেশন থেকে সংখ্যা মুছে ফেলি, তারপর ভেরিয়েবলের মান প্রদর্শন করার চেষ্টা করি, যার ফলে NameError তৈরি হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[11]: ডেল সংখ্যায়

[12] তে: সংখ্যা

নাম ক্রটি

ট্রেসব্যাক (সর্বশেষ কলটি শেষ

<module>() এ ipythonininput12426f8401232b>

> ১টি সংখ্যা

নাম ক্রটি: নাম 'সংখ্যা' সংজ্ঞায়িত করা হয়নি

৫.৭ ফাংশনগুলিতে তালিকা পাস করা

গত অধ্যায়ে, আমরা উল্লেখ করেছি যে সমস্ত বস্তু রেফারেন্স দ্বারা পাস করা হয় এবং একটি অপরিবর্তনীয় বস্তুকে একটি ফাংশন আর্গুমেন্ট হিসাবে পাস করা দেখানো হয়েছে। এখানে, আমরা রেফারেন্সগুলি আরও আলোচনা করব যখন একটি প্রোগ্রাম একটি ফাংশনে একটি পরিবর্তনযোগ্য তালিকা বস্তু পাস করে তখন কী ঘটে তা পরীক্ষা করে।

একটি ফাংশনে একটি সম্পূর্ণ তালিকা প্রেরণ করা

`modify_elements` ফাংশনটি বিবেচনা করুন, যা একটি তালিকার একটি রেফারেন্স প্রহণ করে এবং তালিকার প্রতিটি উপাদানের মানকে 2 দিয়ে গুণ করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: def modify_elements(items):
...:     """আইটেমের সকল উপাদানের মানকে 2 দিয়ে গুণ করে।"""
...:     i এর জন্য range (len(items)):
...:         আইটেম [i] *= ২
...:
```

[2] তে: সংখ্যা = [10, 3, 7, 1, 9]

[3] তে: modify_elements(numbers)

[4] তে: সংখ্যা

আউট[4]: [20, 6, 14, 2, 18]

ফাংশন `modify_elements` এর আইটেম প্যারামিটারটি মূল তালিকার একটি রেফারেন্স পায়, তাই লুপের সুট্টের স্টেটমেন্টটি মূল তালিকার বস্তুর প্রতিটি উপাদানকে পরিবর্তন করে।

একটি ফাংশনে একটি টুপল পাস করা

যখন আপনি একটি ফাংশনে একটি টুপল পাস করেন, তখন টুপলের অপরিবর্তনীয় উপাদানগুলি পরিবর্তন করার চেষ্টা করলে একটি TypeError দেখা দেয়:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[5] তে: সংখ্যা_টুপল = (10, 20, 30)

[6] তে: সংখ্যা_টুপল
আউট[6]: (10, 20, 30)

[7] তে: modify_elements(numbers_tuple)

টাইপ এর                                            ট্রেসব্যাক (সর্বশেষ কলটি শেষ

ipythoninput79339741cd595> <module>() এ
> ১টি পরিবর্তন_উপাদান (সংখ্যা_টুপল)

<ipythoninput127acb8f8f44c> modify_elements(আইটেম) এ
 ২      """আইটেমের সকল উপাদানের মানকে 2 দিয়ে গুণ করে।"""
 ৩      i এর জন্য range (len(items)): items[i] *= 2
> ৪
 ৫
 ৬

TypeError: 'tuple' অবজেক্ট আইটেম অ্যাসাইনমেন্ট সমর্থন করে না
```

মনে রাখবেন যে টিপলগুলিতে পরিবর্তনযোগ্য বস্তু থাকতে পারে, যেমন তালিকা। যখন একটি টিপল একটি ফাংশনে পাস করা হয় তখনও সেই বস্তুগুলি পরিবর্তন করা যেতে পারে।

ট্রেসব্যাক সম্পর্কিত একটি নোট

পূর্ববর্তী ট্রেসব্যাকে দুটি স্লিপেট দেখানো হয়েছে যা TypeError-এর দিকে পরিচালিত করেছিল। প্রথমটি হল স্লিপেট [7]-এর ফাংশন কল। দ্বিতীয়টি হল স্লিপেট [1]-এর ফাংশন সংজ্ঞা। প্রতিটি স্লিপেটের কোডের আগে লাইন নম্বর থাকে। আমরা বেশিরভাগ সিঙ্গেললাইন স্লিপেট প্রদর্শন করেছি।

যখন এই ধরনের স্লিপেটে ব্যতিক্রম দেখা দেয়, তখন সর্বদা > 1 এর আগে থাকে, যা নির্দেশ করে যে লাইন 1 (স্লিপেটের একমাত্র লাইন) ব্যতিক্রমটির কারণ। `modify_elements` এর সংজ্ঞার মতো মাল্টিলাইন স্লিপেটগুলি 1 থেকে শুরু করে ধারাবাহিক লাইন সংখ্যা দেখায়।

উপরের > 4 নম্বর নোটেশনটি নির্দেশ করে যে ব্যতিক্রমটি `modify_elements` এর লাইন 4-এ ঘটেছে। ট্রেসব্যাক যত দীর্ঘই হোক না কেন, $>$ সহ কোডের শেষ লাইনটি ব্যতিক্রমটি তৈরি করেছে।

৫.৮ তালিকা বাছাইকরণ

সাজানোর মাধ্যমে আপনি ডেটা উর্ধ্বমুখী বা অবরোহী ক্রমে সাজাতে পারবেন।

উর্ধক্রম অনুসারে তালিকা সাজানো

তালিকা পদ্ধতির মাধ্যমে তালিকার উপাদানগুলিকে উর্ধক্রমানুসারে সাজানোর জন্য তালিকা পরিবর্তন করা হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: সংখ্যা = [10, 3, 7, 1, 9, 4, 2, 8, 5, 6]

[2] তে: numbers.sort()

[3] তে: সংখ্যা

আউট[3]: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

তালিকাটি অবরোহ ক্রমে সাজানো

একটি তালিকাকে অবরোহী ক্রমে সাজানোর জন্য, এছিক কীওয়ার্ড আর্গমেন্টের বিপরীতে True (False হল ডিফল্ট) সেট করে তালিকা পদ্ধতিকে সাজানোর জন্য কল করুন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: numbers.sort(reverse=True)

[5] তে: সংখ্যা

আউট[5]: [10, 9, 8, 7, 6, 5, 4, 3, 2, 1]

বিল্ট-ইন ফাংশন সাজানো হয়েছে

বিল্ট-ইন ফাংশন sorted তার আর্গমেন্ট সিকোয়েলের সাজানো উপাদানগুলি ধারণ করে একটি নতুন তালিকা প্রদান করে —মূল সিকোয়েলটি অপরিবর্তিত। নিম্নলিখিত কোডটি একটি তালিকা, একটি স্ট্রিং এবং একটি টিপলের জন্য সাজানো ফাংশন প্রদর্শন করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6] তে: সংখ্যা = [10, 3, 7, 1, 9, 4, 2, 8, 5, 6]

[7] তে: ascending_numbers = sorted(numbers)

[8] তে: ascending_numbers

আউট[8]: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

[9] তে: সংখ্যা

আউট[9]: [10, 3, 7, 1, 9, 4, 2, 8, 5, 6]

[10] তে: অক্ষর = 'fadgchjebi'

[11] তে: ascending_letters = sorted(letters) www.EBooksWorld.ir

[12] তে: ascending_letters
আউট[12]: ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i', 'j']

[13] তে: অক্ষর
আউট[13]: 'ফাদগচজেবি'

[14] তে: রঙ = ('লাল', 'কমলা', 'হলুদ', 'সবুজ', 'নীল')

[15] তে: ascending_colors = sorted(colors)

[16] তে: ascending_colors
আউট[16]: ['নীল', 'সবুজ', 'কমলা', 'লাল', 'হলুদ']

[17] তে: রঙ
আউট[17]: ('লাল', 'কমলা', 'হলুদ', 'সবুজ', 'নীল')

উপাদানগুলিকে অবরোহী ক্রমে সাজানোর জন্য ঐচ্ছিক কীওয়ার্ড আর্গুমেন্টটি True মান সহ বিপরীত ব্যবহার করুন।

৫.৯ অনুসন্ধানের ক্রম

প্রায়শই, আপনাকে নির্ধারণ করতে হবে যে একটি ক্রম (যেমন একটি তালিকা, টিপল বা স্ট্রিং) একটি নির্দিষ্ট কী মানের সাথে মেলে এমন একটি মান আছে কিনা। **অনুসন্ধান** হল একটি কী সনাক্ত করার প্রক্রিয়া।

তালিকা পদ্ধতি সূচক

তালিকা পদ্ধতি **সূচক** একটি আর্গুমেন্ট হিসেবে একটি অনুসন্ধান কী নেয়—তালিকায় যে মানটি সনাক্ত করতে হবে— তারপর সূচক 0 থেকে তালিকাটি অনুসন্ধান করে এবং অনুসন্ধান কী-এর সাথে মেলে এমন প্রথম উপাদানের সূচক প্রদান করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: সংখ্যা = [3, 7, 1, 4, 2, 8, 5, 6]

[2] তে: numbers.index(5)
আউট[2]: 6

আপনি যে মানটি খুঁজছেন তা যদি তালিকায় না থাকে তবে একটি ValueError ঘটে।

একটি অনুসন্ধানের শুরুর সূচী নির্দিষ্ট করা

মেথড ইনডেক্সের ঐচ্ছিক আর্গুমেন্ট ব্যবহার করে, আপনি একটি তালিকার উপাদানের একটি উপসেট অনুসন্ধান করতে পারেন।
আপনি *= ব্যবহার করে একটি ক্রমকে গুণ করতে পারেন—অর্থাৎ, একটি ক্রমকে নিজের সাথে একাধিক যোগ করতে পারেন

বার। নিম্নলিখিত স্লিপেটের পরে, সংখ্যাগুলিতে মূল তালিকার দুটি কপি থাকে

সূচিপত্র:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3] তে: সংখ্যা *= 2

[4] তে: সংখ্যা

আউট[4]: [3, 7, 1, 4, 2, 8, 5, 6, 3, 7, 1, 4, 2, 8, 5, 6]

নিম্নলিখিত কোডটি আপডেট করা তালিকার মান ৫ অনুসন্ধান করে যা সূচক ৭ থেকে শুরু করে তালিকার শেষ পর্যন্ত অব্যাহত থাকে:

[5] তে: numbers.index(5, 7)

আউট[5]: ১৪

অনুসন্ধানের শুরু এবং শেষের সূচকগুলি নির্দিষ্ট করা

শুরু এবং শেষ সূচকগুলি নির্দিষ্ট করার ফলে সূচকটি শুরুর সূচক থেকে শেষ সূচকের অবস্থান পর্যন্ত অনুসন্ধান করতে পারে কিন্তু অন্তর্ভুক্ত নয়। স্লিপেটে সূচকের কল

[5]:

সংখ্যা.সূচক(৫, ৭)

সংখ্যার দৈর্ঘ্যকে তার ঐচ্ছিক তৃতীয় ঘূর্ণি হিসেবে ধরে নেয় এবং এর সমতুল্য:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

numbers.index(5, 7, len(সংখ্যা))

০ থেকে ৩ সূচক সহ উপাদানগুলির পরিসরে নিম্নলিখিত মান ৭ অনুসন্ধান করা হয়েছে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6] তে: numbers.index(7, 0, 4)

আউট[6]: ১

অপারেটররা ইন এবং ইন নয়

অপারেটর পরীক্ষা করে যে তার ডান অপারেটের পুনরাবৃত্তিযোগ্যতিতে বাম অপারেটের মান রয়েছে কিনা:

[7] তে : **সংখ্যায় ১০০০**

আউট[7]: মিথ্যা

[8]: **সংখ্যায় ৫**

আউট[8]: সত্য

একইভাবে, অপারেটর পরীক্ষা করে না যে তার ডান অপারেন্ডের পুনরাবৃত্তিযোগ্যতিতে বাম অপারেন্ডের মান নেই কিনা:

[9] তে: **১০০০ সংখ্যায় নয়**

আউট[9]: সত্য

[10] তে: **৫ সংখ্যায় নয়**

আউট[10]: মিথ্যা

ValueError প্রতিরোধ করতে অপারেটর in ব্যবহার করা

মেথড ইনডেক্সে কল করার ফলে যাতে কোনও সমস্যা না হয় তা নিশ্চিত করতে আপনি in অপারেটর ব্যবহার করতে পারেন।

সংশ্লিষ্ট ক্রমানুসারে না থাকা অনুসন্ধান কীণ্ডলির জন্য ValueErrors:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[11] তে: কী = **1000**

[12] তে: যদি সংখ্যায় কী থাকে :

...: মুদ্রণ করুন (f'found {key} at index {numbers.index(search_key)}')

...: অন্যথায়:

...: মুদ্রণ (f'{key} পাওয়া যায়নি')

...:

১০০০ পাওয়া যায়নি



বিল্ট-ইন ফাংশন যেকোনো এবং সকল

কখনও কখনও আপনার কেবল জানতে হবে যে একটি পুনরাবৃত্তের মধ্যে কোনও আইটেম সত্য কিনা নাকি সমস্ত আইটেম সত্য। বিল্ট-ইন ফাংশন **any** যদি তার পুনরাবৃত্ত আর্গমেন্ট কোনও আইটেম সত্য হয় তবে True প্রদান করে। বিল্ট-ইন ফাংশন **all** যদি তার পুনরাবৃত্ত আর্গমেন্ট সমস্ত আইটেম সত্য হয় তবে True প্রদান করে। মনে রাখবেন যে অ-শূন্য মানগুলি সত্য এবং 0 হল মিথ্যা। খালি পুনরাবৃত্ত বস্তুগুলি সত্য হিসাবে মূল্যায়ন করে, যেখানে কোনও খালি পুনরাবৃত্ত মিথ্যা হিসাবে মূল্যায়ন করে। ফাংশন **any** এবং **all** হল ফাংশনালস্টাইল প্রোগ্রামিংয়ে অভ্যন্তরীণ পুনরাবৃত্তির অতিরিক্ত উদাহরণ।

৫.১০ অন্যান্য তালিকা পদ্ধতি

তালিকাগুলিতে উপাদান যোগ এবং অপসারণের পদ্ধতিও রয়েছে। তালিকাটি বিবেচনা করুন

রঙের নাম:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: color_names = ['কমলা', 'হলুদ', 'সবুজ']
```

একটি নির্দিষ্ট তালিকা সূচীতে একটি উপাদান সন্নিবেশ করানো

মেথড ইনসার্ট একটি নির্দিষ্ট সূচীতে একটি নতুন আইটেম যোগ করে। নিম্নলিখিতটি 'লাল' এ সন্নিবেশ করায়

সূচক ০:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[2] তে: color_names.insert(0, 'red')
```

```
[3] তে: color_names
```

আউট[3]: ['লাল', 'কমলা', 'হলুদ', 'সবুজ']

তালিকার শেষে একটি উপাদান যোগ করা

আপনি তালিকার শেষে একটি নতুন আইটেম যোগ করতে পারেন method **append** ব্যবহার করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[4] তে: color_names.append('blue')
```

```
[5] তে: রঙ_নাম
```

আউট[5]: ['লাল', 'কমলা', 'হলুদ', 'সবুজ', 'নীল']

তালিকার শেষে একটি ক্রমের সমস্ত উপাদান যোগ করা

তালিকার শেষে অন্য ক্রমের সমস্ত উপাদান যোগ করতে list পদ্ধতি **extend** ব্যবহার করুন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[6] তে: color_names.extend(['নীল', 'ভায়োলেট'])
```

```
[7] তে: রঙ_নাম
```

আউট[7]: ['লাল', 'কমলা', 'হলুদ', 'সবুজ', 'নীল',

'নীল', 'বেগুনি']

এটি += ব্যবহারের সমতুল্য। নিচের কোডটি একটি স্ট্রিং-এর সমস্ত অক্ষর যোগ করে তারপর একটি তালিকায় একটি টুপলের সমস্ত উপাদান যোগ করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[8] তে: নমুনা_তালিকা = []

[9] তে: s = 'abc'

[10] তে: sample_list.extend(s)

[11] তে: নমুনা_তালিকা

আউট[11]: ['a', 'b', 'c']

[12] তে: t = (1, 2, 3)

[13] তে: sample_list.extend(t)

[14] তে: নমুনা_তালিকা

আউট[14]: ['a', 'b', 'c', 1, 2, 3]

একটি তালিকায় যোগ করার আগে একটি টুপল সংরক্ষণ করার জন্য t এর মতো একটি অস্থায়ী চলক তৈরি করার পরিবর্তে, আপনি extend করার জন্য সরাসরি একটি টুপল পাস করতে চাইতে পারেন। এই ক্ষেত্রে, টুপলের বন্ধনী প্রয়োজন, কারণ extend একটি পুনরাবৃত্তিযোগ্য যুক্তি আশা করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[15]: sample_list.extend((4, 5, 6)) # অতিরিক্ত বন্ধনীগুলি লক্ষ্য করুন

[16] তে: নমুনা_তালিকা

আউট[16]: ['a', 'b', 'c', 1, 2, 3, 4, 5, 6]

প্রয়োজনীয় বন্ধনী বাদ দিলে একটি TypeError দেখা দেয়।

তালিকার একটি উপাদানের প্রথম ঘটনা অপসারণ

মেথড **রিমুভ** একটি নির্দিষ্ট মান সহ প্রথম উপাদানটি মুছে ফেলে - যদি রিমুভের আর্গুমেন্ট তালিকায় না থাকে তবে একটি ValueError ঘটে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[17] তে: color_names.remove('green')

[18]-এ: color_names Out[18]: ['লাল', 'কমলা', 'হনুম',

'নীল', 'নীল', www.EBooksWorld.ir

'বেগুনি']

একটি তালিকা খালি করা হচ্ছে

তালিকার সমস্ত উপাদান মুছে ফেলার জন্য, **clear** পদ্ধতিটি কল করুন:

```
[19] তে: color_names.clear()
```

```
[20] তে: color_names
```

```
আউট[20]: []
```

এটি পূর্বে দেখানো স্লাইস অ্যাসাইনমেন্টের সমতুল্য।

```
রঙের_নাম[:] = []
```

একটি আইটেমের ঘটনার সংখ্যা গণনা করা

List method **count** তার আগমেন্ট অনুসন্ধান করে এবং এটি কতবার করা হয়েছে তার সংখ্যা প্রদান করে

পাওয়া গেছে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[21] তে: প্রতিক্রিয়া = [1, 2, 5, 4, 3, 5, 2, 1, 3, 3,
```

```
...: 1, 8, 3, 3, 3, 2, 3, 3 , 2, 2]
```

```
...:
```

```
[22] তে: i এর জন্য range (1, 6):
```

```
...: print(f'{i} প্রতিক্রিয়াগুলিতে {responses.count(i)} বার প্রদর্শিত হয় '
```

```
...:
```

উত্তরগুলিতে ১টি ৩ বার প্রদর্শিত হয়েছে

প্রতিক্রিয়াগুলিতে ২টি ৫ বার প্রদর্শিত হয়েছে

উত্তরগুলিতে ৩টি ৮ বার প্রদর্শিত হয়েছে

প্রতিক্রিয়াগুলিতে ৪টি ২ বার প্রদর্শিত হয়েছে

প্রতিক্রিয়াগুলিতে ৫টি ২ বার প্রদর্শিত হয়েছে

তালিকার উপাদানগুলিকে উল্টানো

তালিকা পদ্ধতি **বিপরীত** একটি বিপরীত অনুলিপি তৈরি করার পরিবর্তে, তালিকার বিষয়বস্তুগুলিকে বিপরীত করে, যেমনটি আমরা আগে একটি স্লাইসের সাথে করেছি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[23] তে: color_names = ['লাল', 'কমলা', 'হলুদ', 'সবুজ', 'নীল']
```

[24] তে: color_names.reverse()

[25] তে: color_names

আউট[25]: ['নীল', 'সবুজ', 'হলুদ', 'কমলা', 'লাল']

একটি তালিকা অনুলিপি করা হচ্ছে

তালিকা পদ্ধতির অনুলিপি মূল তালিকার একটি অগভীর অনুলিপি ধারণকারী একটি নতুন তালিকা প্রদান করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[26] তে: copied_list = color_names.copy()

[27] তে: কপি_লিস্ট

আউট[27]: ['নীল', 'সবুজ', 'হলুদ', 'কমলা', 'লাল']

এটি পূর্বে প্রদর্শিত স্লাইস অপারেশনের সমতুল্য:

কপি করা_তালিকা = রঙের_নাম[:]

৫.১১ তালিকার সাথে স্ট্যাক সিমুলেট করা

পূর্ববর্তী অধ্যায়ে ফাংশনকল স্ট্যাক সম্পর্কে আলোচনা করা হয়েছে। পাইথনে কোনও বিল্ট-ইন স্ট্যাক টাইপ নেই, তবে আপনি স্ট্যাককে একটি সীমাবদ্ধ তালিকা হিসাবে ভাবতে পারেন। আপনি তালিকা পদ্ধতি অ্যাপেন্ড ব্যবহার করে পুশ করেন, যা তালিকার শেষে একটি নতুন উপাদান যুক্ত করে। আপনি তালিকা পদ্ধতি পপ ব্যবহার করে কোনও আর্গমেন্ট ছাড়াই **পপ করেন**, যা শেষে আইটেমটি সরিয়ে দেয় এবং ফেরত দেয়।

তালিকা।

চলুন stack নামক একটি খালি তালিকা তৈরি করি, এতে দুটি স্ট্রিং পুশ (সংযোজন) করি, তারপর স্ট্রিংগুলিকে পপ করে নিশ্চিত করি যে সেগুলি lastin, firstout (LIFO) ক্রমে পুনরুদ্ধার করা হয়েছে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

n [1]: স্ট্যাক = []

[2] তে: stack.append('red')

[3] তে: স্ট্যাক

আউট[3]: ['লাল']

[4] তে: stack.append('green')

[5] তে: স্ট্যাক

আউট[5]: ['লাল', 'সবুজ']

[6] തോറ്റ്: stack.pop()

আউট[6]: 'সবুজ'

[7] ତେ: ସଟ୍ୟାକ

আউট[7]: ['লাল']

[8] തുറന്നു: stack.pop()

ଆଉଟ[8]: 'ଲାଲ'

[9] তে: স্ট্যাক

আউট[9]: []

[10] തുറന്ന് stack.pop()

সূচক ক্রটি

ট্রেসব্যাক (সর্বশেষ কল শেষ)

```
<ipythoninput1050ea7ec13fbe> <module>() এ  
> ১টি স্ট্যাক.পপ()
```

IndexError: খালি তালিকা থেকে পপ করুন

অথবা প্রতিটি পপ স্লিপেটে, পপ যে মানটি সরিয়ে দেয় এবং ফেরত দেয় তা প্রদর্শিত হয়। পপিং

খালি স্ট্যাক থেকে IndexError তৈরি হয়, ঠিক যেমন [] দিয়ে একটি অস্তিত্বহীন তালিকা উপাদান অ্যাক্সেস করা হল।

IndexError প্রতিরোধ করতে, pop কল করার আগে len(stack) 0 এর চেয়ে বেশি কিনা তা নিশ্চিত করুন। আপনি যদি আইটেমগুলিকে পপ করার চেয়ে দ্রুত পুশ করতে থাকেন তবে আপনার মেমরি ফুরিয়ে যেতে পারে।

আপনি আরেকটি জনপ্রিয় সংগ্রহের অনুকরণ করার জন্য একটি তালিকা ব্যবহার করতে পারেন যাকে **কিউ** বলা হয় যেখানে আপনি পিছনে সন্নিবেশ করান এবং সামনে থেকে মুছে ফেলুন। আইটেমগুলি কিউ থেকে **firstin, firstout (FIFO)** ক্রমে পুনরুদ্ধার করা হয়।

৫.১২ তালিকাভুক্তির ধারণা

এখানে, আমরা **তালিকার বোধগম্যতার** সাথে কার্যকরী শৈলীর বৈশিষ্ট্যগুলি নিয়ে আলোচনা চালিয়ে যাচ্ছি - নতুন তালিকা তৈরির জন্য একটি সংক্ষিপ্ত এবং সুবিধাজনক স্বরলিপি। তালিকার বোধগম্যতা এমন অনেক বিবৃতি প্রতিস্থাপন করতে পারে যা বিদ্যমান ক্রমগুলির উপর পুনরাবৃত্তি করে এবং নতুন তালিকা তৈরি করে, যেমন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: তালিকা1 = [

[2] তে: **রেঞ্জের আইটেমের জন্য** (1, 6):

তালিকা 1.অ্যাপেন্ড(আইটেম)

[3] তে: তালিকা ১

পূর্ণসংখ্যার তালিকা তৈরি করতে তালিকা বোঝার পদ্ধতি ব্যবহার করা

আমরা তালিকা বোঝার মাধ্যমে কোডের একটি লাইনে একই কাজ সম্পন্ন করতে পারি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: list2 = [রেঞ্জে থাকা আইটেমের জন্য আইটেম (1 , 6)]

[5] তে: তালিকা 2

আউট[5]: [1, 2, 3, 4, 5]

বিবৃতির জন্য স্লিপেট [2] এর মতো, তালিকার বোধগম্যতা **ধারার জন্য**

পরিসরে থাকা আইটেমের **জন্য** (১, ৬)

range(1, 6) দ্বারা উৎপাদিত ক্রম ধরে পুনরাবৃত্তি করে। প্রতিটি আইটেমের জন্য, তালিকার বোধগম্যতা for ধারার বাম দিকের অভিব্যক্তিটি মূল্যায়ন করে এবং অভিব্যক্তির মান (এই ক্ষেত্রে, আইটেমটি নিজেই) নতুন তালিকায় রাখে। স্লিপেট [4] এর বিশেষ বোধগম্যতা ফাংশন তালিকা ব্যবহার করে আরও সংক্ষিপ্তভাবে প্রকাশ করা যেত:

তালিকা2 = তালিকা(পরিসর(১, ৬))

ম্যাপিং: একটি তালিকা বোঝার অভিব্যক্তিতে ক্রিয়াকলাপ সম্পাদন করা

একটি তালিকার বোধগম্যতার অভিব্যক্তি গণনার মতো কাজ সম্পাদন করতে পারে, যা উপাদানগুলিকে নতুন মানের (সম্ভবত বিভিন্ন ধরণের) সাথে **ম্যাপ** করে। ম্যাপিং হল একটি সাধারণ কার্যকরী-শৈলীর প্রোগ্রামিং অপারেশন যা মূল ডেটা ম্যাপ করা উপাদানের সমান সংখ্যক ফলাফল তৈরি করে। নিম্নলিখিত বোধগম্যতা প্রতিটি মানকে তার ঘনকের সাথে এক্সপ্রেশন আইটেমের সাথে ম্যাপ করে।

** ৩:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6] তে: list3 = [রেঞ্জে থাকা আইটেমের **জন্য** আইটেম ** 3]

[7] তে: তালিকা 3

আউট[7]: [1, 8, 27, 64, 125]

ফিল্টারিং: যদি ধারা সহ বোধগম্যতা তালিকাভুক্ত করুন

আরেকটি সাধারণ ফাংশনাল স্টাইল প্রোগ্রামিং অপারেশন হল উপাদানগুলিকে ফিল্টার করা যাতে শুধুমাত্র শর্তের সাথে মেলে এমনগুলি নির্বাচন করা যায়। এটি সাধারণত ফিল্টার করা ডেটার চেয়ে কম উপাদান সহ একটি তালিকা তৈরি করে। তালিকা বোঝার ক্ষেত্রে এটি করার জন্য, if ধারাটি ব্যবহার করুন। নিম্নলিখিতটিতে list4-এ শুধুমাত্র for দ্বারা উৎপাদিত জোড় মান অন্তর্ভুক্ত রয়েছে।

ধারা:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[8] তে: list4 = [ যদি আইটেম % 2 == 0 হয় তবে range(1, 11) এর আইটেমের জন্য আইটেম]
```

```
[9] তে: তালিকা 4
```

```
আউট[9]: [2, 4, 6, 8, 10]
```

তালিকার বোধগম্যতা যা অন্য তালিকার উপাদানগুলিকে প্রক্রিয়া করে

for ক্লজটি যেকোনো পুনরাবৃত্তিযোগ্য স্ট্রিং প্রক্রিয়া করতে পারে। আসুন ছোট হাতের স্ট্রিংগুলির একটি তালিকা তৈরি করি এবং তাদের বড় হাতের সংস্করণগুলি ধারণ করে একটি নতুন তালিকা তৈরি করতে একটি তালিকা বোধগম্যতা ব্যবহার করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[10] তে: রঙ = ['লাল', 'কমলা', 'হলুদ', 'সবুজ', 'নীল']
```

```
[11] তে: colors2 = [ রঙের আইটেমের জন্য item.upper() ]
```

```
[12] তে: colors2
```

```
আউট[12]: ['লাল', 'কমলা', 'হলুদ', 'সবুজ', 'নীল']
```

```
[13] তে: রঙ
```

```
আউট[13]: ['লাল', 'কমলা', 'হলুদ', 'সবুজ', 'নীল']
```

৫.১৩ জেনারেটর এক্সপ্রেশন

একটি জেনারেটর এক্সপ্রেশন একটি তালিকা বোধগম্যতার অনুরূপ, তবে এটি একটি পুনরাবৃত্তিযোগ্য জেনারেটর অবজেক্ট তৈরি করে যা চাহিদা অনুসারে মান তৈরি করে। এটি অলস মূল্যায়ন হিসাবে পরিচিত। তালিকা বোধগম্যতা লোভী মূল্যায়ন ব্যবহার করে - তারা তালিকাগুলি কার্যকর করার সাথে সাথেই তৈরি করে। প্রচুর সংখ্যক আইটেমের জন্য, একটি তালিকা তৈরি করতে যথেষ্ট মেমরি এবং সময় লাগতে পারে। তাই জেনারেটর এক্সপ্রেশন আপনার প্রোগ্রামের মেমরি খরচ কমাতে পারে এবং যদি একবারে পুরো তালিকাটি প্রয়োজন না হয় তবে কর্মক্ষমতা উন্নত করতে পারে।

জেনারেটর এক্সপ্রেশনের তালিকা বোঝার মতোই ক্ষমতা রয়েছে, তবে আপনি সেগুলিকে বর্গকার বক্ষনীর পরিবর্তে বক্ষনীতে সংজ্ঞায়িত করতে পারেন। স্নিপেট [2]-এ জেনারেটর এক্সপ্রেশনটি বর্গক্ষেত্রে এবং সংখ্যায় শুধুমাত্র বিজোড় মানগুলি প্রদান করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: সংখ্যা = [10, 3, 7, 1, 9, 4, 2, 8, 5, 6]

[2] তে: মানের জন্য ($x \text{ ** 2 } \text{ সংখ্যায় } x \text{ এর জন্য যদি } x \% 2 != 0 \text{ হয় }$):

...: মুদ্রণ (মান, শেষ = ')

...:

৯ ৪৯ ১ ৮১ ২৫

একটি জেনারেটর এক্সপ্রেশন একটি তালিকা তৈরি করে না তা দেখানোর জন্য, আসুন পূর্ববর্তী স্লিপেটের জেনারেটর এক্সপ্রেশনটি একটি ভেরিয়েবলের সাথে বরাদ্দ করি এবং ভেরিয়েবলটি মূল্যায়ন করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3] তে: square_of_odds = ($x \text{ ** 2 } \text{ সংখ্যায় } x \text{ এর জন্য যদি } x \% 2 != 0 \text{ হয় }$)

[3] তে: square_of_odds

আউট[3]: <generator object <genexpr> at 0x1085e84c0>

"জেনারেটর অবজেক্ট <genexpr>" লেখাটি নির্দেশ করে যে square_of_odds হল একটি জেনারেটর অবজেক্ট যা একটি জেনারেটর এক্সপ্রেশন (genexpr) থেকে তৈরি করা হয়েছে।

৫.১৪ ফিল্টার, মানচিত্র এবং হ্রাস

পূর্ববর্তী বিভাগে বেশ কয়েকটি কার্যকরী শৈলীর বৈশিষ্ট্য উপস্থাপন করা হয়েছে — তালিকা বোঝা, ফিল্টারিং এবং ম্যাপিং। এখানে আমরা যথাক্রমে ফিল্টারিং এবং ম্যাপিংয়ের জন্য অন্তর্নির্মিত ফিল্টার এবং মানচিত্র ফাংশনগুলি প্রদর্শন করি। আমরা সেই হ্রাসগুলি নিয়ে আলোচনা চালিয়ে যাচ্ছি যেখানে আপনি উপাদানগুলির একটি সংগ্রহকে একটি একক মানে প্রক্রিয়া করেন, যেমন তাদের গণনা, মোট, পণ্য, গড়, সর্বনিম্ন বা সর্বোচ্চ।

বিল্ট-ইন ফিল্টার ফাংশন ব্যবহার করে একটি সিকোয়েলের মান ফিল্টার করা

সংখ্যার বিজোড় মান পেতে বিল্টইন ফাংশন ফিল্টার ব্যবহার করা যাক :

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: সংখ্যা = [10, 3, 7, 1, 9, 4, 2, 8, 5, 6]

[2]: def is_odd(x): """শুধুমাত্র x বিজোড়

...: হলেই সত্য ফেরত আসে!"""

...: $x \% 2 != 0$ প্রদান করুন

...:

[3] তে: তালিকা (ফিল্টার (বিজোড়_, সংখ্যা))

আউট[3]: [3, 7, 1, 9, 5]

ডেটার মতো, পাইথন ফাংশন হল এমন বস্তু যা আপনি ভেরিয়েবলে বরাদ্দ করতে পারেন, অন্য ভেরিয়েবলে পাস করতে পারেন ফাংশন এবং ফাংশন থেকে রিটার্ন। যে ফাংশনগুলি অন্যান্য ফাংশন গ্রহণ করে যেমন

আর্গুমেন্ট হল একটি ফাংশনাল স্টাইল ক্যাপাবিলিটি যাকে **হাইঅর্ডার ফাংশন বলা হয়**। উদাহরণস্বরূপ, ফিল্টারের প্রথম আর্গুমেন্টটি এমন একটি ফাংশন হতে হবে যা একটি আর্গুমেন্ট গ্রহণ করে এবং ফলাফলে মান অন্তর্ভুক্ত করা উচিত হলে True প্রদান করে। ফাংশন is_odd যদি এর আর্গুমেন্ট বিজোড় হয় তবে True প্রদান করে। ফিল্টার ফাংশনটি তার দ্বিতীয় আর্গুমেন্টের পুনরাবৃত্তিযোগ্য (সংখ্যা) প্রতিটি মানের জন্য is_odd একবার কল করে। হাইঅর্ডার ফাংশনগুলি একটি রিটার্নও করতে পারে ফলে কাজ করে।

ফাংশন ফিল্টার একটি ইটারেটর ফেরত দেয়, তাই ফিল্টারের ফলাফলগুলি আপনি যতক্ষণ না সেগুলি পুনরাবৃত্তি করেন ততক্ষণ পর্যন্ত তৈরি হয় না। এটি অলস মূল্যায়নের আরেকটি উদাহরণ। স্লিপেট [3]-এ, ফাংশন তালিকা ফলাফলগুলির মধ্য দিয়ে পুনরাবৃত্তি করে এবং সেগুলি ধারণ করে একটি তালিকা তৈরি করে। আমরা একটি if ধারা সহ একটি তালিকা বোধগম্যতা ব্যবহার করে উপরের মতো একই ফলাফল পেতে পারি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: [is_odd (item) হলে সংখ্যায় আইটেমের জন্য আইটেম]

আউট[4]: [3, 7, 1, 9, 5]

ফাংশনের পরিবর্তে ল্যাষ্বড়া ব্যবহার করা

is_odd এর মতো সাধারণ ফাংশনগুলির জন্য যা শুধুমাত্র একটি একক এক্সপ্রেশনের মান প্রদান করে, আপনি একটি ল্যাষ্বড়া এক্সপ্রেশন (অথবা কেবল একটি **ল্যাষ্বড়া**) ব্যবহার করে ফাংশনটি ইনলাইনে সংজ্ঞায়িত করতে পারেন যেখানে এটি প্রয়োজন - সাধারণত যখন এটি অন্য ফাংশনে স্থানান্তরিত হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[5] তে: list(filter(lambda x: x % 2 != 0, সংখ্যা))

আউট[5]: [3, 7, 1, 9, 5]

ফলাফলগুলিকে একটি তালিকায় রূপান্তর করতে এবং সেগুলি প্রদর্শন করতে আমরা এখানে ফাংশন তালিকাতে ফিল্টারের রিটার্ন মান (একটি ইটারেটর) পাস করি।

ল্যাষ্বড়া এক্সপ্রেশন হলো একটি বেনামী ফাংশন—অর্থাৎ, নামবিহীন একটি ফাংশন। ফিল্টার কলে

কোড ইমেজ দেখতে এখানে ক্লিক করুন

ফিল্টার (ল্যাষ্টডা x: x % 2 != 0, সংখ্যা)

প্রথম যুক্তি হল ল্যাষ্টডা

ল্যাষ্টডা x: x % 2 != 0

একটি ল্যাষ্টডা শুরু হয় **ল্যাষ্টডা** কীওয়ার্ড দিয়ে, তারপর থাকে কমা দ্বারা পৃথক করা প্যারামিটার তালিকা, একটি কোলন (:) এবং একটি এক্সপ্রেশন। এই ক্ষেত্রে, প্যারামিটার তালিকায় x নামে একটি প্যারামিটার থাকে। একটি ল্যাষ্টডা পরোক্ষভাবে তার এক্সপ্রেশনের মান ফেরত দেয়। সুতরাং যেকোনো সহজ ফাংশন ফর্মেট

কোড ইমেজ দেখতে এখানে ক্লিক করুন

ডিফাংশন_নাম (প্যারামিটার_তালিকা):

রিটার্ন এক্সপ্রেশন

ফর্মের আরও সংক্ষিপ্ত ল্যাষ্টডা হিসাবে প্রকাশ করা যেতে পারে

কোড ইমেজ দেখতে এখানে ক্লিক করুন

ল্যাষ্টডা প্যারামিটার_তালিকা: এক্সপ্রেশন

একটি সিকোয়েন্সের মানগুলিকে নতুন মানের সাথে ম্যাপ করা

প্রতিটি মানকে সংখ্যায় বর্গ করার জন্য ল্যাষ্টডা সহ বিল্টইন ফাংশন **ম্যাপ** ব্যবহার করা যাক :

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6] তে: সংখ্যা

আউট[6]: [10, 3, 7, 1, 9, 4, 2, 8, 5, 6]

[7] তে: list(map(lambda x: x

** ২, সংখ্যা))

আউট[7]: [100, 9, 49, 1, 81, 16, 4, 64, 25, 36]

ফাংশন ম্যাপের প্রথম আর্গমেন্ট হল এমন একটি ফাংশন যা একটি মান গ্রহণ করে এবং একটি নতুন মান প্রদান করে—এই ক্ষেত্রে, একটি ল্যাষ্টডা যা তার আর্গমেন্টকে বর্গ করে। দ্বিতীয় আর্গমেন্টটি হল ম্যাপের জন্য মানগুলির একটি পুনরাবৃত্তিযোগ্য। ফাংশন ম্যাপ অলস মূল্যায়ন ব্যবহার করে। সুতরাং, আমরা তালিকা ফাংশনে সেই ইটারেটরটি প্রেরণ করি যা ম্যাপটি ফেরত দেয়। এটি আমাদের পুনরাবৃত্তি করতে এবং ম্যাপ করা মানগুলির একটি তালিকা তৈরি করতে সক্ষম করে। এখানে একটি সমতুল্য তালিকা বোধগম্যতা দেওয়া হল:

কোড ইমেজ দেখতে এখানে লিংক করুন

[8] তে: [সংখ্যায় আইটেমের জন্য আইটেম ** 2]

আউট[8]: [100, 9, 49, 1, 81, 16, 4, 64, 25, 36]

ফিল্টার এবং মানচিত্র একত্রিত করা

আপনি পূর্ববর্তী ফিল্টার এবং মানচিত্রের ক্রিয়াকলাপগুলিকে নিম্নরূপ একত্রিত করতে পারেন:

কোড ইমেজ দেখতে এখানে লিংক করুন

```
[9] তে: list(map(lambda x: x
                  ** 2,
                  ...
                  ফিল্টার (ল্যাষ্টড একটি মানের বর্গক্ষেত্র উপস্থাপন করে একটি
                  ...
                  আউট[9]: [9, 49, 1, 81, 25]
```

স্লিপেট [9] তে অনেক কিছু ঘটছে, তাই আসুন এটি আরও ঘনিষ্ঠভাবে দেখে নেওয়া যাক। প্রথমে, ফিল্টার সংখ্যার বিজোড় মান উপস্থাপন করে একটি পুনরাবৃত্তযোগ্য ফেরত দেয়। তারপর map ফিল্টার করা মানের বর্গক্ষেত্র উপস্থাপন করে একটি পুনরাবৃত্তযোগ্য ফেরত দেয়। অবশেষে, তালিকা তৈরি করতে তালিকাটি মানচিত্রের পুনরাবৃত্তযোগ্য ব্যবহার করে। আপনি পূর্ববর্তী স্লিপেটের চেয়ে নিম্নলিখিত তালিকার বোধগম্যতা পছন্দ করতে পারেন:

কোড ইমেজ দেখতে এখানে লিংক করুন

```
[10] তে: [x ** 2 সংখ্যায় x এর জন্য যদি x % 2 != 0]
আউট[10]: [9, 49, 1, 81, 25]
```

সংখ্যায় x এর প্রতিটি মানের জন্য, $x^{**} 2$ রাশিটি কেবল তখনই সম্পাদিত হয় যদি
শর্ত $x \% 2 != 0$ সত্য।

হ্রাস: যোগফল সহ একটি ক্রমের উপাদানগুলির যোগফল

আপনি জানেন যে রিডাকশন একটি সিকোয়েন্সের উপাদানগুলিকে একটি একক মানে রূপান্তর করে। আপনি বিল্টইন ফাংশন len, sum, min এবং max দিয়ে রিডাকশন সম্পাদন করেছেন। আপনি functools মডিউলের রিডাকশন ফাংশন ব্যবহার করে কাস্টম রিডাকশনও তৈরি করতে পারেন। দেখুন

কোড উদাহরণের জন্য <https://docs.python.org/3/library/functools.html> দেখুন।

যখন আমরা chapter 16- এ বিগ ডেটা এবং Hadoop অনুসন্ধান করব , তখন আমরা MapReduce প্রোগ্রামিং প্রদর্শন করব, যা ফাংশনাল-স্টাইল প্রোগ্রামিং-এ ফিল্টার, ম্যাপ এবং রিডুস অপারেশনের উপর ভিত্তি করে তৈরি।

৫.১৫ অন্যান্য সিকোয়েল প্রক্রিয়াকরণ ফাংশন

পাইথন সিকোয়েল ম্যানিপুলেট করার জন্য অন্যান্য বিল্টইন ফাংশন প্রদান করে।

একটি কী ফাংশন ব্যবহার করে সর্বনিম্ন এবং সর্বোচ্চ মান খুঁজে বের করা

আমরা পূর্বে ints অথবা ints এর তালিকার মতো আর্গুমেন্ট ব্যবহার করে min এবং max বিল্টইন রিডাকশন ফাংশনগুলি দেখিয়েছি। কখনও কখনও আপনাকে স্ট্রিং এর মতো জটিল বস্তুর সর্বনিম্ন এবং সর্বোচ্চ খুঁজে বের করতে হবে। নিম্নলিখিত তুলনাটি বিবেচনা করুন:

[1] তে: 'লাল' < 'কমলা'

আউট[1]: সত

'R' অক্ষরটি বর্ণমালায় 'o' এর পরে আসে, তাই আপনি 'Red' 'কমলা' এর চেয়ে কম এবং উপরের শর্তটি 'False' বলে আশা করতে পারেন। তবে, স্ট্রিংগুলিকে তাদের অক্ষরের অন্তর্নিহিত সংখ্যাসূচক মান দিয়ে তুলনা করা হয় এবং ছোট হাতের অক্ষরের সংখ্যাসূচক মান বড় হাতের অক্ষরের চেয়ে বেশি। আপনি এটি বিল্টইন ফাংশন ord দিয়ে নিশ্চিত করতে পারেন, যা একটি অক্ষরের সংখ্যাসূচক মান প্রদান করে:

[2] তে: ord('R')

আউট[2]: 82

[3] তে: ord('o')

আউট[3]: 111

তালিকার রঙগুলি বিবেচনা করুন, যেখানে বড় হাতের এবং ছোট হাতের অক্ষর সহ স্ট্রিং রয়েছে:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

[4] তে: রঙ = ['লাল', 'কমলা', 'হলুদ', 'সবুজ', 'নীল']

ধরা যাক আমরা সংখ্যাসূচক (আভিধানিক) ক্রম ব্যবহার করে নয়, বরং বর্ণানুক্রমিক ক্রম ব্যবহার করে সর্বনিম্ন এবং সর্বাধিক স্ট্রিং নির্ধারণ করতে চাই। যদি আমরা বর্ণানুক্রমিকভাবে রঙগুলি সাজাই

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

'নীল', 'সবুজ', 'কমলা', 'লাল', 'হলুদ'

বর্ণমালা), এবং 'হলুদ' হল সর্বোচ্চ (অর্থাৎ, বর্ণমালার শেষের সবচেয়ে কাছে)।

যেহেতু পাইথন সংখ্যাসূচক মান ব্যবহার করে স্ট্রিংগুলির তুলনা করে, তাই আপনাকে প্রথমে প্রতিটি স্ট্রিংকে সমস্ত ছোট হাতের অক্ষরে বা সমস্ত বড় হাতের অক্ষরে রূপান্তর করতে হবে। তারপর তাদের সংখ্যাসূচক মানগুলি বর্ণানুক্রমিক ক্রমও উপস্থাপন করবে। নিম্নলিখিত স্লিপেটগুলি সর্বনিম্ন এবং সর্বোচ্চ স্ট্রিংগুলিকে বর্ণানুক্রমিকভাবে নির্ধারণ করতে min এবং max সক্ষম করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[5] তে: min(colors, key=lambda s: s.lower())
```

আউট[5]: 'নীল'

```
[6] তে: max(colors, key=lambda s: s.lower())
```

আউট[6]: 'হলুদ'

কীওয়ার্ড আর্গুমেন্টটি অবশ্যই একটি oneparameter ফাংশন হতে হবে যা একটি মান প্রদান করে। এই ক্ষেত্রে, এটি একটি ল্যাষড়া যা স্ট্রিং মেথড lower কে কল করে একটি স্ট্রিংয়ের ছোট হাতের সংস্করণ পেতে পারে। min এবং max ফাংশন প্রতিটি উপাদানের জন্য কী আর্গুমেন্টের ফাংশন কল করে এবং ফলাফল ব্যবহার করে উপাদানগুলির তুলনা করে।

একটি ক্রমানুসারে পিছনের দিকে পুনরাবৃত্তি করা

বিল্টইন ফাংশন **রিভার্সড** একটি ইটারেটর প্রদান করে যা আপনাকে একটি সিকোয়েলের মানগুলিকে পিছনের দিকে পুনরাবৃত্তি করতে সক্ষম করে। নিম্নলিখিত তালিকার বোধগম্যতা বিপরীত ক্রমে সংখ্যার মানের বর্গ ধারণকারী একটি নতুন তালিকা তৈরি করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[7] তে: সংখ্যা = [10, 3, 7, 1, 9, 4, 2, 8, 5, 6]
```

```
[7] তে: reversed_numbers = [ reversed(numbers) তে আইটেমের জন্য আইটেম]
```

```
[8] তে: বিপরীত_সংখ্যা
```

আউট[8]: [36, 25, 64, 4, 16, 81, 1, 49, 9, 100]

সংশ্লিষ্ট উপাদানের টুপলে পুনরাবৃত্তিযোগ্যগুলিকে একত্রিত করা

বিল্টইন ফাংশন **zip** আপনাকে একই সময়ে একাধিক পুনরাবৃত্ত ডেটার উপর পুনরাবৃত্তি করতে সক্ষম করে। ফাংশনটি আর্গুমেন্ট হিসাবে যেকোনো সংখ্যক পুনরাবৃত্ত ডেটা গ্রহণ করে এবং একটি পুনরাবৃত্ত ডেটা প্রদান করে যা প্রতিটিতে একই সূচকে উপাদান ধারণকারী টিপল তৈরি করে। উদাহরণস্বরূপ, স্লিপেট [11] এর zip-এ কল করলে টিপল ('Bob', 3.5), ('Sue', 4.0) এবং ('Amanda', 3.75) তৈরি হয় যা প্রতিটির 0, 1 এবং 2 সূচকের উপাদানগুলি নিয়ে গঠিত।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[9] তে: নাম = ['বব', 'সু', 'আমান্ডা']

[10] তে: গ্রেড_পয়েন্ট_গড় = [3.5,

8.0, 0.75]

[11] তে: নামের জন্য , জিপ তে জিপিএ (নাম, গ্রেড_পয়েন্ট_গড়):

...: মুদ্রণ (f'Name={name}; GPA={gpa}')

...:

নাম= বব; জিপিএ= 3.5

নাম= মামলা; জিপিএ=8.0

নাম=আমান্ডা; জিপিএ=3.75

আমরা প্রতিটি টুপলকে name এবং gpa তে আনপ্যাক করি এবং প্রদর্শন করি। ফাংশন zip এর সংক্ষিপ্ততম আর্গুমেন্ট উৎপাদিত টুপলের সংখ্যা নির্ধারণ করে। এখানে উভয়ের দৈর্ঘ্য একই।

৫.১৬ দ্বি-মাত্রিক তালিকা

তালিকাগুলিতে উপাদান হিসেবে অন্যান্য তালিকাও থাকতে পারে। এই ধরনের নেস্টেড (বা বহুমাত্রিক)

তালিকার একটি সাধারণ ব্যবহার হল সারি এবং কলামে সাজানো তথ্য সমষ্টিত মানগুলির সারণি উপস্থাপন করা। একটি

নির্দিষ্ট সারণি উপাদান সনাক্ত করার জন্য, আমরা দুটি সূচক নির্দিষ্ট করি - প্রচলিত পদ্ধতি অনুসারে, প্রথমটি উপাদানটির সারি চিহ্নিত করে, দ্বিতীয়টি উপাদানটির

কলাম।

যে তালিকাগুলিতে একটি উপাদান সনাক্ত করার জন্য দুটি সূচকের প্রয়োজন হয় তাকে দ্বিমাত্রিক তালিকা (অথবা দ্বি-সূচক তালিকা বা দ্বি-সারক্ষিপ্ট তালিকা) বলা হয়। বহুমাত্রিক তালিকায় দুটিরও বেশি সূচক থাকতে পারে। এখানে, আমরা দ্বিমাত্রিক তালিকার পরিচয় করিয়ে দিচ্ছি।

দ্বিমাত্রিক তালিকা তৈরি করা

তিনটি সারি এবং চারটি কলাম (অর্থাৎ, একটি 3by4 তালিকা) সহ একটি দ্বিমাত্রিক তালিকা বিবেচনা করুন যা একটি কোর্সে চারটি পরীক্ষায় অংশগ্রহণকারী তিনজন শিক্ষার্থীর গ্রেড উপস্থাপন করতে পারে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: a = [[77, 68, 86, 73], [96, 87, 89, 81], [70, 90, 86, 81]]

তালিকাটি নিম্নরূপ লিখলে এর সারি এবং কলামের সারণী কাঠামো আরও স্পষ্ট হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
a = [[77, 68, 86, 73], # প্রথম শিক্ষার্থীর গ্রেড  
[96, 87, 89, 81], # দ্বিতীয় শিক্ষার্থীর গ্রেড [70, 90, 86, 81]] # তৃতীয় শিক্ষার্থীর গ্রেড
```

একটি দ্বি-মাত্রিক তালিকা চিত্রিত করা

নিচের চিত্রটিতে পরীক্ষার গ্রেডের মান সহ তালিকা a দেখানো হয়েছে:

	Column 0	Column 1	Column 2	Column 3
Row 0	77	68	86	73
Row 1	96	87	89	81
Row 2	70	90	86	81

দ্঵িমাত্রিক তালিকার উপাদানগুলি সনাক্তকরণ

নিম্নলিখিত চিত্রটি তালিকা a এর উপাদানগুলির নাম দেখায়:

	Column 0	Column 1	Column 2	Column 3
Row 0	a[0][0]	a[0][1]	a[0][2]	a[0][3]
Row 1	a[1][0]	a[1][1]	a[1][2]	a[1][3]
Row 2	a[2][0]	a[2][1]	a[2][2]	a[2][3]

Column index
 Row index
 List name

প্রতিটি উপাদানকে $a[i][j]$ রূপের একটি নাম দ্বারা চিহ্নিত করা হয়—a হল তালিকার নাম, এবং i এবং j হল সূচক যা যথাক্রমে প্রতিটি উপাদানের সারি এবং কলামকে স্বতন্ত্রভাবে সনাক্ত করে। সারি 0-এর সকল উপাদানের নাম প্রথম সূচক হিসেবে 0 থাকে। কলাম 3-এর সকল উপাদানের নাম দ্বিতীয় সূচক হিসেবে 3 থাকে।

দ্বিমাত্রিক তালিকায় a:

- ৭৭, ৬৮, ৮৬ এবং ৭৩ যথাক্রমে $a[0][0]$, $a[0][1]$, $a[0][2]$ এবং $a[0][3]$ আরম্ভ করে,
- ৯৬, ৮৭, ৮৯ এবং ৮১ $a[1][0]$, $a[1][1]$, $a[1][2]$ এবং $a[1][3]$ আরম্ভ করে,

যথাক্রমে, এবং

- ৭০, ৯০, ৮৬ এবং ৮১ $a[2][0]$, $a[2][1]$, $a[2][2]$ এবং $a[2][3]$ আরঙ্গ করে,
যথাক্রমে।

m সারি এবং n কলাম বিশিষ্ট একটি তালিকাকে **mbyn** তালিকা বলা হয় এবং এতে $m \times n$ উপাদান থাকে।

নিম্নলিখিত নেস্টেড ফর স্টেটমেন্টটি পূর্ববর্তী দ্বিমাত্রিকের সারিগুলি আউটপুট করে

একবারে একটি সারি তালিকাভুক্ত করুন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[2] তে: **a** তে সারির জন্য :

```
...:           সারিতে থাকা আইটেমের জন্য :
...:           ...
...:           মুদ্রণ (আইটেম, শেষ = ' ')
...:           মুদ্রণ()
...:
৭৭ ৬৮ ৮৬ ৭৩
৯৬ ৮৭ ৮৯ ৮১
৯০ ৯০ ৮৬ ৮১
```

নেস্টেড লুপগুলি কীভাবে কার্যকর হয়

তালিকার নাম, সারি ও কলামের সূচক এবং প্রতিটি উপাদানের মান প্রদর্শনের জন্য নেস্টেড লুপটি পরিবর্তন করা যাক:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3] তে: **i** এর জন্য , **enumerate(a)** তে সারি:

```
...:           j এর জন্য , enumerate(row) এ থাকা আইটেম :
...:           ...
...:           মুদ্রণ(f'a[{i}][{j}]={আইটেম}', শেষ=' ')
...:           মুদ্রণ()
...:
a[0][0]=৭৭ a[0][1]=৬৮ a[0][2]=৮৬ a[0][3]=৭৩
a[1][0]=৯৬ a[1][1]=৮৭ a[1][2]=৮৯ a[1][3]=৮১
a[2][0]=৭০ a[2][1]=৯০ a[2][2]=৮৬ a[2][3]=৮১
```

বাইরের for স্টেটমেন্টটি দ্বিমাত্রিক তালিকার সারির উপর এক সারিতে পুনরাবৃত্তি করে

সময়। বাইরের for স্টেটমেন্টের প্রতিটি পুনরাবৃত্তির সময়, ভিতরের for স্টেটমেন্টটি বর্তমান সারির প্রতিটি কলামের উপর পুনরাবৃত্তি করে। তাই বাইরের লুপের প্রথম পুনরাবৃত্তিতে,
সারি ০ হল

এবং নেস্টেড লুপটি এই তালিকার চারটি উপাদান $a[0][0]=77$, $a[0][1]=68$, $a[0][2]=86$ এবং $a[0][3]=73$

এর মধ্য দিয়ে পুনরাবৃত্তি করে।

বাইরের লুপের দ্বিতীয় পুনরাবৃত্তিতে, সারি 1 হল

[৯৩, ৮৭, ৮৯, ৮১]

এবং নেস্টেড লুপটি এই তালিকার চারটি উপাদান $a[1][0]=96$, $a[1][1]=87$, $a[1][2]=89$ এবং $a[1][3]=81$

এর মধ্য দিয়ে পুনরাবৃত্তি করে।

বাইরের লুপের তৃতীয় পুনরাবৃত্তিতে, সারি 2 হল

[৭০, ৯০, ৮৬, ৮১]

এবং নেস্টেড লুপটি এই তালিকার চারটি উপাদানের মধ্য দিয়ে পুনরাবৃত্তি করে $a[2][0]=70$, $a[2]$

$[1]=90$, $a[2][2]=86$ এবং $a[2][3]=81$ ।

"ArrayOriented Programming with NumPy" অধ্যায়ে, আমরা NumPy লাইব্রেরির ndarray সংগ্রহ এবং Pandas লাইব্রেরির DataFrame সংগ্রহ সম্পর্কে আলোচনা করব। এইগুলি আপনাকে এই বিভাগে দেখা দ্বিমাত্রিক তালিকার ম্যানিপুলেশনের তুলনায় বহুমাত্রিক সংগ্রহগুলিকে আরও সংক্ষিপ্ত এবং সুবিধাজনকভাবে পরিচালনা করতে সক্ষম করে।

৫.১৭ তথ্য বিজ্ঞানের ভূমিকা: সিমুলেশন এবং স্ট্যাটিক ডিজুয়ালাইজেশন

গত কয়েকটি অধ্যায়ের 'ডেটা সায়েন্সের ভূমিকা' বিভাগে মৌলিক বর্ণনামূলক পরিসংখ্যান নিয়ে আলোচনা করা হয়েছে।

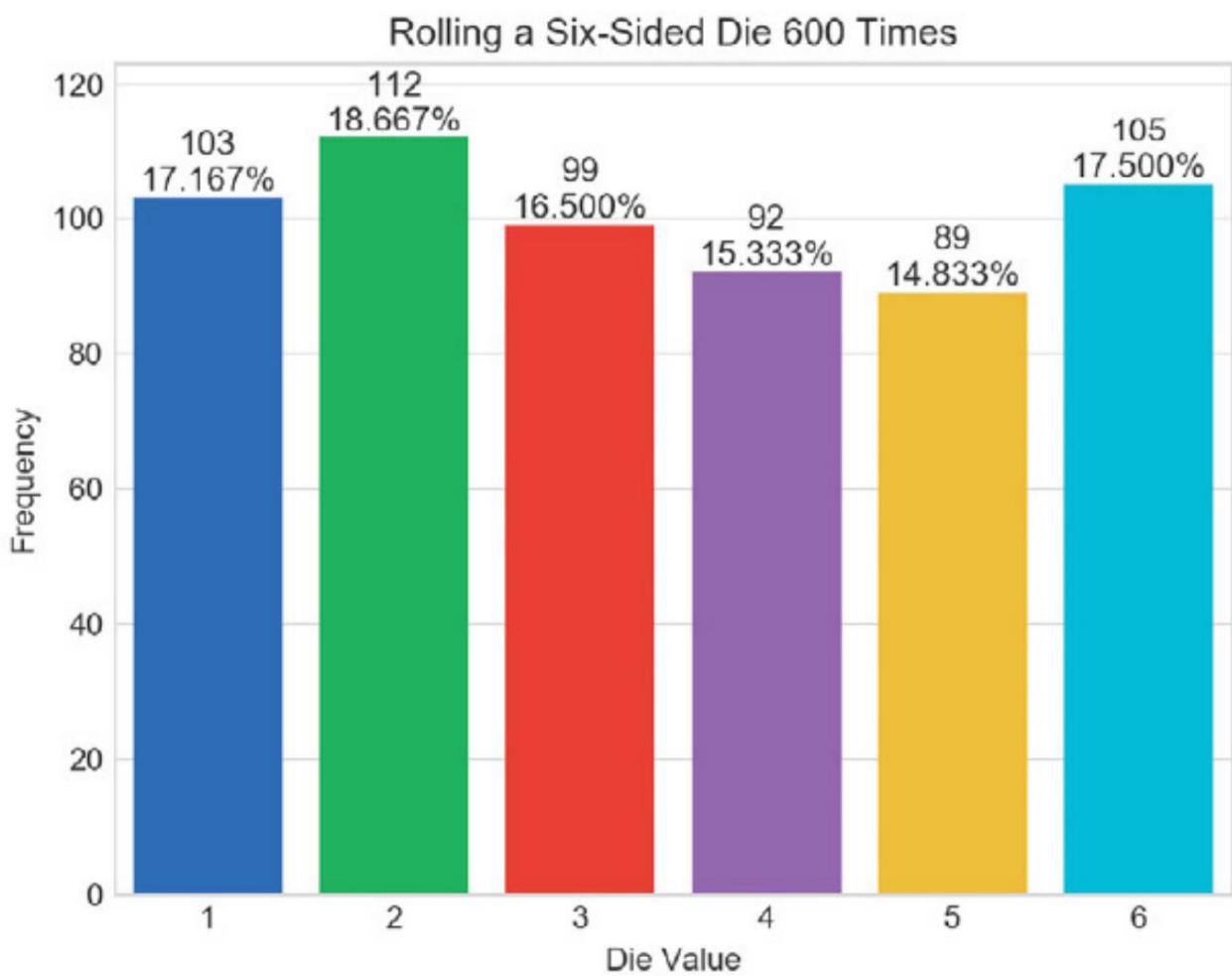
এখানে, আমরা ডিজুয়ালাইজেশনের উপর আলোকপাত করি, যা আপনাকে আপনার ডেটা "জানতে" সাহায্য করে।

ডিজুয়ালাইজেশন আপনাকে ডেটা বোঝার একটি শক্তিশালী উপায় দেয় যা কেবল কাঁচা ডেটা দেখার বাইরেও যায়।

আমরা দুটি ওপেনসোর্স ডিজুয়ালাইজেশন লাইব্রেরি ব্যবহার করি - সির্বন এবং ম্যাটপ্লটলিব - ছয় পার্শ্বযুক্ত ডাইরোলিং সিমুলেশনের চূড়ান্ত ফলাফল দেখানোর জন্য স্ট্যাটিক বার চার্ট প্রদর্শন করতে। **সির্বন ডিজুয়ালাইজেশন লাইব্রেরিটি ম্যাটপ্লটলিব ডিজুয়ালাইজেশন লাইব্রেরির** উপর ভিত্তি করে তৈরি এবং অনেক ম্যাটপ্লটলিব অপারেশনকে সহজ করে তোলে। আমরা উভয় লাইব্রেরির দিকগুলি ব্যবহার করব, কারণ সির্বন অপারেশনের কিছু অংশ ম্যাটপ্লটলিব লাইব্রেরি থেকে বস্তু ফেরত পাঠায়। পরবর্তী অধ্যায়ের ডেটা সায়েন্সের ভূমিকা বিভাগে, আমরা গতিশীল ডিজুয়ালাইজেশনের মাধ্যমে জিনিসগুলিকে "জীবন্ত" করে তুলব।

৫.১৭.১ ৬০০, ৬০,০০০ এবং ৬,০০০,০০০ ডাই রোলের নমুনা গ্রাফ

নীচের স্ক্রিন ক্যাপচারে একটি উল্লম্ব বার চার্ট দেখানো হয়েছে যা 600টি ডাই রোলের জন্য সারসংক্ষেপ করে ছয়টি মুখের প্রতিটি যে ফ্রিকোয়েল্সি সহ প্রদর্শিত হয় এবং মোটের তাদের শতাংশ। সিবর্ন এই ধরণের গ্রাফকে একটি বার প্লট হিসাবে উল্লেখ করে:



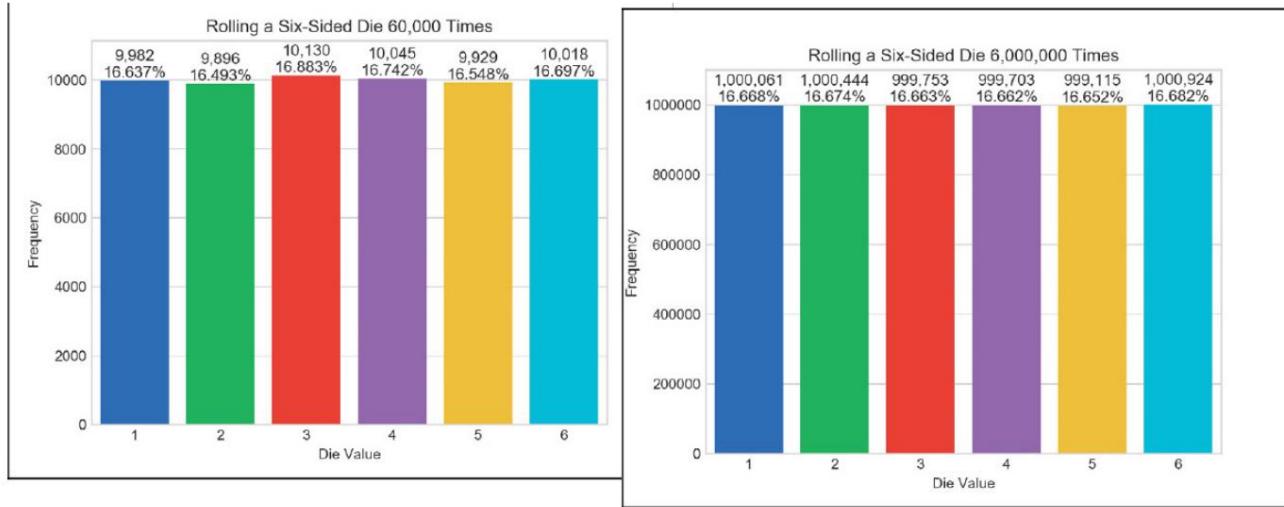
এখানে আমরা প্রতিটি ডাই ফেসের প্রায় ১০০টি ঘটনা আশা করছি। তবে, এত কম সংখ্যক রোলের ক্ষেত্রে, কোনও ফ্রিকোয়েল্সি ঠিক ১০০ নয় (যদিও বেশ কয়েকটি কাছাকাছি) এবং বেশিরভাগ শতাংশই ১৬.৬৬৭% (প্রায় ১/৬ ভাগ) এর কাছাকাছি নয়। আমরা যখন ৬০,০০০ ডাই রোলের জন্য সিমুলেশনটি চালাবো, তখন বারগুলি আকারে অনেক কাছাকাছি হয়ে যাবে। ৬,০০০,০০০ ডাই রোল, তারা ঠিক একই আকারের বলে মনে হবে। এটি "বড় সংখ্যার aw" কাজ করছে। পরবর্তী অধ্যায়ে বারগুলির দৈর্ঘ্য গতিশীলভাবে পরিবর্তিত হতে দেখা যাবে।

আমরা আলোচনা করব কিভাবে প্লটের চেহারা এবং বিষয়বস্তু নিয়ন্ত্রণ করা যায়, যার মধ্যে রয়েছে:

- জানালার ভেতরে গ্রাফের শিরোনাম (একটি ছয় পার্শ্বযুক্ত ডাই ৬০০ বার ঘূর্ণায়মান),
- বর্ণনামূলক লেবেলগুলি x অক্ষের জন্য ডাই মান এবং yaxis এর জন্য ফ্রিকোয়েল্সি,
- প্রতিটি বারের উপরে প্রদর্শিত টেক্সট, মোট রোলের ফ্রিকোয়েল্সি এবং শতাংশ প্রতিনিধিত্ব করে, এবং
- বারের রঙ।

আমরা বিভিন্ন Seaborn ডিফল্ট বিকল্প ব্যবহার করব। উদাহরণস্বরূপ, Seaborn ডাই ফেস মান 1–6 থেকে x অক্ষ বরাবর টেক্সট লেবেল এবং প্রকৃত ডাই ফ্রিকোয়েলি থেকে yaxis বরাবর টেক্সট লেবেল নির্ধারণ করে। পর্দার আড়ালে, Matplotlib বারগুলির অবস্থান এবং আকার নির্ধারণ করে, উইন্ডোর আকার এবং বারগুলি যে মানগুলি উপস্থাপন করে তার মাত্রার উপর ভিত্তি করে। এটি বারগুলি যে প্রকৃত ডাই ফ্রিকোয়েলিগুলি উপস্থাপন করে তার উপর ভিত্তি করে ফ্রিকোয়েলি অক্ষের সংখ্যাসূচক লেবেলগুলিকেও অবস্থান দেয়। আপনি আরও অনেক বৈশিষ্ট্য কাস্টমাইজ করতে পারেন। আপনার ব্যক্তিগত পছন্দ অনুসারে এই বৈশিষ্ট্যগুলি পরিবর্তন করা উচিত।

নীচের প্রথম স্ক্রিন ক্যাপচারে ৬০,০০০ ডাই রোলের ফলাফল দেখানো হয়েছে—কল্পনা করুন হাতে এটি করার চেষ্টা করা হচ্ছে। এই ক্ষেত্রে, আমরা প্রতিটি ফেসের প্রায় ১০,০০০টি ফলাফল আশা করি। নীচের দ্বিতীয় স্ক্রিন ক্যাপচারে ৬০,০০,০০০টি রোলের ফলাফল দেখানো হয়েছে—নিশ্চয়ই এমন কিছু যা আপনি কখনও হাতে করবেন না! এই ক্ষেত্রে, আমরা প্রতিটি ফেসের প্রায় ১০,০০,০০০টি ফলাফল আশা করি, এবং ফ্রিকোয়েলি বারগুলি দৈর্ঘ্য একই রকম বলে মনে হয় (এগুলি কাছাকাছি কিন্তু ঠিক একই দৈর্ঘ্যের নয়)। মনে রাখবেন যে আরও ডাই রোলের সাথে, ফ্রিকোয়েলি শতাংশ প্রত্যাশিত ১৬.৬৬৭% এর অনেক কাছাকাছি।



৫.১৭.২ ডাই-রোল ফ্রিকোয়েলি এবং শতাংশের কল্পনা করা

এই বিভাগে, আপনি পূর্ববর্তী বিভাগে দেখানো বার প্লটগুলি ইন্টারেক্শনভাবে বিকাশ করবেন।

ইন্টারেক্শন ম্যাটপ্লটলিব ডেভেলপমেন্টের জন্য আইপিথন চালু করা হচ্ছে

IPython-এ Matplotlib গ্রাফ তৈরির জন্য বিল্ট-ইন সাপোর্ট রয়েছে, যা আপনাকে Seaborn গ্রাফ তৈরি করতেও প্রয়োজন। কেবল এই কমান্ডটি দিয়ে IPython চালু করুন:

```
আইপাইথন ম্যাটপ্লটলিব
```

লাইব্রেরিগুলি আমদানি করা

প্রথমে, আমরা যে লাইব্রেরিগুলি ব্যবহার করব তা আমদানি করা যাক:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: matplotlib.pyplot কে plt হিসেবে আমদানি করুন

[2] তে: np হিসেবে numpy আমদানি করুন

[3] তে: র্যান্ডম আমদানি করুন

[4] তে: sns হিসেবে আমদানি seaborn

১. matplotlib.pyplot মডিউলটিতে আমরা যে Matplotlib লাইব্রেরির গ্রাফিং ক্ষমতা ব্যবহার করি তা রয়েছে। এই মডিউলটি সাধারণত plt নামে আমদানি করা হয়।

২. NumPy (Numerical Python) লাইব্রেরিতে unique ফাংশনটি রয়েছে যা আমরা ডাই রোলগুলি সারসংক্ষেপ করার জন্য ব্যবহার করব। numpy মডিউলটি সাধারণত np হিসাবে আমদানি করা হয়।

৩. র্যান্ডম মডিউলটিতে পাইথনের র্যান্ডম নম্বর জেনারেশন ফাংশন রয়েছে।

৪. seaborn মডিউলটিতে আমরা যে Seaborn লাইব্রেরি ব্যবহার করি তার গ্রাফিং ক্ষমতা রয়েছে।

এই মডিউলটি সাধারণত sns নাম দিয়ে আমদানি করা হয়। কেন এই কোতৃহল তা অনুসন্ধান করুন সংক্ষিপ্ত রূপটি বেছে নেওয়া হয়েছিল।

ডাই রোলিং এবং ডাই ফ্রিকোয়েন্সি গণনা করা

এরপর, আসুন একটি তালিকা বোঝার পদ্ধতি ব্যবহার করে 600টি র্যান্ডম ডাই মানের একটি তালিকা তৈরি করি, তারপর ব্যবহার করি অনন্য রোল মান (সম্ভবত ছয়টি সম্ভাব্য মুখ মান) এবং তাদের ফ্রিকোয়েন্সি নির্ধারণের জন্য NumPy-এর অনন্য ফাংশন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[5] তে: rolls = [random.randrange(1, 7) for i in range(600)]

[6] তে: মান, ফ্রিকোয়েন্সি = np.unique(rolls, return_counts=True)

NumPy লাইব্রেরি উচ্চ-পারফরম্যান্স ndarray সংগ্রহ প্রদান করে, যা সাধারণত তালিকার তুলনায় অনেক দ্রুত। যদিও আমরা এখানে সরাসরি ndarray ব্যবহার করি না^১, NumPy অনন্য ফাংশনটি একটি ndarray আর্গুমেন্ট আশা করে এবং একটি ndarray প্রদান করে। যদি আপনি একটি তালিকা পাস করেন (যেমন রোলস), NumPy আরও ভাল কর্মক্ষমতার জন্য এটিকে ndarray তে রূপান্তর করে।

আমরা যে ndarray দিয়ে অনন্য রিটার্ন করি তা কেবল একটি ভেরিয়েবলকে বরাদ্দ করব যা Seaborn প্লটিং ফাংশন দ্বারা ব্যবহারের জন্য।

^১ হ্যাপ্টার ৭- এ একটি পারফরম্যান্স তুলনা চালান যেখানে আমরা ndarray-তে আলোচনা করব www.EBooksWorld.ir

গতীরতা।

`return_counts=True` কীওয়ার্ড আর্গুমেন্টটি নির্দিষ্ট করলে `unique` কে প্রতিটি অনন্য মানের ঘটনার সংখ্যা গণনা করতে বলা হয়। এই ক্ষেত্রে, `unique` যথাক্রমে সাজানো অনন্য মান এবং সংশ্লিষ্ট ফ্রিকোয়েলি ধারণকারী দুটি এক-মাত্রিক `ndarrays` এর একটি টুপল প্রদান করে। আমরা টুপলের `ndarrays` কে ভেরিয়েবলের মান এবং ফ্রিকোয়েলিতে আনপ্যাক করি। যদি `return_counts False` হয়, তাহলে শুধুমাত্র অনন্য মানের তালিকা হবে

ফিরে এসেছে।

প্রাথমিক বার প্লট তৈরি করা

চলুন বার প্লটের শিরোনাম তৈরি করি, এর স্টাইল সেট করি, তারপর ডাই ফেস এবং ফ্রিকোয়েলি গ্রাফ করি:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[7] তে: title = f'Rolling a SixSided Die {len(rolls):,} Times'

[8] তে: sns.set_style('whitegrid')

[9] তে: axes = sns.barplot(x=মান, y=ফ্রিকোয়েলি, প্যালেট='উজ্জ্বল')
```

স্লিপেট [7]'sfstring-এ বার প্লটের শিরোনামে ডাই রোলের সংখ্যা অন্তর্ভুক্ত থাকে। কমা (,) ফর্ম্যাট স্পেসিফায়ার

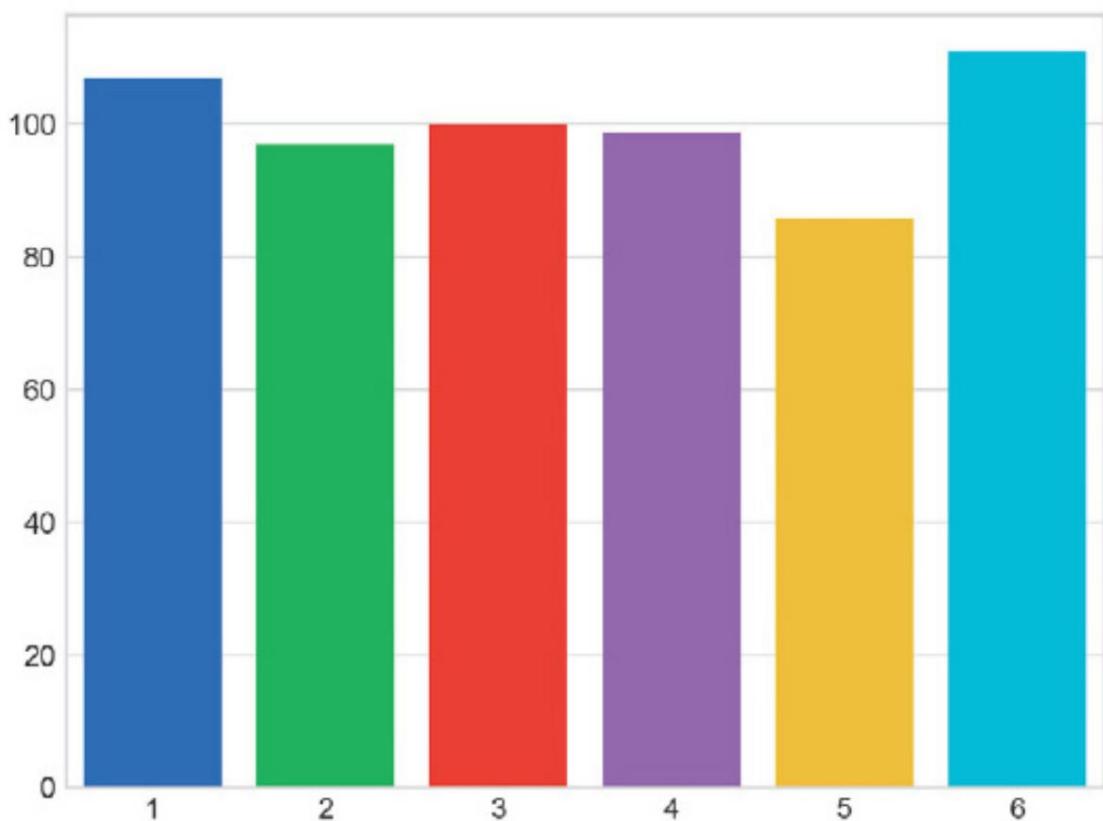
{লেন(রোলস):,}

হাজার বিভাজক সহ সংখ্যাটি প্রদর্শন করে—তাই, 60000 প্রদর্শিত হবে

৬০,০০০।

ডিফল্টরূপে, সির্বন একটি সাদা পটভূমিতে গ্রাফ প্লট করে, তবে এটি বেছে নেওয়ার জন্য বেশ কয়েকটি স্টাইল প্রদান করে ('ডার্কগ্রিড', 'হোয়াইটগ্রিড', 'ডার্ক', 'হোয়াইট' এবং 'টিকস')। স্লিপেট [8] 'হোয়াইটগ্রিড' স্টাইল নির্দিষ্ট করে, যা উল্লম্ব বার প্লটে হালকা ধূসুর অনুভূমিক রেখা প্রদর্শন করে। এগুলি আপনাকে আরও সহজে দেখতে সাহায্য করে যে প্রতিটি বারের উচ্চতা বার প্লটের বাম দিকের সংখ্যাসূচক ফ্রিকোয়েলি লেবেলের সাথে কীভাবে মিলে যায়।

স্লিপেট [9] সির্বনের বারপ্লট ফাংশন ব্যবহার করে ডাই ফ্রিকোয়েলি গ্রাফ করে। যখন আপনি এই স্লিপেটটি কার্যকর করেন, তখন নিম্নলিখিত উইন্ডোটি প্রদর্শিত হয় (কারণ আপনি matplotlib বিকল্পটি দিয়ে IPython চালু করেছেন):



Seaborn Matplotlib এর সাথে ইন্টারঅ্যাক্ট করে বারগুলি প্রদর্শন করে একটি Matplotlib **Axes** অবজেক্ট তৈরি করে, যা উইন্ডোতে প্রদর্শিত বিষয়বস্তু পরিচালনা করে। পর্দার আড়ালে, Seaborn একটি Matplotlib **Figure** অবজেক্ট ব্যবহার করে যে উইন্ডোতে Axes প্রদর্শিত হবে তা পরিচালনা করে। Function barplot এর প্রথম দুটি আর্গুমেন্ট হল ndarrays যার মধ্যে যথাক্রমে xaxis এবং yaxis মান রয়েছে। আমরা Seaborn এর পূর্বনির্ধারিত রঙ প্যালেট 'bright' নির্বাচন করতে ঐচ্ছিক প্যালেট কীওয়ার্ড আর্গুমেন্ট ব্যবহার করেছি। আপনি প্যালেট বিকল্পগুলি দেখতে পারেন।

এ:

https://seaborn.pydata.org/tutorial/color_palettes.html

ফাংশন barplot এটি কনফিগার করা Axes অবজেক্টটি ফেরত দেয়। আমরা এটিকে চলক অক্ষগুলিতে বরাদ্দ করি যাতে আমরা এটি ব্যবহার করে আমাদের চূড়ান্ত প্লটের অন্যান্য দিকগুলি কনফিগার করতে পারি। এই পয়েন্টের পরে বার প্লটে আপনার করা যেকোনো পরিবর্তন আপনি সংশ্লিষ্ট স্নিপেটটি কার্যকর করার সাথে সাথেই প্রদর্শিত হবে।

উইন্ডোর শিরোনাম নির্ধারণ এবং x- এবং y-অক্ষ লেবেল করা

পরবর্তী দুটি স্নিপেট বার প্লটে কিছু বর্ণনামূলক টেক্সট যোগ করে:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

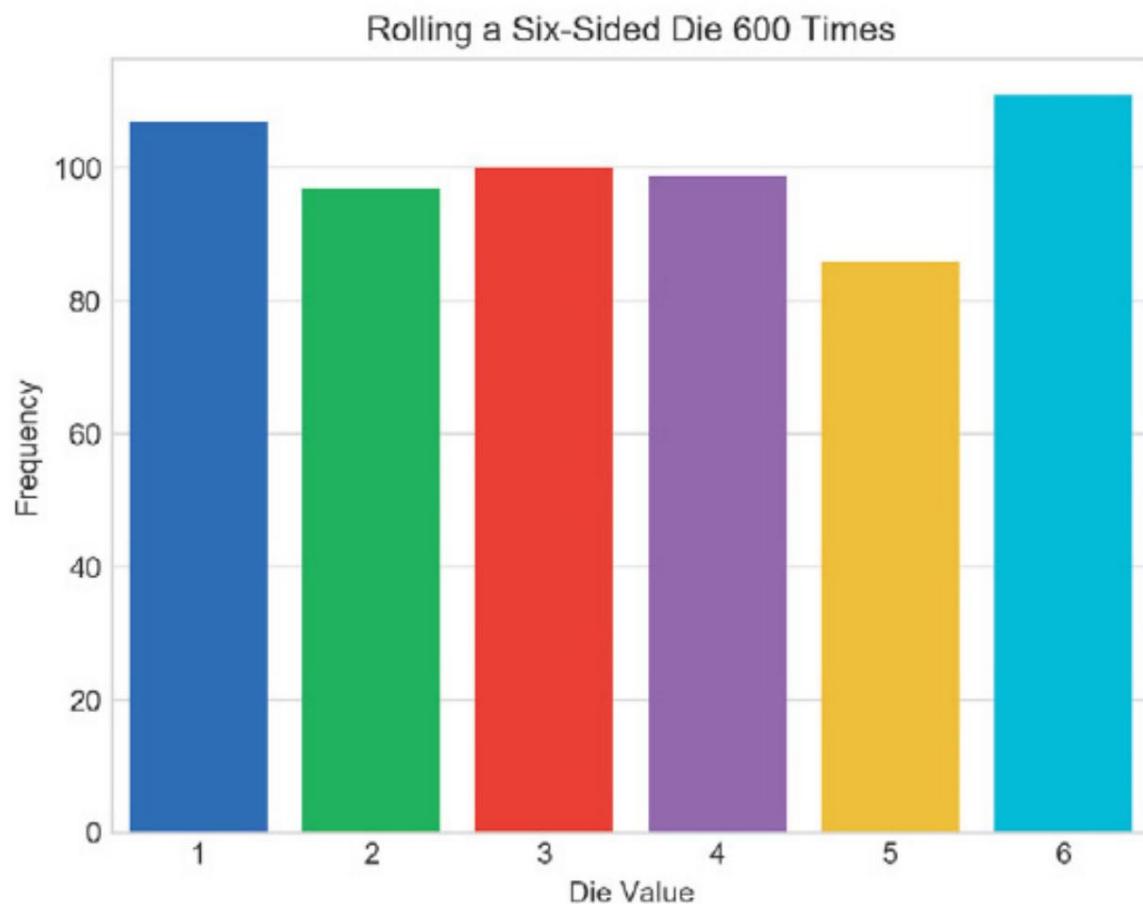
[10] তে: axes.set_title(title)

আউট[10]: টেক্সট(0.5, 1, 'একটি ছয় পার্শ্বযুক্ত ডাই ৬০০ বার ঘূর্ণায়মান')

```
[11]: তে: axes.set(xlabel='Die Value', ylabel='Frequency')
আউট[11]: [টেক্সট(92.6667,0.5,'ফ্রিকোয়েন্সি'), টেক্সট(0.5,58.7667,'ডাই ভ্যালু')]
```

স্নিপেট [10] প্লটের উপরে কেন্দ্রীভূত টাইটেল স্ট্রিং প্রদর্শনের জন্য axes অবজেক্টের `set_title` পদ্ধতি ব্যবহার করে। এই পদ্ধতিটি উইন্ডোতে শিরোনাম এবং এর অবস্থান সম্পর্কিত একটি টেক্সট অবজেক্ট প্রদান করে, যা IPython কেবল নিশ্চিতকরণের জন্য আউটপুট হিসাবে প্রদর্শন করে। আপনি উপরের স্নিপেটগুলিতে Out[]গুলি উপেক্ষা করতে পারেন।

স্নিপেট [11] প্রতিটি অক্ষে লেবেল যোগ করুন। `সেট` পদ্ধতিটি Axes অবজেক্টের বৈশিষ্ট্য সেট করার জন্য কীওয়ার্ড আর্গুমেন্ট গ্রহণ করে। পদ্ধতিটি x-অক্ষ বরাবর xlabel টেক্সট এবং yaxis বরাবর ylabel টেক্সট প্রদর্শন করে এবং লেবেল এবং তাদের অবস্থান ধারণকারী টেক্সট অবজেক্টের একটি তালিকা প্রদান করে। বার প্লটটি এখন নিষ্পত্তি প্রদর্শিত হবে:



[বার প্লট চূড়ান্ত করা হচ্ছে](#)

পরবর্তী দুটি স্নিপেট প্রতিটি বারের উপরে লেখার জন্য জায়গা করে গ্রাফটি সম্পূর্ণ করে, তারপর এটি প্রদর্শন করে:

[কোড ইমেজ দেখতে এখানে লিঙ্ক করুন](#)

```
[13] তে: বারের জন্য, জিপ (axes.patches, ফ্রিকোয়েলি) তে ফ্রিকোয়েলি: text_x = bar.get_x() + bar.get_width() / 2.0 text_y
...:           = bar.get_height() text = f'{frequency:.}\n{frequency / len(rolls):.3%}'
...:
...:
...:           axes.text(টেক্স্ট_এক্স, টেক্স্ট_ওয়াই, টেক্স্ট,
...:           ফন্টসাইজ=১১, হা='কেন্দ্র', ভা='নীচে')
...:
```

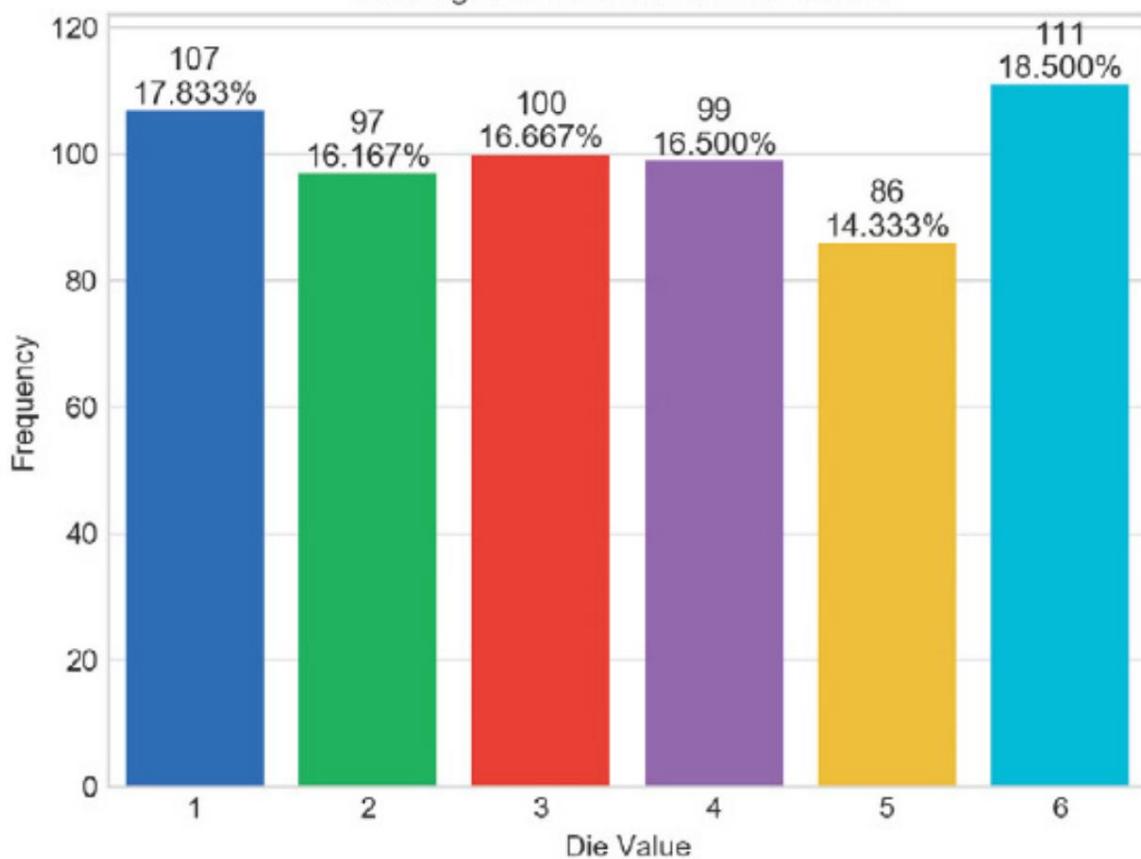
বারগুলির উপরে লেখার জন্য জায়গা তৈরি করতে, স্লিপেট [12] yaxis কে 10% স্কেল করে। আমরা পরীক্ষার মাধ্যমে এই মানটি বেছে নিয়েছি। Axes অবজেক্টের [set_ylim](#) পদ্ধতিতে অনেক গ্রিচিক কীওয়ার্ড আর্গুমেন্ট রয়েছে। এখানে, আমরা yaxis দ্বারা উপস্থাপিত সর্বোচ্চ মান পরিবর্তন করতে শুধুমাত্র top ব্যবহার করি। আমরা সর্বোচ্চ ফ্রিকোয়েলি 1.10 দ্বারা গুণ করেছি যাতে নিশ্চিত করা যায় যে yaxisটি সবচেয়ে লম্বা বারের চেয়ে 10% লম্বা।

অবশেষে, স্লিপেট [13] প্রতিটি বারের ফ্রিকোয়েলি মান এবং মোট রোলের শতাংশ প্রদর্শন করে। অক্ষ বস্তুর প্যাচ সংগ্রহে দ্বিমাত্রিক রঙিন আকার রয়েছে যা প্লটের বারগুলিকে প্রতিনিধিত্ব করে। for স্টেটমেন্টটি প্যাচ এবং তাদের সংশ্লিষ্ট ফ্রিকোয়েলি মানগুলির মাধ্যমে পুনরাবৃত্তি করার জন্য zip ব্যবহার করে। প্রতিটি পুনরাবৃত্তি বারে আনপ্যাক করে এবং ফ্রিকোয়েলিটি টুপলগুলির মধ্যে একটি জিপ ফেরত দেয়। for স্টেটমেন্টের সুট্টি এইভাবে কাজ করে

অনুসরণ করে:

- প্রথম স্টেটমেন্টটি কেন্দ্র xcoordinate গণনা করে যেখানে টেক্স্টটি প্রদর্শিত হবে। আমরা এটি বারের বাম প্রান্ত xcoordinate (bar.get_x()) এবং বারের প্রস্তুর অর্ধেক (bar.get_width() / 2.0) এর যোগফল হিসাবে গণনা করি।
- দ্বিতীয় স্টেটমেন্টটি ycoordinate পায় যেখানে টেক্স্টটি প্রদর্শিত হবে —bar.get_y() বারের শীর্ষকে প্রতিনিধিত্ব করে।
- তৃতীয় বিবৃতিটি একটি দুই-লাইনের স্ট্রিং তৈরি করে যাতে সেই বারের ফ্রিকোয়েলি এবং মোট ডাই রোলের সংশ্লিষ্ট শতাংশ থাকে।
- শেষ বিবৃতিটি Axes অবজেক্টের [টেক্স্ট](#) পদ্ধতিকে বারের উপরে টেক্স্ট প্রদর্শনের জন্য বলে। এই পদ্ধতির প্রথম দুটি আর্গুমেন্ট টেক্স্টের x-y অবস্থান নির্দিষ্ট করে এবং তৃতীয় আর্গুমেন্টটি হল টেক্স্টটি প্রদর্শনের জন্য। কীওয়ার্ড আর্গুমেন্ট ha অনুভূমিক সারিবদ্ধকরণ নির্দিষ্ট করে - আমরা xcoordinate এর চারপাশে অনুভূমিকভাবে টেক্স্ট কেন্দ্রীভূত করেছি। কীওয়ার্ড আর্গুমেন্ট va উল্লম্ব সারিবদ্ধকরণ নির্দিষ্ট করে - আমরা টেক্স্টের নীচে ycoordinate এ সারিবদ্ধ করেছি। চূড়ান্ত বার খাটটি নীচে দেখানো হয়েছে:

Rolling a Six-Sided Die 600 Times



আবার ঘুরছি এবং বার প্লট আপডেট করছি—আইপিথন ম্যাজিস্ট্রের সাথে পরিচয় করিয়ে দিচ্ছি এখন যেহেতু আপনি একটি সুন্দর বার প্লট তৈরি করেছেন, আপনি সম্ভবত ভিন্ন সংখ্যক ডাই রোল চেষ্টা করতে চাইবেন। প্রথমে, Matplotlib এর `cla` (স্পষ্ট অক্ষ) ফাংশনটি কল করে বিদ্যমান গ্রাফটি পরিষ্কার করুন:

```
[14] তে: plt.cla()
```

IPython বিভিন্ন কাজ সহজে করার জন্য `magics` নামক বিশেষ কমান্ড প্রদান করে। চলুন `%recall magic` ব্যবহার করে snippet [5] পাই, যা রোল তালিকা তৈরি করে, এবং পরবর্তী In [] প্রম্পটে কোডটি রাখি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[15] তে: % রিকল 5
```

```
[16] তে: rolls = [random.randrange(1, 7) for i in range(600)]
```

আপনি এখন স্লিপেটটি সম্পাদনা করে রোলের সংখ্যা 60000 এ পরিবর্তন করতে পারেন, তারপর এন্টার টিপুন। একটি নতুন তালিকা তৈরি করতে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[16] ☺: rolls = [random.randrange(1, 7) for i in range(60000)]
```

এরপর, [6] থেকে [13] স্লিপেটগুলি প্রত্যাহার করুন। এটি পরবর্তী In [] প্রম্পটে নির্দিষ্ট পরিসরের সমস্ত স্লিপেট প্রদর্শন করে। এই স্লিপেটগুলি পুনরায় কার্যকর করতে Enter টিপুন:

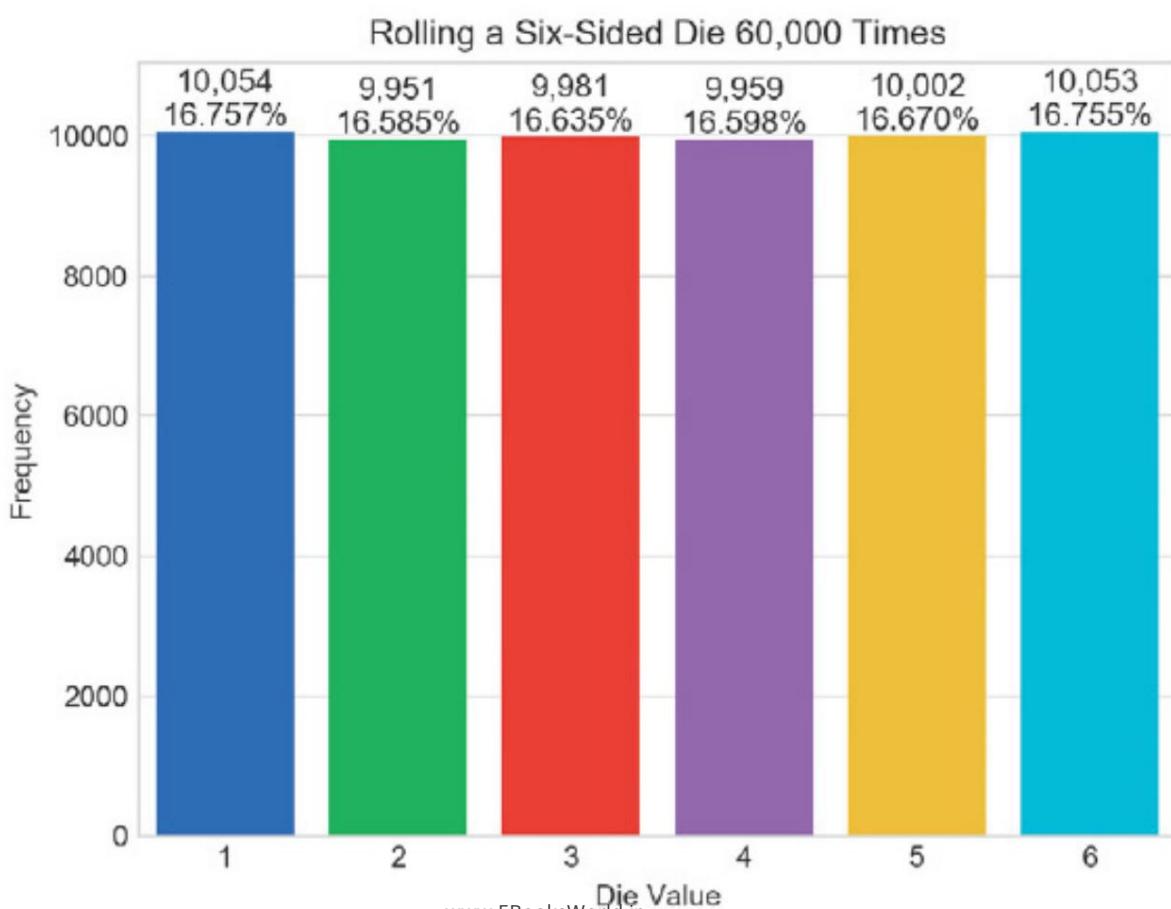
কোড ইমেজ দেখতে এখানে ক্লিক করুন

[17] তে: % রিকল 613

[18] তে: মান, ফ্রিকোয়েলি = np.unique(rolls, return_counts=True)

```
...: title = f'Rolling a SixSided Die {len(rolls)}: Times' ...: sns.set_style('whitegrid') ...: axes = sns.barplot(x=মান,  
y=ফ্রিকোয়েলি, palette='উজ্জ্বল') ...: axes.set_title(title) ...:  
axes.set(xlabel='ডাই ভ্যালু', ylabel='ফ্রিকোয়েলি')
```

আপডেট কৰা বাব প্লটটি বীচে দেখানো হয়েছে:



%save ম্যাজিক ব্যবহার করে একটি ফাইলে স্লিপেট সংরক্ষণ করা হচ্ছে

একবার আপনি ইন্টারেক্শনে একটি প্লট তৈরি করার পরে, আপনি কোডটি একটি ফাইলে সংরক্ষণ করতে চাইতে পারেন যাতে আপনি এটিকে স্লিপেট রূপান্তর করতে পারেন এবং ভবিষ্যতে এটি চালাতে পারেন। আসুন %save ম্যাজিক ব্যবহার করে RollDie.py নামক একটি ফাইলে স্লিপেট 1 থেকে 13 সংরক্ষণ করি। IPython সেই ফাইলটি নির্দেশ করে যেখানে লাইনগুলি লেখা হয়েছিল, তারপর এটি যে লাইনগুলি সংরক্ষণ করেছিল তা প্রদর্শন করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[19] তে: %save RollDie.py 113 `RollDie.py` ফাইল করার

জন্য নিয়ন্ত্রিত করার জন্য import matplotlib.pyplot as plt

np হিসেবে numpy আমদানি করুন

এলোমেলোভাবে আমদানি করুন

এসএনএস হিসেবে সিবর্ন আমদানি করুন

রোলস = [range(600) এর i এর জন্য random.randrange(1, 7)]

মান, ফ্রিকোয়েল্সি = np.unique(rolls, return_counts=True) শিরোনাম = একটি সিঙ্গাইডেড ডাই রোলিং করার সময়

{len(rolls):} সময়ের sns.set_style("whitegrid") অক্ষ = sns.barplot(মান, ফ্রিকোয়েল্সি, প্যালেট='উজ্জ্বল')

```
axes.set_title(title) axes.set(xlabel='Die
Value', ylabel='Frequency')
axes.set_xlim(top=max(frequencies) * 1.10) বারের জন্য, জিপে
ফ্রিকোয়েল্সি(axes.patches, ফ্রিকোয়েল্সি): text_x = bar.get_x() + bar.get_width() / 2.0 text_y =
    bar.get_height() text = f'{frequency:,}\n{frequency / len(rolls):.3%}'
    axes.text(text_x, text_y, text,
```

ফন্টসাইজ=১১, হা='কেন্দ্র', ভা='নীচে')

কমান্ড-লাইন আর্গুমেন্ট; একটি স্লিপেট থেকে একটি প্লট প্রদর্শন করা

এই অধ্যায়ের উদাহরণগুলির সাথে উপরে সংরক্ষিত RollDie.py ফাইলের একটি সম্পাদিত সংস্করণ দেওয়া হয়েছে। আমরা মন্তব্য এবং দুটি পরিবর্তন ঘোগ করেছি যাতে আপনি একটি আর্গুমেন্ট দিয়ে স্লিপেটটি চালাতে পারেন যা ডাই রোলের সংখ্যা নির্দিষ্ট করে, যেমন:

ipython RollDie.py 600 সম্পর্কে

পাইথন স্ট্যান্ডার্ড লাইব্রেরির sys মডিউল একটি স্লিপেটকে প্রোগ্রামে পাস করা কমান্ডলাইন আর্গুমেন্ট গ্রহণ করতে সক্ষম করে। এর মধ্যে স্লিপেটের নাম এবং স্লিপেটটি কার্যকর করার সময় এর ডানদিকে প্রদর্শিত যেকোনো মান অন্তর্ভুক্ত থাকে। sys মডিউলের argv তালিকায় আর্গুমেন্টগুলি রয়েছে। উপরের কমান্ডে, argv[0] হল 'RollDie.py' স্ট্রিং এবং argv[1] হল '600' স্ট্রিং। কমান্ডলাইন আর্গুমেন্টের মান দিয়ে ডাই রোলের সংখ্যা নিয়ন্ত্রণ করতে, আমরা স্টেটমেন্টটি পরিবর্তন করেছি যা তৈরি করে

রোল তালিকা নিম্নরূপ:

```
রোলস = [random.randrange(1, 7) for i in range(int(sys.argv[1]))]
```

মনে রাখবেন যে আমরা argv[1] স্ট্রিংটিকে একটি int-এ রূপান্তর করেছি।

যখন আপনি স্ক্রিপ্ট প্লট তৈরি করেন তখন Matplotlib এবং Seaborn স্বয়ংক্রিয়ভাবে আপনার জন্য প্লটটি প্রদর্শন করে না। তাই স্ক্রিপ্টের শেষে আমরা Matplotlib এর `show` ফাংশনে নিম্নলিখিত কলটি যুক্ত করেছি, যা গ্রাফ ধারণকারী উইডোটি প্রদর্শন করে:

```
plt.show()
```

৫.১৮ র্যাপ-আপ

এই অধ্যায়ে তালিকা এবং টুপল সিকোয়েল্স সম্পর্কে আরও বিস্তারিত আলোচনা করা হয়েছে। আপনি তালিকা তৈরি করেছেন, তাদের উপাদানগুলি অ্যাক্সেস করেছেন এবং তাদের দৈর্ঘ্য নির্ধারণ করেছেন। আপনি দেখেছেন যে তালিকাগুলি পরিবর্তনযোগ্য, তাই আপনি তাদের বিষয়বস্তু পরিবর্তন করতে পারেন, যার মধ্যে আপনার প্রোগ্রামগুলি কার্যকর করার সময় তালিকাগুলি বৃদ্ধি এবং সঙ্কুচিত করা অন্তর্ভুক্ত। আপনি দেখেছেন যে একটি অস্তিত্বহীন উপাদান অ্যাক্সেস করার ফলে একটি IndexError হয়। আপনি তালিকা উপাদানগুলির মাধ্যমে পুনরাবৃত্তি করার জন্য for বিবৃতি ব্যবহার করেছেন।

আমরা টিপল নিয়ে আলোচনা করেছি, যা তালিকার মতো ক্রম, কিন্তু অপরিবর্তনীয়। আপনি একটি টিপলের উপাদানগুলিকে পৃথক ভেরিয়েবলে আনপ্যাক করেছেন। আপনি enumerate ব্যবহার করে টিপলগুলির একটি পুনরাবৃত্তিযোগ্য তৈরি করেছেন, প্রতিটির একটি তালিকা সূচক এবং সংশ্লিষ্ট উপাদান মান রয়েছে।

তুমি জেনেছো যে সকল সিকোয়েল্স স্লাইসিং সমর্থন করে, যা মূল উপাদানের উপসেট দিয়ে নতুন সিকোয়েল্স তৈরি করে। তুমি তালিকা থেকে উপাদানগুলি সরাতে এবং ইন্টারেক্টিভ সেশন থেকে ভেরিয়েবলগুলি মুছে ফেলার জন্য del স্টেটমেন্ট ব্যবহার করেছ। আমরা তালিকা, তালিকা উপাদান এবং তালিকার স্লাইসগুলিকে ফাংশনগুলিতে পাস করেছি। তুমি তালিকাগুলি কীভাবে অনুসন্ধান এবং সার্জানো ঘায় এবং টিপলগুলি কীভাবে অনুসন্ধান করতে হয় তা দেখেছো। আমরা উপাদানগুলি সরিবেশ করতে, সংযোজন করতে এবং অপসারণ করতে এবং তালিকার উপাদানগুলি এবং অনুলিপি তালিকাগুলি বিপরীত করতে তালিকা পদ্ধতি ব্যবহার করেছি।

আমরা তালিকার সাহায্যে স্ট্যাক সিমুলেট করার পদ্ধতি দেখিয়েছি। নতুন তালিকা তৈরি করতে আমরা সংক্ষিপ্ত তালিকা বোঝার স্বরলিপি ব্যবহার করেছি। তালিকার উপাদানগুলিকে যোগ করার জন্য, তালিকার মধ্য দিয়ে পিছনের দিকে পুনরাবৃত্তি করার জন্য, সর্বনিম্ন এবং সর্বাধিক মান খুঁজে বের করার জন্য, মানগুলিকে ফিল্টার করার জন্য এবং মানগুলিকে নতুন মানগুলিতে ম্যাপ করার জন্য আমরা অতিরিক্ত অন্তর্নির্মিত পদ্ধতি ব্যবহার করেছি। আমরা দেখিয়েছি কিভাবে নেস্টেড তালিকা দ্বি-মাত্রিক টেবিল উপস্থাপন করতে পারে যেখানে ডেটা সারি এবং কলামে সাজানো থাকে। আপনি দেখেছেন কিভাবে নেস্টেড ফর লুপ দ্বি-মাত্রিক তালিকা প্রক্রিয়া করে।

অধ্যায়টি ডেটা সায়েন্সের ভূমিকা বিভাগের মাধ্যমে শেষ হয়েছে যেখানে একটি ডাই-

অলিং সিমুলেশন এবং স্ট্যাটিক ভিজুয়ালাইজেশন। সির্বন এবং ম্যাটপ্লটলির ভিজুয়ালাইজেশন লাইব্রেরি ব্যবহার করে সিমুলেশনের চূড়ান্ত ফলাফলের একটি স্ট্যাটিক বার প্লট ভিজুয়ালাইজেশন তৈরি করার জন্য একটি বিস্তারিত কোড উদাহরণ। পরবর্তী ডেটা সায়েন্সের ভূমিকা বিভাগে, আমরা প্লটটিকে "জীবন্ত" করার জন্য একটি ডায়নামিক বার প্লট ভিজুয়ালাইজেশন সহ একটি ডাইরোলিং সিমুলেশন ব্যবহার করব।

পরবর্তী অধ্যায়, "অভিধান এবং সেট"-এ আমরা পাইথনের অন্তর্নির্মিত সংগ্রহ সম্পর্কে আমাদের আলোচনা চালিয়ে যাব। আমরা অভিধান ব্যবহার করে কী-মান জোড়ার অ-ক্রমাগত সংগ্রহ সংরক্ষণ করব যা অপরিবর্তনীয় কীগুলিকে মানগুলিতে ম্যাপ করে, ঠিক যেমন একটি প্রচলিত অভিধান শব্দগুলিকে সংজ্ঞায় ম্যাপ করে। আমরা অনন্য উপাদানগুলির অ-ক্রমাগত সংগ্রহ সংরক্ষণ করতে সেট ব্যবহার করব।

"ArrayOriented Programming with NumPy" অধ্যায়ে, আমরা NumPy-এর ndarray সংগ্রহ সম্পর্কে আরও বিস্তারিত আলোচনা করব। আপনি দেখতে পাবেন যে তালিকাগুলি অল্ল পরিমাণে ডেটার জন্য ঠিক থাকলেও, বড় ডেটা অ্যানালিটিক্যাল অ্যাপ্লিকেশনগুলিতে আপনি যে বৃহৎ পরিমাণ ডেটার মুখোমুখি হবেন তার জন্য এগুলি কার্যকর নয়। এই ধরনের ক্ষেত্রে, NumPy লাইব্রেরির অত্যন্ত অপ্টিমাইজড ndarray সংগ্রহ ব্যবহার করা উচিত। ndarray (ndimensional array) তালিকার তুলনায় অনেক দ্রুত হতে পারে। আমরা পাইথন প্রোফাইলিং পরীক্ষা চালাব যাতে দেখা যায় যে কত দ্রুত। আপনি দেখতে পাবেন, NumPy-তে অনেক মাত্রার অ্যারে সুবিধাজনক এবং দক্ষতার সাথে পরিচালনা করার জন্য অনেক ক্ষমতাও রয়েছে। বড় ডেটা অ্যানালিটিক্যাল অ্যাপ্লিকেশনগুলিতে, প্রক্রিয়াকরণের চাহিদা বিশাল হতে পারে, তাই কর্মক্ষমতা উন্নত করার জন্য আমরা যা কিছু করতে পারি তা উল্লেখযোগ্যভাবে গুরুত্বপূর্ণ।

^১ আমাদের ^২ "Big Data: Hadoop, Spark, NoSQL-এবং IoT"-অধ্যায়ে, আপনি সবচেয়ে জনপ্রিয় উচ্চ-পারফরম্যান্স বিগডেটা ডাটাবেসগুলির মধ্যে একটি ব্যবহার করবেন - MongoDB।

^৩

ডাটাবেসের নামটি "humongous" শব্দটি থেকে উদ্ভূত।

লে-লিস্ট

গুরুত্বপূর্ণ অভিধান এবং সেট

উদ্দেশ্যমূলক মতামত

এই অধ্যায়ে, আপনি পাবেন: উপার্জনের
পথ

■ কী-মান জোড়ার অক্রমিক সংগ্রহ উপস্থাপন করতে অভিধান ব্যবহার করুন।
অফার্স এবং ডিল

■ অনন্য মানের অক্রমিক সংগ্রহ উপস্থাপন করতে সেট ব্যবহার করুন। highlights

■ অভিধান এবং সেটের উপাদান তৈরি করুন, আরও করুন এবং উল্লেখ করুন।

এটিং

■ একটি অভিধানের কী, মান এবং কী-মান জোড়ার মাধ্যমে পুনরাবৃত্তি করুন।

অভিধানের

■ কী-মান জোড়া যোগ, অপসারণ এবং আপডেট সমর্থন করুন।

সাইন

■ আউট করুন অভিধান ব্যবহার করুন এবং তুলনামূলক অপারেটর সেট করুন।

■ সেট অপারেটর এবং পদ্ধতির সাথে সেট একত্রিত করুন।

■ অভিধানে কী আছে নাকি সেট আছে তা নির্ধারণ করতে in অপারেটর ব্যবহার করুন এবং in ব্যবহার না করে একটি মান ধারণ করো।

■ একটি সেটের বিষয়বস্তু পরিবর্তন করতে পরিবর্তনযোগ্য সেট অপারেশন ব্যবহার করুন।

■ দ্রুত এবং সুবিধাজনকভাবে অভিধান এবং সেট তৈরি করতে বোধগম্যতা ব্যবহার করুন।

■ গতিশীল ডিজ্যুয়ালাইজেশন কীভাবে তৈরি করবেন তা শিখুন।

■ পরিবর্তনশীলতা এবং অপরিবর্তনীয়তা সম্পর্কে আপনার বোধগম্যতা বৃদ্ধি করুন।

রূপরেখা

.1 ভূমিকা

.২ অভিধান

.2.1 একটি অভিধান তৈরি করা

.2.2 একটি অভিধানের মাধ্যমে পুনরাবৃত্তি করা

.2.3 মৌলিক অভিধান পরিচালনা

.2.4 অভিধান পদ্ধতি কী এবং মান

.2.5 অভিধান তুলনা

.2.6 উদাহরণ: ছাত্র প্রেডের অভিধান

.2.7 উদাহরণ: শব্দ গণনা

.2.8 অভিধান পদ্ধতি আপডেট

.2.9 অভিধানের বোধগম্যতা

.3 সেট

.3.1 সেট তুলনা করা

.3.2 গাণিতিক সেট অপারেশন

.3.3 পরিবর্তনযোগ্য সেট অপারেটর এবং পদ্ধতি

.3.4 সেট কম্পিউটেশন

.4 ডেটা সায়েন্সের ভূমিকা: গতিশীল ডিজুয়ালাইজেশন

.4.1 ডায়নামিক ডিজুয়ালাইজেশন কীভাবে কাজ করে

.4.2 একটি গতিশীল ডিজুয়ালাইজেশন বাস্তবায়ন

.5 সারসংক্ষেপ

৬.১ ভূমিকা

আমরা তিনটি অন্তর্নির্মিত ক্রম সংগ্রহ নিয়ে আলোচনা করেছি—স্ট্রিং, তালিকা এবং টিপল। এখন, আমরা অন্তর্নির্মিত ননসিকোয়েল সংগ্রহ—অভিধান এবং সেট বিবেচনা করব। একটি **অভিধান** হল একটি অ-ক্রমযুক্ত সংগ্রহ যা **কী-মান জোড়া** সংরক্ষণ করে যা মানগুলির জন্য অপরিবর্তনীয় কীগুলিকে ম্যাপ করে, ঠিক যেমন একটি প্রচলিত অভিধান শব্দগুলিকে সংজ্ঞার সাথে ম্যাপ করে। একটি **সেট** হল একটি

৬.২ অভিধান

একটি অভিধান কীগুলিকে মানের সাথে সংযুক্ত করে। প্রতিটি কী একটি নির্দিষ্ট মানের সাথে ম্যাপ করে।

নিম্নলিখিত টেবিলে অভিধানের উদাহরণ রয়েছে যার মধ্যে রয়েছে তাদের কী, কী প্রকার, মান এবং মান প্রকার:

চারি আদর্শ	চাবি	মূল্যবোধ	মানের ধরণ
দেশের নাম	স্ট্র	ইন্টারনেট দেশ কোড	স্ট্র
দশমিক সংখ্যা int		রোমান সংখ্যা	স্ট্র
রাজ্যসমূহ	স্ট্র	কৃষি পণ্য	str এর তালিকা
হাসপাতালের রোগীরা	স্ট্র	গুরুত্বপূর্ণ লক্ষণ	ints এর টুপল এবং ভেসে ওঠে
বেসবল খেলোয়াড়	স্ট্র	ব্যাটিং গড়	ভাসমান
মেট্রিক পরিমাপ	স্ট্র	সংক্ষেপণ	স্ট্র
ইনভেন্টরি কোড	স্ট্র	মজুদে পরিমাণ	int-এর বিবরণ

নিক কিস

একটি অভিধানের কীগুলি অবশ্যই অপরিবর্তনীয় (যেমন স্ট্রিং, সংখ্যা বা টিপল) এবং অনন্য (অর্থাৎ কোনও সদৃশ নয়) হতে হবে। একাধিক কীগুলির মান একই হতে পারে, যেমন দুটি ভিন্ন ইনভেন্টরি কোড যার স্টকে একই পরিমাণ রয়েছে।

৬.২.১ একটি অভিধান তৈরি করা

আপনি কোকড়া বন্ধনীতে {}, কমা দ্বারা পৃথক করা কী-মান জোড়ার তালিকা, প্রতিটি ফর্ম কী: মান দিয়ে একটি অভিধান তৈরি করতে পারেন। আপনি একটি খালি অভিধান তৈরি করতে পারেন {}.

আসুন 'ফিনল্যান্ড', 'দক্ষিণ আফ্রিকা' এবং 'নেপাল' এই দেশের নাম কীগুলি এবং তাদের সংশ্লিষ্ট ইন্টারনেট দেশের কোড মান 'fi', 'za' এবং দিয়ে একটি অভিধান তৈরি করি।

'যেমন':

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: country_codes = {'ফিনল্যান্ড':	'ফাই', 'দক্ষিণ আফ্রিকা':	'জন্য',
...:	'নেপাল':	'যেমন'}
...:		
[2] তে: country_codes		
আউট[2]: {'ফিনল্যান্ড': 'ফাই', 'দক্ষিণ আফ্রিকা': 'জা', 'নেপাল':		'যেমন'}

যখন আপনি একটি অভিধান আউটপুট করেন, তখন এর কমা দ্বারা পৃথক করা কী-মান জোড়ার তালিকা সর্বদা কোকড়া বন্ধনীতে আবদ্ধ থাকে। যেহেতু অভিধানগুলি অ-ক্রমযুক্ত সংগ্রহ, তাই প্রদর্শন ক্রম অভিধানে কী-মান জোড়া যোগ করার ক্রম থেকে ভিন্ন হতে পারে। স্লিপেট [2] এর আউটপুটে কী-মান জোড়াগুলি যে ক্রমে সরিবেশ করা হয়েছিল সেই ক্রমে প্রদর্শিত হয়, তবে কী-মান জোড়ার ক্রমের উপর নির্ভর করে এমন কোড লিখবেন না।

অভিধান খালি কিনা তা নির্ধারণ করা

বিল্টইন ফাংশন len একটি অভিধানে কী-মান জোড়ার সংখ্যা প্রদান করে:

[3] তে: len(country_codes)	
আউট[3]: 3	

আপনি একটি অভিধানকে শর্ত হিসেবে ব্যবহার করতে পারেন এটি খালি কিনা তা নির্ধারণ করার জন্য—একটি খালি নয় এমন অভিধানকে True হিসেবে মূল্যায়ন করা হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[4] তে: যদি country_codes:
...:         print('country_codes খালি নয়')
...: অন্যথায়:
...:         মুদ্রণ করুন ('দেশ_কোড খালি')
...:
country_codes খালি নেই
```

একটি খালি অভিধান False হিসেবে মূল্যায়ন করে। এটি প্রদর্শনের জন্য, নিম্নলিখিত কোডে আমরা অভিধানের কী-মান জোড়া মুছে ফেলার জন্য method `clear` কল করি, তারপর snippet [6] এ আমরা snippet [4] প্রত্যাহার করি এবং পুনরায় কার্যকর করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[5] তে: country_codes.clear()
[6] তে: যদি country_codes: print('country_codes
...:         খালি নয়')
...: অন্যথায়:
...:         মুদ্রণ করুন ('দেশ_কোড খালি')
...:
country_codes খালি আছে
```

৬.২.২ অভিধানের মাধ্যমে পুনরাবৃত্তি করা

নিম্নলিখিত অভিধানটি মাসের নামের স্ট্রিংগুলিকে int মানের সাথে ম্যাপ করে যা সংশ্লিষ্ট মাসের দিনের সংখ্যা প্রতিনিধিত্ব করে। মনে রাখবেন যে একাধিক কীতে

একই মান:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1]: days_per_month = {'জানুয়ারী': 31, 'ফেব্রুয়ারী': 28, 'মার্চ': 31}
[2] তে: days_per_month
আউট[2]: {'জানুয়ারী': 31, 'ফেব্রুয়ারী': 28, 'মার্চ': 31}
```

আবার, অভিধানের স্ট্রিং উপস্থাপনা কী-মান জোড়াগুলিকে তাদের সন্নিবেশ ক্রমে দেখায়, তবে এটি নিশ্চিত নয় কারণ অভিধানগুলি অ-ক্রমযুক্ত। আমরা এই অধ্যায়ের পরে দেখাব কিভাবে কীগুলিকে সাজানো ক্রমে প্রক্রিয়া করতে হয়।

নিম্নলিখিত for স্টেটমেন্টটি `days_per_month` এর key-value জোড়ার মাধ্যমে পুনরাবৃত্তি করে।

অভিধান পদ্ধতি **আইটেম** প্রতিটি কী-মান জোড়াকে একটি টুপল হিসাবে ফেরত দেয়, যা আমরা মাস এবং দিনে আনপ্যাক করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[3] তে: মাসের জন্য , days_per_month.items() তে দিন: print(f'{month} has {days}' )
```

...:

...:

জানুয়ারী মাসে ৩১ দিন থাকে।

ফেব্রুয়ারি মাসে ২৮ দিন থাকে

মার্চ মাসে ৩১ দিন থাকে।

৬.২.৩ মৌলিক অভিধান পরিচালনা

এই বিভাগের জন্য, আসুন roman_numerals অভিধান তৈরি এবং প্রদর্শন করে শুরু করি। আমরা

ইচ্ছাকৃতভাবে 'X' কী-এর জন্য ভুল মান 100 প্রদান করেছি, যা আমরা শীঘ্ৰই সংশোধন করব:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: রোমান_সংখ্যা = {'আমি':
```

1, 'II': 2, 'III': 3, 'V': 5, 'X': 100}

```
[2] তে: রোমান_সংখ্যা
```

আউট[2]: {'I': 1, 'II': 2, 'III': 3, 'V': 5, 'X': 100}



একটি কী-এর সাথে যুক্ত মান অ্যাক্সেস করা

'V' কী-এর সাথে যুক্ত মানটি দেখা যাক:

```
[3] তে: roman_numerals['V']
```

আউট[3]: 5

একটি বিদ্যমান কী-মান জোড়ার মান আপডেট করা

আপনি একটি অ্যাসাইনমেন্ট স্টেটমেন্ট একটি কী-এর সংশ্লিষ্ট মান আপডেট করতে পারেন, যা আমরা এখানে 'X' কী-এর সাথে

সম্পর্কিত ভুল মান প্রতিস্থাপন করার জন্য করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[4] তে: roman_numerals['X'] = 10
```

```
[5] তে: রোমান_সংখ্যা
```

আউট[5]: {'I': 1, 'II': 2, 'III': 3, 'V': 5, 'X': 10}

একটি অস্তিত্বহীন কী-তে একটি মান নির্ধারণ করলে অভিধানে কী-মান জোড়া সন্নিবেশ করা হয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[6] তে: roman_numerals['L'] = 50
```

[7] তে: রোমান_সংখ্যা

আউট[7]: {'I': 1, 'II': 2, 'III': 3, 'V': 5, 'X': 10, 'L':

50}

স্ট্রিং কীগুলি কেস সংবেদনশীল। একটি অস্তিত্বহীন কীতে বরাদ্দ করলে একটি নতুন কী-মান জোড়া সন্নিবেশ করা হয়। এটি আপনার ইচ্ছামত হতে পারে, অথবা এটি একটি লজিক ত্রুটি হতে পারে।

একটি কী-মান জোড়া অপসারণ করা হচ্ছে

আপনি del স্টেটমেন্ট ব্যবহার করে অভিধান থেকে একটি কী-মান জোড়া মুছে ফেলতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[8] তে: del roman_numerals['III']
```

[9] তে: রোমান_সংখ্যা

আউট[9]: {'I': 1, 'II': 2, 'V': 5, 'X': 10, 'L': 50}

আপনি অভিধান পদ্ধতি পপ ব্যবহার করে একটি কী-মান জোড়াও সরাতে পারেন, যা সরানো কীটির মান প্রদান করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[10] তে: roman_numerals.pop('X')
```

আউট[10]: 10

[11] তে: রোমান_সংখ্যা

আউট[11]: {'I': 1, 'II': 2, 'V': 5, 'L': 50}

একটি অস্তিত্বহীন কী অ্যাক্সেস করার চেষ্টা করা হচ্ছে

একটি অস্তিত্বহীন কী অ্যাক্সেস করলে একটি KeyError দেখা দেয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
n [12]: রোমান_সংখ্যা ['III']
```

কী-এরর

ট্রেসব্যাক (সর্বশেষ কল শেষ)

<ipythoninput12ccd50c7f0c8b> <module>() > 1 roman_numerals['III']

এ

কী ক্রটি: 'III'

আপনি অভিধান পদ্ধতি `get` ব্যবহার করে এই ক্রটিটি প্রতিরোধ করতে পারেন, যা সাধারণত এটি ফেরত দেয় আর্গমেন্টের সংশ্লিষ্ট মান। যদি সেই কীটি খুঁজে না পাওয়া যায়, তাহলে `get` None রিটার্ন করবে। স্থিতে None রিটার্ন করলে IPython কিছুই প্রদর্শন করে না [13]। যদি আপনি `get` এর জন্য দ্বিতীয় আর্গমেন্ট নির্দিষ্ট করেন, তাহলে কীটি খুঁজে না পাওয়া গেলে এটি সেই মান রিটার্ন করবে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[13] তে: `roman_numerals.get('III')`[14] তে: `roman_numerals.get('III', 'III অভিধানে নেই')`

[14]: 'III অভিধানে নেই'

[15] তে: `roman_numerals.get('V')`

আউট[15]: 5

একটি অভিধানে একটি নির্দিষ্ট কী আছে কিনা তা পরীক্ষা করা

একটি অভিধানে একটি নির্দিষ্ট কী আছে কিনা তা নির্ধারণ করতে `in` এবং `not in` অপারেটরের নির্ধারণ করতে পারে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[16] তে: `রোমান_সংখ্যায় 'V'`

আউট[16]: সত্য

[17] তে: `রোমান_সংখ্যায় 'III'`

আউট[17]: মিথ্যা

[18] তে: `'III' রোমান_সংখ্যায় নয়`

আউট[18]: সত্য

৬.২.৪ অভিধান পদ্ধতি কী এবং মান

পূর্বে, আমরা অভিধানের কী-মান জোড়ার টিপলের মাধ্যমে পুনরাবৃত্তি করার জন্য অভিধান পদ্ধতি আইটেম ব্যবহার করতাম।

একইভাবে, পদ্ধতি কী এবং মানগুলি যথাক্রমে কেবল একটি অভিধানের কী বা মানের মাধ্যমে পুনরাবৃত্তি করার জন্য ব্যবহার করা

যেতে পারে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] মাসে: মাস = {'জানুয়ারী': 1, 'ফেব্রুয়ারী': 2, 'মার্চ': 3}

[2] তে: month_name এর জন্য months.keys() তে :

...: মুদ্রণ (মাসের_নাম, শেষ = ')

...:

জানুয়ারি ফেব্রুয়ারি মার্চ

[3] তে: মাসের সংখ্যার জন্য মাসের মান ():

...: মুদ্রণ (মাস_সংখ্যা, শেষ = ')

...:

১ ২ ৩

অভিধানের দৃশ্য

অভিধান পদ্ধতির আইটেম, কী এবং মান প্রতিটি অভিধানের ডেটার একটি ভিউ প্রদান করে।

যখন আপনি কোনও ভিউয়ের উপর পুনরাবৃত্তি করেন, তখন এটি অভিধানের বর্তমান বিষয়বস্তু "দেখে" - এর ডেটার নিজস্ব কপি থাকে না।

ভিউগুলি অভিধানের ডেটার নিজস্ব কপি বজায় রাখে না তা দেখানোর জন্য, প্রথমে কী দ্বারা ফেরত আসা ভিউটি মাস_ভিউ ডেরিয়েবলে সংরক্ষণ করা যাক, তারপর পুনরাবৃত্তি করা যাক

এটা:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: মাস_ভিউ = মাস.কি()

[5] তে: মাসের_দৃশ্য কী-এর জন্য :

...: মুদ্রণ (কী, শেষ = ' ')

...:

জানুয়ারি ফেব্রুয়ারি মার্চ

এরপর, মাসের সাথে একটি নতুন কী-মান জোড়া যোগ করা যাক এবং আপডেট করা অভিধানটি প্রদর্শন করা যাক:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6] মাসে: মাস [ডিসেম্বর] = 12

[7] মাসে: মাস

আউট[7]: {'জানুয়ারি': 1, 'ফেব্রুয়ারী': 2, 'মার্চ': 3,

'ডিসেম্বর': 12}

এখন, আবার months_view এর মাধ্যমে পুনরাবৃত্তি করা যাক। উপরে আমরা যে কীটি যোগ করেছি তা প্রকৃতপক্ষে প্রদর্শিত হচ্ছে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[8] তে: মাসের_দৃশ্যের জন্য কী :
...:           মুদ্রণ (কী, শেষ = ' ')
...:
জানুয়ারি ফেব্রুয়ারি মার্চ ডিসেম্বর
```

ডিউয়ের মাধ্যমে পুনরাবৃত্তি করার সময় অভিধান পরিবর্তন করবেন না। পাইথন স্ট্যান্ডার্ড লাইব্রেরি ডকুমেন্টেশনের ধারা 4.10.1 অনুসারে, হয় আপনি একটি RuntimeError পাবেন্ত অথবা লুপটি ডিউয়ের সমস্ত মান প্রক্রিয়া নাও করতে পারে।

ৱ <https://docs.python.org/3/library/stdtypes.html#dictionary-বস্তু>

অভিধান কী, মান এবং কী-মান জোড়াগুলিকে তালিকায় রূপান্তর করা

আপনার মাঝে মাঝে অভিধানের কী, মান বা কী-মান জোড়ার তালিকার প্রয়োজন হতে পারে। এই ধরনের তালিকা পেতে, কী, মান বা আইটেম দ্বারা প্রদত্ত ডিউটি বিল্টইন তালিকা ফাংশনে পাস করুন। এই তালিকাগুলি পরিবর্তন করলে সংশ্লিষ্ট অভিধানটি পরিবর্তন হয় না:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[9] তে: list(months.keys())
আউট[9]: ['জানুয়ারি', 'ফেব্রুয়ারি', 'মার্চ', 'ডিসেম্বর']

[10] তে: তালিকা (মাস.মান ())
আউট[10]: [1, 2, 3, 12]

[11] তে: তালিকা (months.items())
আউট[11]: [('জানুয়ারী', ১), ('ফেব্রুয়ারী', ২), ('মার্চ', ৩), ('ডিসেম্বর', ১২)]
```

সাজানো ক্রমে কীগুলি রোসেস করা হচ্ছে

সাজানো ক্রমে কীগুলি প্রক্রিয়া করার জন্য, আপনি নিম্নরূপ সাজানো বিল্টইন ফাংশন ব্যবহার করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[12] তে: month_name এর জন্য sorted (months.keys()):
...:           print(month_name, end=' ')
...:
ফেব্রুয়ারি ডিসেম্বর জানুয়ারী মার্চ
```

৬.২.৫ অভিধানের তুলনা

তুলনা অপারেটর == এবং != ব্যবহার করে দুটি অভিধানের বিষয়বস্তু একই নাকি ভিন্ন তা নির্ধারণ করা যেতে পারে। যদি উভয় অভিধানের কী-মান জোড়া একই থাকে, তাহলে একটি সমান (==) তুলনা সত্য হিসাবে মূল্যায়ন করা হয়, প্রতিটি অভিধানে সেই কী-মান জোড়াগুলি যে ক্রমেই যোগ করা হয়েছে তা নির্বিশেষে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: country_capitals1 = {'বেলজিয়াম':  
...:  
...:  
...:  
  
[2] তে: country_capitals2 = {'নেপাল':  
...:  
'উরুগুয়ে':  
...:  
  
[3] তে: country_capitals3 = {'হাইতি': 'PortauPrince',  
...:  
...:  
  
[4] তে: country_capitals1 == country_capitals2  
আউট[4]: মিথ্যা  
  
[5] তে: country_capitals1 == country_capitals3  
আউট[5]: সত্য  
  
[6] এর মধ্যে: country_capitals1 != country_capitals2 আউট[6]: সত্য
```

৬.২.৬ উদাহরণ: শিক্ষার্থীর গ্রেডের অভিধান

নিচের স্ক্রিপ্টটি একজন প্রশিক্ষকের গ্রেড বইকে একটি অভিধান হিসেবে উপস্থাপন করে যা প্রতিটি শিক্ষার্থীর নাম (একটি স্ট্রিং) কে তিনটি পরীক্ষায় সেই শিক্ষার্থীর গ্রেড ধারণকারী পূর্ণসংখ্যার তালিকার সাথে ম্যাপ করে। লুপের প্রতিটি পুনরাবৃত্তিতে যা ডেটা প্রদর্শন করে (লাইন 13-17), আমরা একটি কী-মান জোড়া ভেরিয়েবলের নাম এবং গ্রেড ধারণকারী একজন শিক্ষার্থীর নাম এবং তিনটি গ্রেডের সংশ্লিষ্ট তালিকার মধ্যে আনপ্যাক করি। লাইন 14 একটি নির্দিষ্ট শিক্ষার্থীর গ্রেডের মোট সংখ্যা নির্ধারণের জন্য বিল্টইন ফাংশন sum ব্যবহার করে, তারপর লাইন 15 সেই শিক্ষার্থীর গড় গণনা করে এবং সেই শিক্ষার্থীর গড়কে সেই শিক্ষার্থীর গ্রেডের সংখ্যা (len(গ্রেড)) দিয়ে ভাগ করে প্রদর্শন করে। লাইন 16-17 যথাক্রমে চারটি শিক্ষার্থীর গ্রেডের মোট সংখ্যা এবং সমস্ত শিক্ষার্থীর গ্রেডের সংখ্যা ট্র্যাক করে। লাইন 19 সমস্ত শিক্ষার্থীর গ্রেডের ক্লাস গড় প্রিন্ট করে

সব পরীক্ষা।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```

3 যেগীর_বই = {
8   'সুসান': [92, 85, 100],
5   'এডুয়ার্ড': [83, 95, 79],
6   'আজিজি': [91, 89, 82],
7   'পাঞ্চিপা': [97, 91, 92]
8 }
9

10টি মোট_গ্রেড_মোট = ০ ১১ টি_গ্রেড_গণনা = ০

12

নামের জন্য ১৩ , grade_book.items() তে গ্রেড :
14   মোট = যোগফল(গ্রেড) মূদ্রণ(f' {name} এর
15   গড় হল {total/len(গ্রেড)::2f}') all_grades_total += মোট all_grades_count += len(গ্রেড)
16
17
18

19 print(f"ক্লাসের গড় হল: {all_grades_total / all_grades_count:.2f}")

```

The screenshot shows a code editor window with a red border around the code area. The code calculates the total and count of grades for each student and then prints the average grade for the class.

```

কোড ইমেজ দেখতে এখানে ক্লিক করুন
-----
```

```

সুসানের গড় ৯২.৩০
এডুয়ার্ডের গড় ৮৫.৬৭
আজিজির গড় ৮৭.৩৩
পাঞ্চিপার গড় ৯৩.৩০
ক্লাসের গড় হল: ৮৯.৬৭

```

৬.২.৭ উদাহরণ: শব্দ গণনা

২

^২ প্রকাশিত রচনা বিশ্লেষণের জন্য প্রায়শই শব্দ ফ্রিকোয়েন্সি গণনার মতো কৌশল ব্যবহার করা হয়।

উদাহরণস্বরূপ, কিছু লোক বিশ্বাস করে যে উইলিয়াম শেক্সপিয়ারের রচনাগুলি আসলে স্যার ফ্রান্সিস বেকন, ক্রিস্টোফার মার্লি বা অন্যদের দ্বারা লেখা হতে পারে। শেক্সপিয়ারের রচনাগুলির সাথে তাদের রচনাগুলির শব্দ ফ্রিকোয়েন্সি তুলনা করলে লেখার ধরণে মিল খুঁজে পাওয়া যেতে পারে। প্রাকৃতিক ভাষা প্রক্রিয়াকরণ (NLP) অধ্যায়ে অন্যান্য ডকুমেন্ট বিশ্লেষণ কৌশলগুলি দেখুন।

নিচের ক্ষিটে একটি স্ট্রিং-এ প্রতিটি শব্দের উপস্থিতির সংখ্যা গণনা করার জন্য একটি অভিধান তৈরি করে। লাইন ৪-৫ একটি স্ট্রিং টেক্সট তৈরি করে যা আমরা শব্দে বিভক্ত করব—একটি প্রক্রিয়া যা স্ট্রিংকে টোকেনাইজিং নামে পরিচিত। পাইথন স্বয়ংক্রিয়ভাবে বন্ধনীতে হোয়াইটস্পেস দ্বারা পৃথক করা স্ট্রিংগুলিকে সংযুক্ত করে। লাইন ৭ একটি খালি অভিধান তৈরি করে। অভিধানের কীগুলি হবে অনন্য শব্দ এবং এর মানগুলি হবে প্রতিটি শব্দ টেক্সটে কতবার প্রদর্শিত হবে তার পূর্ণসংখ্যা গণনা।

```

১ # চিত্র০৬_০২.পিআই
২ """একটি স্ট্রিংকে টোকেনাইজ করা এবং অনন্য শব্দ গণনা করা।"""
৩
৪টি লেখা = ('এটি বেশ কয়েকটি শব্দ সহ নমুনা লেখা')
৫           'এটি কিছু ভিন্ন শব্দ সহ আরও নমুনা লেখা'
৬
৭টি শব্দের সংখ্যা =
৮
৯ # প্রতিটি অনন্য শব্দের সংখ্যা গণনা করুন
text.split() তে শব্দের জন্য ১০ :
১১      যদি শব্দ word_counts- এ থাকে :
১২          word_counts[word] += ১ # বিদ্যমান কীভ্যালু জোড়া আপডেট করুন
১৩      অন্যথায়:
১৪          word_counts[word] = ১ # নতুন কীভ্যালু জোড়া সন্নিবেশ করান
১৫
১৬টি মুদ্রণ (f'{"শব্দ":<১২}COUNT')
১৭
শব্দের জন্য ১৮ , সাজানোর মধ্যে গণনা করুন (word_counts.items()):
১৯      মুদ্রণ করুন (f'{শব্দ:<12}{গণনা}')
২০
২১টি print('\nঅনন্য শব্দের সংখ্যা:', len(word_counts))

```

শব্দ	COUNT টি
ভিন্ন ১	
হল	২
আরও	১
নমুনা	২
বেশ কিছু	১
কিছু	১
টেক্স্ট	২
এই	২
সঙ্গে	২
শব্দ	২
অনন্য শব্দের সংখ্যা:	১০টি

লাইন ১০ স্ট্রিং মেথড স্প্লিট কল করে টেক্স্টকে টোকেনাইজ করে , যা শব্দগুলিকে আলাদা করে।

পদ্ধতির ডিলিমিটার স্ট্রিং আর্গুমেন্ট ব্যবহার করে। যদি আপনি একটি আর্গুমেন্ট প্রদান না করেন,

split একটি স্পেস ব্যবহার করে। পদ্ধতিটি টোকেনের একটি তালিকা (অর্থাৎ, টেক্স্টের শব্দ) প্রদান করে।

১০-১৪ নম্বর লাইন শব্দের তালিকার মধ্য দিয়ে পুনরাবৃত্তি করে। প্রতিটি শব্দের জন্য, লাইন ১১ নির্ধারণ করে

সেই শব্দটি (কী) ইতিমধ্যেই অভিধানে আছে কিনা। যদি তাই হয়, তাহলে ১২ নম্বর লাইনটি বৃদ্ধি করে

শব্দের গণনা; অন্যথায়, ১৪ নম্বর লাইনে সেই শব্দের জন্য একটি নতুন কী-মান জোড়া সন্নিবেশ করানো হবে যার মধ্যে একটি

১৬-২১ নং লাইনে ফলাফলগুলিকে দুটি কলামের টেবিলে সংক্ষিপ্ত করা হয়েছে যেখানে প্রতিটি শব্দ এবং তার সংশ্লিষ্ট গণনা। ১৮ এবং ১৯ লাইনের for স্টেটমেন্টটি পুনরাবৃত্তি করে অভিধানের কী-মান জোড়া। এটি প্রতিটি কী এবং মানকে ডেরিয়েবল শব্দে আনপ্যাক করে এবং গণনা করুন, তারপর দুটি কলামে প্রদর্শন করুন। লাইন 21 অনন্য সংখ্যা প্রদর্শন করে শব্দ।

পাইথন স্ট্যান্ডার্ড লাইব্রেরি মডিউল সংগ্রহ

পাইথন স্ট্যান্ডার্ড লাইব্রেরিতে ইতিমধ্যেই গণনা কার্যকারিতা রয়েছে যা আমরা ১০-১৪ লাইনের অভিধান এবং লুপ ব্যবহার করে বাস্তবায়িত। মডিউল সংগ্রহগুলিতে কাউন্টার টাইপ থাকে, যা একটি পুনরাবৃত্তিযোগ্য গ্রহণ করে এবং সারসংক্ষেপ করে এর উপাদানগুলি। আসুন পূর্ববর্তী স্লিপটি কোডের কয়েকটি লাইনে পুনরায় প্রয়োগ করি কাউন্টার:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: সংগ্রহ থেকে আমদানি কাউন্টার

[2] তে: text = ('এটি বেশ কয়েকটি শব্দ সহ নমুনা টেক্স্ট')

```
...:  
    'এটি কিছু ভিন্ন শব্দ সহ আরও নমুনা লেখা')  
...:
```

[3] তে: কাউন্টার = কাউন্টার(text.split())

[4] তে: শব্দের জন্য, sorted(counter.items()) তে গণনা করুন:

```
...:  
    মুদ্রণ করুন (f'{শব্দ:<12}{গণনা}')  
...:  
ভিন্ন ১  
হল ২  
আরও ১  
নমুনা ২  
বেশ কিছু ১  
কিছু ১  
টেক্স্ট ২  
এই ২  
সঙ্গে ২  
শব্দ ২
```

[5] তে: print('Number of unique key:', len(counter.keys()))

অনন্য কী সংখ্যা: ১০টি

স্লিপট [3] কাউন্টার তৈরি করে, যা দ্বারা ফেরত আসা স্ট্রিংগুলির তালিকার সারসংক্ষেপ করে

text.split()। স্লিপট [4]-এ, কাউন্টার মেথড আইটেম প্রতিটি স্ট্রিং এবং তার

associated count কে tuple হিসেবে ব্যবহার করি। এই tuples এর তালিকা পেতে আমরা builtin function sorted ব্যবহার করি।
www.EBooksWorld.ir

উর্ধ্বক্রমানুসারে। ডিফল্টভাবে সাজানো হলে, টুপলগুলিকে তাদের প্রথম উপাদান অনুসারে সাজানো হয়। যদি সেগুলি অভিন্ন হয়, তাহলে এটি দ্বিতীয় উপাদানটিকে দেখায়, ইত্যাদি। for স্টেটমেন্টটি পুনরাবৃত্তি করে ফলস্বরূপ সাজানো তালিকার উপর, প্রতিটি শব্দ এবং গণনা দুটি কলামে প্রদর্শিত হবে।

৬.২.৮ অভিধান পদ্ধতি আপডেট

আপনি অভিধান পদ্ধতি আপডেট ব্যবহার করে কী-মান জোড়া সন্নিবেশ এবং আপডেট করতে পারেন। প্রথমে, একটি খালি country_codes অভিধান তৈরি করা যাক:

```
[1] তে: country_codes = {}
```

নিম্নলিখিত আপডেট কলটি সন্নিবেশ বা আপডেট করার জন্য কী-মান জোড়ার একটি অভিধান গ্রহণ করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[2]: country_codes.update({'দক্ষিণ আফ্রিকা': 'za'})
```

```
[3] তে: country_codes  
আউট[3]: {'দক্ষিণ আফ্রিকা": "za'}
```

মেথড আপডেট কীওয়ার্ড আর্গুমেন্টগুলিকে কী-মান জোড়ায় রূপান্তর করে সন্নিবেশ করাতে পারে। নিম্নলিখিত কলটি স্বয়ংক্রিয়ভাবে অস্ট্রেলিয়া প্যারামিটার নামটিকে 'অস্ট্রেলিয়া' স্ট্রিং কীতে রূপান্তরিত করে এবং 'ar' মানটিকে সেই কী-এর সাথে সংযুক্ত করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[4] তে: country_codes.update(Australia='ar')
```

```
[5] তে: country_codes  
আউট[5]: {'দক্ষিণ আফ্রিকা": "za", 'অস্ট্রেলিয়া": "ar'}
```

স্লিপেট [4] অস্ট্রেলিয়ার জন্য একটি ভুল দেশের কোড প্রদান করেছে। 'অস্ট্রেলিয়া'-এর সাথে সম্পর্কিত মান আপডেট করার জন্য আরেকটি কীওয়ার্ড আর্গুমেন্ট ব্যবহার করে এটি সংশোধন করা যাক:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[6] তে: country_codes.update(Australia='au')
```

```
[7] তে: country_codes  
আউট[7]: {'দক্ষিণ আফ্রিকা": "za", 'অস্ট্রেলিয়া": "au'}
```

মেথড আপডেটে কী-মান জোড়া ধারণকারী একটি পুনরাবৃত্ত বস্ত্রও পাওয়া যেতে পারে, যেমন দুটি উপাদান টিপলের তালিকা।

৬.২.৯ অভিধানের বোধগম্যতা

অভিধানের বোধগম্যতা দ্রুত অভিধান তৈরির জন্য একটি সুবিধাজনক স্বরলিপি প্রদান করে, প্রায়শই একটি অভিধানের সাথে অন্য অভিধান ম্যাপ করে। উদাহরণস্বরূপ, অনন্য মান সহ একটি অভিধানে, আপনি কী-মান জোড়া বিপরীত করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] মাসে: মাস = {'জানুয়ারী': 1, 'ফেব্রুয়ারী': 2, 'মার্চ': 3}
```

```
[2] তে: মাস2 = {সংখ্যা: নামের জন্য নাম, মাসের মধ্যে সংখ্যা.আইটেম ()}
```

```
[3]: মাস2
```

```
আউট[3]: {১: 'জানুয়ারী', ২: 'ফেব্রুয়ারী', ৩: 'মার্চ'}
```

কোকড়া বন্ধনী একটি অভিধানের বোধগম্যতাকে সীমাবদ্ধ করে, এবং for ক্লজের বাম দিকের এক্সপ্রেশনটি key: value ফর্মের একটি key-value জোড়া নির্দিষ্ট করে। এই কম্পিহেনশনটি months.items() এর মাধ্যমে পুনরাবৃত্তি হয়, প্রতিটি key-value জোড়া টুপলকে ভেরিয়েবলের name এবং number-এ আনপ্যাক করে। number: name এক্সপ্রেশনটি key এবং value কে বিপরীত করে, তাই নতুন অভিধানটি মাসের সংখ্যাগুলিকে মাসের নামের সাথে ম্যাপ করে।

যদি মাসের মধ্যে ডুপ্লিকেট মান থাকে? যেহেতু এগুলো মাস-এ কী হয়ে যায়, তাই একটি ডুপ্লিকেট কী সন্ধিবেশ করার চেষ্টা করলে বিদ্যমান কী-এর মান আপডেট হয়। তাই যদি 'ফেব্রুয়ারী' এবং 'মার্চ' উভয়ই মূলত ২-তে ম্যাপ করা হয়, তাহলে পূর্ববর্তী কোডটি তৈরি হত

```
{১: 'জানুয়ারী', ২: 'মার্চ'}
```

একটি অভিধানের বোধগম্যতা একটি অভিধানের মানগুলিকে নতুন মানগুলিতে ম্যাপ করতে পারে। নিম্নলিখিত বোধগম্যতা নাম এবং গ্রেডের তালিকার একটি অভিধানকে নাম এবং গ্রেডপয়েন্ট গড়ের অভিধানে রূপান্তরিত করে। k এবং v ভেরিয়েবলগুলি সাধারণত কী এবং মান বোঝায়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[4] তে: গ্রেড = {'স্থ': [98, 87, 94], 'ব্ব': [84, 95, 91]}
```

```
[5] তে: grades2 = {k: sum(v) / k এর জন্য len(v), grades.items() তে v }
```

```
[6] তে: গ্রেড2
```

```
আউট[6]: {'সু': 93.0, 'বব': 90.0}
```

এই কম্পিহেনশন `grades.items()` দ্বারা প্রদত্ত প্রতিটি টুপলকে `k` (নাম) এবং `v` (গ্রেডের তালিকা) তে আনপ্যাক করে। তারপর, কম্পিহেনশন `k` কী এবং `sum(v) / len(v)` এর মানের সাথে একটি নতুন কী-মান জোড়া তৈরি করে, যা তালিকার উপাদানগুলির গড় করে।

৬.৩ সেট

একটি সেট হল অনন্য মানের একটি অ-ক্রমিক সংগ্রহ। সেটগুলিতে কেবল অপরিবর্তনীয় বস্তু থাকতে পারে, যেমন স্ট্রিং, ইন্ট, ফ্লোট এবং টিপল যেখানে কেবল অপরিবর্তনীয় উপাদান থাকে। যদিও সেটগুলি পুনরাবৃত্তিযোগ্য, সেগুলি ক্রম নয় এবং বর্গাকার বন্ধনী দিয়ে সূচীকরণ এবং স্লাইসিং সমর্থন করে না, []। অভিধানগুলি স্লাইসিং সমর্থন করে না।

কোকড়া ব্রেস দিয়ে একটি সেট তৈরি করা

নিম্নলিখিত কোডটি `colors` নামে স্ট্রিংগুলির একটি সেট তৈরি করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[1] তে: colors = {'লাল', 'কমলা', 'হলুদ', 'সবুজ', 'লাল', 'নীল'}
```

```
[2] তে: রঙ
```

```
[2]: {'নীল', 'সবুজ', 'কমলা', 'লাল', 'হলুদ'}
```

লক্ষ্য করুন যে ডুপ্লিকেট স্ট্রিং 'লাল' উপেক্ষা করা হয়েছে (কোনও ক্রটি না করে)। সেটের একটি গুরুত্বপূর্ণ ব্যবহার হল ডুপ্লিকেট এলিমিনেশন, যা একটি সেট তৈরি করার সময় স্বয়ংক্রিয়ভাবে ঘটে। এছাড়াও, ফলাফল সেটের মানগুলি স্লিপেট [1]-এ তালিকাভুক্ত একই ক্রমে প্রদর্শিত হয় না। যদিও রঙের নামগুলি সাজানো ক্রমে প্রদর্শিত হয়, সেটগুলি অক্রমযুক্ত। আপনার এমন কোড লেখা উচিত নয় যা তাদের উপাদানগুলির ক্রমের উপর নির্ভর করে।

একটি সেটের দৈর্ঘ্য নির্ধারণ করা

বিল্টইন লেন ফাংশন ব্যবহার করে আপনি একটি সেটে আইটেমের সংখ্যা নির্ধারণ করতে পারেন:

```
[3] তে: len(রঙ)
```

```
আউট[3]: 5
```

একটি সেটে একটি মান আছে কিনা তা পরীক্ষা করা

আপনি `in` ব্যবহার করে একটি সেটে একটি নির্দিষ্ট মান আছে কিনা তা পরীক্ষা করতে পারেন এবং `in` ব্যবহার করে না

অপারেটর:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] ভাষায়: রঙে 'লাল'

আউট[4]: সত্য

[5] তে: রঙে 'বেগুনি'

আউট[5]: মিথ্যা

[6] তে: 'বেগুনি' রঙে নয়

আউট[6]: সত্য

একটি সেটের মাধ্যমে পুনরাবৃত্তি করা

সেটগুলি পুনরাবৃত্তিযোগ্য, তাই আপনি প্রতিটি সেট উপাদানকে for লুপ দিয়ে প্রক্রিয়া করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[7] তে: রঙের জন্য রঙে :

...
মুদ্রণ (রঙ.উপরের (), শেষ = '
...
লাল সবুজ হলুদ শীল কমলা

সেটগুলি অক্রমিক, তাই পুনরাবৃত্তির ক্রমের কোনও তাঁপর্য নেই।

বিল্ট-ইন সেট ফাংশন ব্যবহার করে একটি সেট তৈরি করা

আপনি বিল্ট-ইন সেট ফাংশন ব্যবহার করে অন্য মানের সংগ্রহ থেকে একটি সেট তৈরি করতে পারেন — এখানে আমরা একটি তালিকা তৈরি করি যাতে বেশ কয়েকটি ডুপ্লিকেট পূর্ণসংখ্যার মান থাকে এবং সেই তালিকাটিকে সেটের আর্গমেন্ট হিসেবে ব্যবহার করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[8] তে: সংখ্যা = তালিকা (পরিসর (10)) + তালিকা (পরিসর (5))

[9] তে: সংখ্যা

আউট[9]: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 0, 1, 2, 3, 4]

[10] তে: সেট(সংখ্যা)

আউট[10]: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

যদি আপনার একটি খালি সেট তৈরি করতে হয়, তাহলে আপনাকে খালি বন্ধনী সহ সেট ফাংশনটি ব্যবহার করতে হবে, খালি বন্ধনীর পরিবর্তে, {}, যা একটি খালি অভিধান উপস্থাপন করে:

[11] তে: সেট()

আউট[11]: সেট()

পাইথনের একটি খালি অভিধান ({}) এর স্ট্রিং উপস্থাপনার সাথে বিভ্রান্তি এড়াতে পাইথন একটি খালি সেটকে set() হিসাবে প্রদর্শন করে।

ফ্রোজেনসেট: একটি অপরিবর্তনীয় সেট টাইপ

সেটগুলি পরিবর্তনযোগ্য—আপনি উপাদান যোগ করতে এবং অপসারণ করতে পারেন, তবে সেট উপাদানগুলিকে অবশ্যই অপরিবর্তনীয় হতে হবে। অতএব, একটি সেটে উপাদান হিসাবে অন্য সেট থাকতে পারে না। একটি **হিমায়িত সেট** হল একটি অপরিবর্তনীয় সেট—আপনার তৈরি করার পরে এটি পরিবর্তন করা যাবে না, তাই একটি সেটে উপাদান হিসেবে ফ্রোজেনসেট থাকতে পারে। বিল্ট-ইন ফাংশন **ফ্রোজেনসেট** যেকোনো পুনরাবৃত্ত থেকে একটি ফ্রোজেনসেট তৈরি করে।

৬.৩.১ সেটের তুলনা

সেট তুলনা করার জন্য বিভিন্ন অপারেটর এবং পদ্ধতি ব্যবহার করা যেতে পারে। নিম্নলিখিত সেটগুলিতে একই মান রয়েছে, তাই == True প্রদান করে এবং != False প্রদান করে।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: {1, 3, 5} == {3, 5, 1}

আউট[1]: সত্য

[2] তে: {1, 3, 5} != {3, 5, 1}

আউট[2]: মিথ্যা

অপারেটর পরীক্ষা করে যে বাম দিকের সেটটি ডান দিকের সেটের **সঠিক উপসেট** কিনা - অর্থাৎ, বাম অপারেটের সমস্ত উপাদান ডান অপারেটে রয়েছে এবং সেটগুলি সমান নয়:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3] তে: {1, 3, 5} < {3, 5, 1}

আউট[3]: মিথ্যা

[4] তে: {1, 3, 5} < {7, 3, 5, 1}

আউট[4]: সত্য

<= অপারেটর পরীক্ষা করে যে বাম দিকের সেটটি ডান দিকের সেটের **অনুপযুক্ত উপসেট** কিনা —অর্থাৎ, বাম অপারেটের সমস্ত উপাদান ডান অপারেটে রয়েছে এবং সেটগুলি সমান হতে পারে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[5] তে: $\{1, 3, 5\} \leq \{3, 5, 1\}$

আউট[5]: সত্য

[6] তে: $\{1, 3\} \leq \{3, 5, 1\}$

আউট[6]: সত্য

আপনি set পদ্ধতি **issubset** ব্যবহার করে একটি অনুপযুক্ত উপসেট পরীক্ষা করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[7] তে: $\{1, 3, 5\}.issubset(\{3, 5, 1\})$

আউট[7]: সত্য

[8] তে: $\{1, 2\}.issubset(\{3, 5, 1\})$

আউট[8]: মিথ্যা

> অপারেটর পরীক্ষা করে যে বাম দিকের সেটটি ডান দিকের সেটের **সঠিক সুপারসেট** কিনা —অর্থাৎ, ডান অপারেন্টের সমষ্ট উপাদান বাম অপারেন্টে রয়েছে এবং বাম অপারেন্টে আরও উপাদান রয়েছে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[9] তে: $\{1, 3, 5\} > \{3, 5, 1\}$

আউট[9]: মিথ্যা

[10] তে: $\{1, 3, 5, 7\} > \{3, 5, 1\}$

আউট[10]: সত্য

\geq = অপারেটর পরীক্ষা করে যে বাম দিকের সেটটি ডান দিকের সেটের **অনুপযুক্ত সুপারসেট** কিনা —অর্থাৎ, ডান অপারেন্টের সমষ্ট উপাদান বাম অপারেন্টে রয়েছে এবং সেটগুলি সমান হতে পারে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[11] তে: $\{1, 3, 5\} \geq \{3, 5, 1\}$

আউট[11]: সত্য

[12] তে: $\{1, 3, 5\} \geq \{3, 1\}$

আউট[12]: সত্য

[13] তে: $\{1, 3\} \geq \{3, 1, 7\}$

আউট[13]: মিথ্যা

আপনি set পদ্ধতি issuperset ব্যবহার করে একটি অনুপযুক্ত সুপারসেট পরীক্ষা করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

ইন [18]: `{1, 3, 5}.issuperset({3, 5, 1})`

আউট[14]: সত্য

ইন [15]: `{1, 3, 5}.issuperset({3, 2})`

আউট[15]: মিথ্যা

issubset অথবা issuperset এর আর্গমেন্ট যেকোনো পুনরাবৃত্ত হতে পারে। যখন এই পদ্ধতিগুলির যেকোনো একটি একটি ননসেট পুনরাবৃত্ত আর্গমেন্ট পায়, তখন এটি প্রথমে পুনরাবৃত্তকে একটি সেটে রূপান্তর করে, তারপর অপারেশনটি সম্পাদন করে।

৬.৩.২ গাণিতিক সেট ক্রিয়াকলাপ

এই অংশে সেট টাইপের গাণিতিক অপারেটর |, &, এবং ^ এবং সংশ্লিষ্ট পদ্ধতিগুলি উপস্থাপন করা হয়েছে।

ইউনিয়ন

দুটি সেটের মিলন হলো এমন একটি সেট যা উভয় সেটের সমস্ত অনন্য উপাদান নিয়ে গঠিত। আপনি | অপারেটর ব্যবহার করে অথবা সেট টাইপের ইউনিয়ন পদ্ধতি ব্যবহার করে মিলন গণনা করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: `{1, 3, 5} | {2, 3, 4}`

আউট[1]: `{1, 2, 3, 4, 5}`

[2] তে: `{1, 3, 5}.union([20, 20, 3, 40, 40])`

আউট[2]: `{1, 3, 5, 20, 40}`

বাইনারি সেট অপারেটরের অপারেন্ড, যেমন |, উভয়ই সেট হতে হবে। সংশ্লিষ্ট সেট পদ্ধতিগুলি যেকোনো পুনরাবৃত্ত বস্তুকে আর্গমেন্ট হিসেবে গ্রহণ করতে পারে—আমরা একটি তালিকা পাস করেছি। যখন একটি গাণিতিক সেট পদ্ধতি একটি ননসেট পুনরাবৃত্ত যুক্তি গ্রহণ করে, তখন এটি প্রথমে পুনরাবৃত্ত বস্তুকে একটি সেটে রূপান্তর করে, তারপর গাণিতিক ক্রিয়াকলাপ প্রয়োগ করে। আবার, যদিও নতুন সেটের স্ট্রিং উপস্থাপনাগুলি মানগুলিকে উর্ধ্বমুখী করে দেখায়, আপনার এমন কোড লেখা উচিত নয় যা এর উপর নির্ভর করে।

ছেদ

দুটি সেটের ছেদস্থল হল এমন একটি সেট যার মধ্যে রয়েছে সমস্ত অনন্য উপাদান যা দুটি

সেটগুলির মধ্যে মিল রয়েছে। আপনি & অপারেটরের সাহায্যে অথবা সেট টাইপের ছেদ পদ্ধতির সাহায্যে ছেদ গণনা করতে পারেন :

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3] তে: {1, 3, 5} এবং {2, 3, 4}

আউট[3]: {3}

[4] তে: {1, 3, 5}.intersection([1, 2, 2, 3, 3, 4, 4])

আউট[4]: {1, 3}

পার্থক্য

দুটি সেটের মধ্যে পার্থক্য হল এমন একটি সেট যা বাম অপারেটরের উপাদানগুলি নিয়ে গঠিত যা ডান অপারেটের সাহায্যে অথবা সেট টাইপের **পার্থক্য** পদ্ধতি ব্যবহার করে **পার্থক্য** গণনা করতে পারেন :

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[5] তে: {1, 3, 5} {2, 3, 4}

আউট[5]: {1, 5}

[6] তে: {1, 3, 5, 7}.difference([2, 2, 3, 3, 4, 4])

আউট[6]: {1, 5, 7}

প্রতিসম পার্থক্য

দুটি সেটের মধ্যে **প্রতিসম পার্থক্য** হল এমন একটি সেট যা উভয় সেটের উপাদানগুলি নিয়ে গঠিত যা একে অপরের সাথে মিল নেই। আপনি প্রতিসম গণনা করতে পারেন

^ অপারেটরের সাথে অথবা সেট টাইপের **সিমেট্রিক_ডিফারেন্সের** সাথে পার্থক্য

পদ্ধতি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[7] তে: {1, 3, 5}

^ {2, 3, 8}

আউট[7]: {1, 2, 4, 5}

[8] তে: {1, 3, 5, 7}.symmetric_difference([2, 2, 3, 3, 4, 4])

আউট[8]: {1, 2, 4, 5, 7}

বিচ্ছিন্ন

দুটি সেটের মধ্যে যদি কোন সাধারণ উপাদান না থাকে তাহলে সেগুলি **বিচ্ছিন্ন** হবে। তুমি নির্ধারণ করতে পারো

সেট টাইপের **isdisjoint** পদ্ধতিতে এটি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[9] তে: `{1, 3, 5}.isdisjoint({2, 4, 6})`

আউট[9]: সত্য

[10] তে: `{1, 3, 5}.isdisjoint({4, 6, 1})`

আউট[10]: মিথ্যা

৬.৩.৩ পরিবর্তনযোগ্য সেট অপারেটর এবং পদ্ধতি

পূর্ববর্তী বিভাগে উপস্থাপিত অপারেটর এবং পদ্ধতিগুলির প্রতিটির ফলাফল একটি নতুন সেট।

এখানে আমরা অপারেটর এবং বিদ্যমান সেট পরিবর্তনকারী পদ্ধতিগুলি নিয়ে আলোচনা করব।

পরিবর্তনযোগ্য গাণিতিক সেট অপারেশন

অপারেটর | এর মতো, **union augmented assignment** |= একটি সেট union অপারেশন সম্পাদন করে, কিন্তু |= তার বাম অপারেন্ট পরিবর্তন করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: সংখ্যা = `{1, 3, 5}`

[2] তে: সংখ্যা |= `{2, 3, 4}`

[3] তে: সংখ্যা

আউট[3]: `{1, 2, 3, 4, 5}`

একইভাবে, সেট টাইপের **আপডেট** পদ্ধতিটি একটি ইউনিয়ন অপারেশন সম্পাদন করে যার উপর সেটটি বলা হয় সেটিকে সংশোধন করে—আর্গুমেন্টটি পুনরাবৃত্তিযোগ্য যেকোনো হতে পারে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[4] তে: `numbers.update(range(10))`

[5] তে: সংখ্যা

আউট[5]: `{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}`

অন্যান্য পরিবর্তনযোগ্য সেট পদ্ধতিগুলি হল:

- ছেদ অগমেন্টেড অ্যাসাইনমেন্ট &=

- পার্থক্য বর্ধিত অ্যাসাইনমেন্ট =

- প্রতিসম পার্থক্য বর্ধিত অ্যাসাইনমেন্ট ^=

এবং পুনরাবৃত্তিযোগ্য যুক্তি সহ তাদের সংলিঙ্গ পদ্ধতিগুলি হল:

- ইন্টারসেকশন_আপডেট
- পার্থক্য_আপডেট
- প্রতিসম_পার্থক্য_আপডেট

উপাদান যোগ এবং অপসারণের পদ্ধতি

যদি আর্গেমেন্টটি ইতিমধ্যে সেটে না থাকে, তাহলে Set মেথড add তার আর্গেমেন্ট সন্নিবেশ করায়;
অন্যথায়, সেটটি অপরিবর্তিত থাকে:

কোড ইমেজ দেখতে এখানে লিংক করুন

[6] তে: numbers.add(17)

[7] তে: numbers.add(3)

[8] তে: সংখ্যা

আউট[8]: {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 17}

সেট মেথড **রিমুভ** সেট থেকে তার আর্গেমেন্টটি সরিয়ে দেয়—একটি KeyError ঘটে যদি
মান সেটে নেই:

কোড ইমেজ দেখতে এখানে লিংক করুন

[9] তে: numbers.remove(3)

[10] তে: সংখ্যা

আউট[10]: {0, 1, 2, 4, 5, 6, 7, 8, 9, 17}

মেথড **ডিসকার্ড** সেট থেকে তার আর্গেমেন্টটি সরিয়ে দেয় কিন্তু যদি মানটি সেটে না থাকে তবে ব্যতিক্রম ঘটায়
না।

আপনি একটি ইচ্ছামত সেট এলিমেন্ট মুছে ফেলতে পারেন এবং pop দিয়ে এটি ফেরত দিতে পারেন, কিন্তু
সেটগুলি অক্রমিক, তাই আপনি জানেন না কোন এলিমেন্টটি ফেরত দেওয়া হবে:

কোড ইমেজ দেখতে এখানে লিংক করুন

[11] তে: numbers.pop()

আউট[11]: 0

[12] তে: সংখ্যা

আউট[12]: {১, ২, ৪, ৫, ৬, ৭, ৮, ৯, ১৭}

পপ কল করার সময় সেটটি খালি থাকলে একটি KeyError ঘটে।

অবশ্যে, method **clear** যে সেটিতে এটি বলা হয় সেট খালি করে:

[13] তে: numbers.clear()

[14] তে: সংখ্যা

আউট[14]: সেট()

৬.৩.৪ সেট কম্পিউটেশন

অভিধানের বোধগম্যতার মতো, আপনি কোকড়া বন্ধনীতে সেট বোধগম্যতা সংজ্ঞায়িত করেন। আসুন তালিকার সংখ্যাগুলিতে শুধুমাত্র অনন্য জোড় মানগুলি ধারণ করে একটি নতুন সেট তৈরি করি:

কোড ইমেজ দেখতে এখানে লিংক করুন

[1] তে: সংখ্যা = [১, ২, ২, ৩, ৪, ৫, ৬, ৬, ৭, ৮, ৯, ১০, ১০]

[2] তে: evens = { যদি আইটেম % ২ == ০ হয় তবে সংখ্যায় আইটেমের জন্য আইটেম }

[3] তে: সংখ্যা

আউট[3]: {২, ৪, ৬, ৮, ১০}

৬.৪ তথ্য বিজ্ঞানের ভূমিকা: গতিশীল দৃশ্যায়ন

পূর্ববর্তী অধ্যায়ের ডেটা সামগ্রের ভূমিকা বিভাগে ভিজুয়ালাইজেশন চালু করা হয়েছে। আমরা একটি ছয়-পার্শ্বযুক্ত ডাইরোলিং সিমুলেটেড করেছি এবং Seaborn এবং Matplotlib ভিজুয়ালাইজেশন লাইব্রেরি ব্যবহার করে একটি প্রকাশনার মানের স্ট্যাটিক বার প্লট তৈরি করেছি যা প্রতিটি রোল মানের ফ্রিকোয়েন্সি এবং শতাংশ দেখায়। এই বিভাগে, আমরা গতিশীল ভিজুয়ালাইজেশনের মাধ্যমে জিনিসগুলিকে "জীৱন্ত" করে তুলি।

বৃহৎ সংখ্যার সূত্র

র্যান্ডম নাম্বার জেনারেশন চালু করার সময়, আমরা উল্লেখ করেছি যে যদি র্যান্ডম মডিউলের র্যান্ডোজে ফাংশন সত্যিই র্যান্ডমভাবে পূর্ণসংখ্যা তৈরি করে, তাহলে নির্দিষ্ট পরিসরের প্রতিটি সংখ্যার প্রতিবার ফাংশনটি কল করার সময় নির্বাচিত হওয়ার সম্ভাবনা (অথবা সম্ভাবনা) সমান থাকে। একটি ছয় পার্শ্বযুক্ত ডাইয়ের জন্য, প্রতিটি মান 1 থেকে 6 ষষ্ঠ বার হওয়া উচিত, তাই এই মানগুলির যেকোনো একটি হওয়ার সম্ভাবনা $1/6$ বা প্রায়

ম

১৬.৬৬৭%।

পরবর্তী অংশে, আমরা একটি গতিশীল (অর্থাৎ, অ্যানিমেটেড) ডাইরোলিং সিমুলেশন স্ক্রিপ্ট তৈরি এবং কার্যকর করব। সাধারণভাবে, আপনি দেখতে পাবেন যে আমরা যত বেশি রোল করার চেষ্টা করব, মোট রোলের প্রতিটি ডাই মানের শতাংশ ততই 16.667% এর কাছাকাছি চলে আসবে এবং বারগুলির উচ্চতা ধীরে ধীরে প্রায় একই হয়ে যাবে। এটি বৃহৎ সংখ্যার সূত্রের একটি প্রকাশ।

৬.৪.১ ডায়নামিক ডিজ্যুয়ালাইজেশন কীভাবে কাজ করে

পূর্ববর্তী অধ্যায়ের Intro to Data Science বিভাগে Seaborn এবং Matplotlib দিয়ে তৈরি প্লটগুলি সিমুলেশন সম্পর্ক হওয়ার পরে নির্দিষ্ট সংখ্যক ডাই রোলের ফলাফল বিশ্লেষণ করতে আপনাকে সাহায্য করে। এই বিভাগটি Matplotlib অ্যানিমেশন মডিউলের FuncAnimation ফাংশনের সাহায্যে সেই কোডটিকে উন্নত করে, যা বার প্লটকে গতিশীলভাবে আপডেট করে। আপনি বার, ডাই ফ্রিকোয়েন্সি এবং শতাংশ "জীবিত হয়ে" দেখতে পাবেন, রোলগুলি হওয়ার সাথে সাথে ক্রমাগত আপডেট হচ্ছে।

অ্যানিমেশন ফ্রেম

FuncAnimation একটি ফ্রেমবাইফ্রেম অ্যানিমেশন চালায়। প্রতিটি অ্যানিমেশন ফ্রেম একটি প্লট আপডেটের সময় কী পরিবর্তন করা উচিত তা নির্দিষ্ট করে। সময়ের সাথে সাথে এই আপডেটগুলির অনেকগুলিকে একত্রিত করলে অ্যানিমেশন প্রভাব তৈরি হয়। আপনি একটি ফাংশন সংজ্ঞায়িত করে FuncAnimation-এ পাস করার মাধ্যমে প্রতিটি ফ্রেম কী প্রদর্শন করবে তা নির্ধারণ করেন।

প্রতিটি অ্যানিমেশন ফ্রেমে থাকবে:

- নির্দিষ্ট সংখ্যক বার পাশা ঘূরিয়ে নিন (১ থেকে যত খুশি ততবার), প্রতিটি রোলের সাথে ডাই ফ্রিকোয়েন্সি আপডেট করুন,
- বর্তমান প্লটটি পরিষ্কার করুন,
- আপডেট করা ফ্রিকোয়েন্সিগুলির প্রতিনিধিত্বকারী বারগুলির একটি নতুন সেট তৈরি করুন, এবং
- প্রতিটি বারের জন্য নতুন ফ্রিকোয়েন্সি এবং শতাংশের টেক্সট তৈরি করুন।

সাধারণত, প্রতি সেকেন্ডে বেশি ফ্রেম প্রদর্শন করলে মসৃণ অ্যানিমেশন পাওয়া যায়। উদাহরণস্বরূপ, দ্রুত গতিশীল উপাদান সহ ভিডিও গেমগুলি প্রতি সেকেন্ডে কমপক্ষে 30টি ফ্রেম প্রদর্শন করার চেষ্টা করে-

দ্বিতীয় এবং প্রায়শই আরও বেশি। যদিও আপনি অ্যানিমেশন ফ্রেমের মধ্যে মিলিসেকেন্ডের সংখ্যা নির্দিষ্ট করবেন, প্রতি সেকেন্ডে ফ্রেমের প্রকৃত সংখ্যা প্রতিটি ফ্রেমে আপনার কাজের পরিমাণ এবং আপনার কম্পিউটারের প্রসেসরের গতির উপর নির্ভর করতে পারে।

এই উদাহরণে প্রতি ৩০ মিলিসেকেন্ডে একটি অ্যানিমেশন ফ্রেম দেখানো হয়েছে—যা প্রতি সেকেন্ডে প্রায় ৩০ (১০০০ / ৩০) ফ্রেম দেয়। অ্যানিমেশনের উপর তাদের প্রভাব কীভাবে পড়ে তা দেখার জন্য আরও বড় এবং ছোট মান চেষ্টা করুন। সেরা ভিজুয়ালাইজেশন তৈরিতে পরীক্ষা-নিরীক্ষা গুরুত্বপূর্ণ।

RollDieDynamic.py চালানো হচ্ছে

পূর্ববর্তী অধ্যায়ের ডেটা সায়েন্সের ভূমিকা বিভাগে, আমরা স্ট্যাটিক ভিজুয়ালাইজেশনটি ইন্টারেক্টিভভাবে তৈরি করেছি যাতে আপনি দেখতে পারেন যে প্রতিটি বিবৃতি কার্যকর করার সময় কোডটি বার প্লটকে কীভাবে আপডেট করে। চূড়ান্ত ফ্রিকোয়েলি এবং শতাংশ সহ প্রকৃত বার প্লটটি কেবল একবার আঁকা হয়েছিল।

এই গতিশীল ভিজুয়ালাইজেশনের জন্য, স্ক্রিনের ফলাফল ঘন ঘন আপডেট হয় যাতে আপনি অ্যানিমেশনটি দেখতে পারেন। অনেক কিছুই ক্রমাগত পরিবর্তিত হয় - বারগুলির দৈর্ঘ্য, বারগুলির উপরে ফ্রিকোয়েলি এবং শতাংশ, অক্ষের উপর ব্যবধান এবং লেবেল এবং প্লটের শিরোনামে দেখানো ডাই রোলের মোট সংখ্যা। এই কারণে, আমরা এই ভিজুয়ালাইজেশনটিকে ইন্টারেক্টিভভাবে বিকাশ করার পরিবর্তে একটি স্ক্রিপ্ট হিসাবে উপস্থাপন করি।

স্ক্রিপ্টটি দুটি কমান্ডলাইন আঙ্গুমেন্ট গ্রহণ করে:

- ফ্রেমের_সংখ্যা—প্রদর্শনের জন্য অ্যানিমেশন ফ্রেমের সংখ্যা। এই মানটি FuncAnimation গ্রাফটি আপডেট করার মোট সংখ্যা নির্ধারণ করে। প্রতিটি অ্যানিমেশন ফ্রেমের জন্য, FuncAnimation একটি ফাংশন কল করে যা আপনি সংজ্ঞায়িত করেন (এই উদাহরণে, আপডেট) যাতে প্লটটি কীভাবে পরিবর্তন করতে হয় তা নির্দিষ্ট করা যায়।
- রোলস_পার_ফ্রেম—প্রতিটি অ্যানিমেশন ফ্রেমে ডাই কতবার রোল করতে হবে। আমরা একটি লুপ ব্যবহার করে ডাইটি এই সংখ্যক বার রোল করব, ফলাফলগুলি সারসংক্ষেপ করব, তারপর বার এবং নতুন ফ্রিকোয়েলিগুলি প্রতিনিধিত্বকারী টেক্সট সহ গ্রাফটি আপডেট করব।

এই দুটি মান কীভাবে ব্যবহার করা হয় তা বুঝতে, নিম্নলিখিত কমান্ডটি বিবেচনা করুন:

```
ipython RollDieDynamic.py 6000 1
```

এই ক্ষেত্রে, FuncAnimation আমাদের আপডেট ফাংশনটি 6000 বার কল করে, প্রতি ফ্রেমে একটি ডাই রোল করে মোট 6000 রোল করে। এর ফলে আপনি বার, ফ্রিকোয়েলি এবং শতাংশ একবারে একটি রোল আপডেট দেখতে পারবেন। আমাদের সিস্টেমে, এই অ্যানিমেশনটি আপনাকে দেখাতে প্রায় 3.33 মিনিট (6000 ফ্রেম / 30 ফ্রেম প্রতি সেকেন্ড / 60 সেকেন্ড প্রতি মিনিট) সময় নিয়েছে।

স্ক্রিনে অ্যানিমেশন ফ্রেম প্রদর্শন করা ডাই রোলগুলির তুলনায় তুলনামূলকভাবে ধীর ইনপুট-আউটপুটবাটন অপারেশন, যা কম্পিউটারের অতি দ্রুত CPU গতিতে ঘটে। যদি আমরা প্রতি অ্যানিমেশন ফ্রেমে কেবল একটি ডাই রোল করি, তাহলে আমরা যুক্তিসঙ্গত সময়ে প্রচুর সংখ্যক রোল চালাতে সক্ষম হব না। এছাড়াও, অল্প সংখ্যক রোলের জন্য, ডাই শতাংশ মোট রোলের প্রত্যাশিত 16.667% এর সাথে একত্রিত হওয়ার সম্ভাবনা কম।

বৃহৎ সংখ্যার আইনটি কার্যকরভাবে দেখার জন্য, আপনি প্রতিটি অ্যানিমেশন ফ্রেমে ডাই আরও বার ঘূর্ণায়মান করে কার্যকর করার গতি বাড়াতে পারেন। নিম্নলিখিত কমান্ডটি বিবেচনা করুন:

```
ipython RollDieDynamic.py 10000 600
```

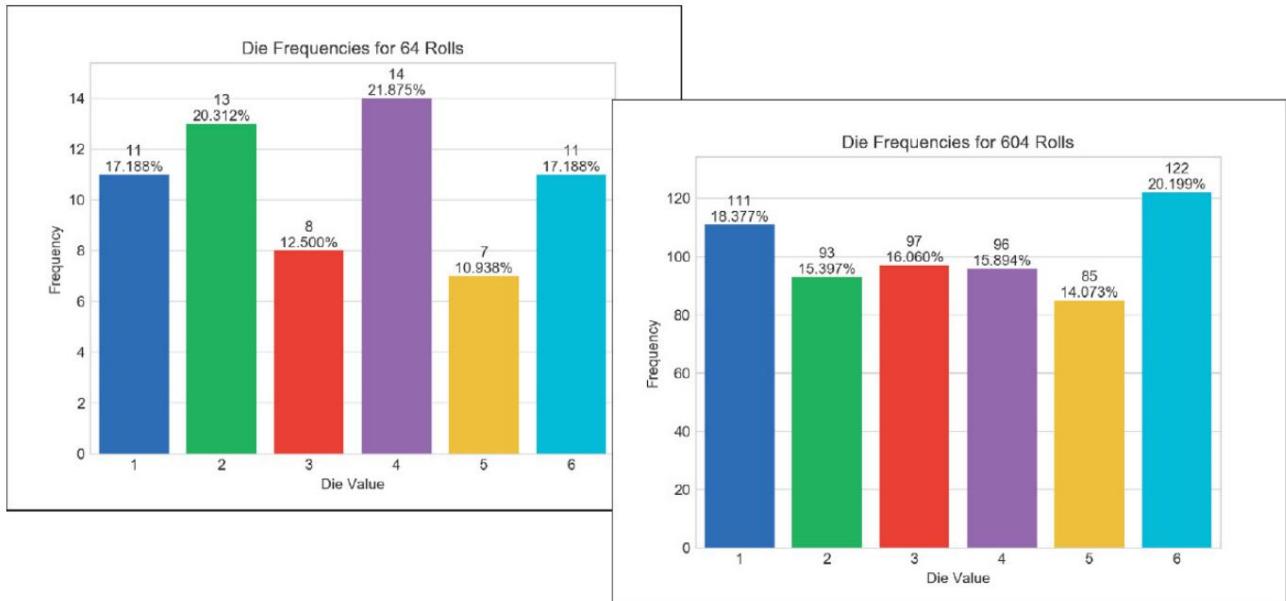
এই ক্ষেত্রে, FuncAnimation আমাদের আপডেট ফাংশনটিকে ১০,০০০ বার কল করবে, ৬০০ রোলসপারফ্রেম পারফর্ম করবে মোট ৬,০০০,০০০ রোলসের জন্য। আমাদের সিস্টেমে, এটি প্রায় ৫.৫৫ মিনিট (১০,০০০ ফ্রেম / ৩০ ফ্রেমপ্রতি সেকেন্ড / ৬০ সেকেন্ডপ্রতি মিনিট) সময় নেয়, কিন্তু প্রায় ১৮,০০০ রোলসপারসেকেন্ড (৩০ ফ্রেমপ্রতি সেকেন্ড * ৬০০ রোলস-পারফ্রেম) প্রদর্শন করে, তাই আমরা দ্রুত দেখতে পারি যে ফ্রিকোয়েন্সি এবং শতাংশগুলি তাদের প্রত্যাশিত মানের উপর একত্রিত হয় যা প্রতি ফেস প্রায় ১,০০০,০০০ রোলস এবং প্রতি ফেস ১৬.৬৬৭%।

রোল এবং ফ্রেমের সংখ্যা নিয়ে পরীক্ষা-নিরীক্ষা করুন যতক্ষণ না আপনি অনুভব করেন যে প্রোগ্রামটি আপনাকে ফলাফলগুলি সবচেয়ে কার্যকরভাবে কল্পনা করতে সাহায্য করছে। এটি চালানো দেখা এবং অ্যানিমেশনের মানের সাথে সন্তুষ্ট না হওয়া পর্যন্ত এটি পরিবর্তন করা মজাদার এবং তথ্যবহুল।

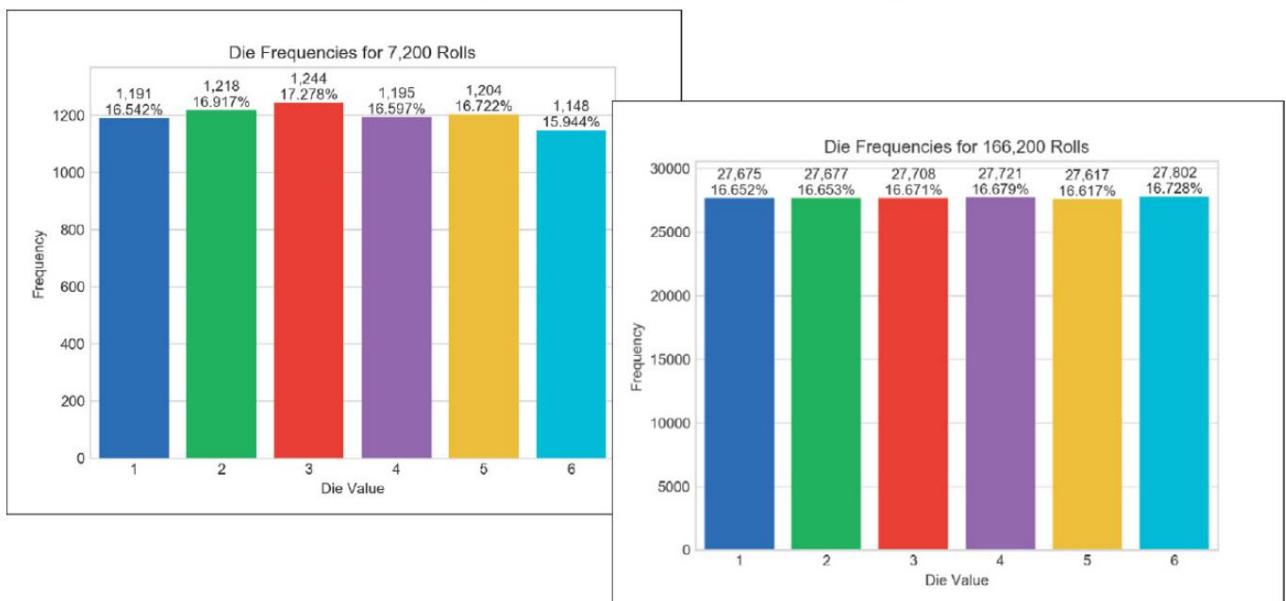
নমুনা সম্পাদন

দুটি নমুনা সম্পাদনের সময় আমরা নিম্নলিখিত চারটি স্ক্রিন ক্যাপচার নিয়েছি। প্রথমটিতে, স্ক্রিনগুলি মাত্র 64টি ডাই রোলের পরে গ্রাফ দেখায়, তারপর আবার 6000টি মোট ডাই রোলের মধ্যে 604টির পরে। সময়ের সাথে সাথে বারগুলি কীভাবে গতিশীলভাবে আপডেট হয় তা দেখার জন্য এই স্ক্রিপ্টটি লাইভ চালান। দ্বিতীয় সম্পাদনে, স্ক্রিন ক্যাপচারগুলি 7200টি ডাই রোলের পরে গ্রাফ দেখায় এবং আবার 6,000,000টি রোলের মধ্যে 166,200টির পরে। আরও রোলের সাহায্যে, আপনি বৃহৎ সংখ্যার আইন অনুসারে পূর্বাভাসিত 16.667% এর প্রত্যাশিত মানের সাথে শতাংশের কাছাকাছি দেখতে পাবেন।

Execute 6000 animation frames rolling the die once per frame:
`ipython RollDieDynamic.py 6000 1`



Execute 10,000 animation frames rolling the die 600 times per frame:
`ipython RollDieDynamic.py 10000 600`



.৪.২ একটি গতিশীল ডিজ্যুয়ালাইজেশন বাস্তবায়ন

এই বিভাগে আমরা যে স্ক্রিপ্টটি উপস্থাপন করছি তাতে পূর্ববর্তী অধ্যায়ের Intro to Data Science বিভাগে দেখানো একই Seaborn এবং Matplotlib বৈশিষ্ট্য ব্যবহার করা হয়েছে। আমরা Matplotlib এর অ্যানিমেশন ক্ষমতার সাথে ব্যবহারের জন্য কোডটি পুনর্গঠন করেছি।

Matplotlib অ্যানিমেশন মডিউল আমদানি করা হচ্ছে

আমরা মূলত এই উদাহরণে ব্যবহৃত নতুন বৈশিষ্ট্যগুলির উপর ফোকাস করি। লাইন 3 ম্যাটপ্লটলির অ্যানিমেশন মডিউল আমদানি করে।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
2 """ভাই ৱোলের ফ্রিকোয়েলিগুলির গতিশীল প্রাফিং!"""
3
```

```
matplotlib আমদানি অ্যানিমেশন থেকে 3
```

```
৪টি matplotlib.pyplot plt হিসেবে আমদানি করুন ৫টি র্যান্ডম হিসেবে
```

```
আমদানি করুন ৬টি seaborn
```

```
হিসেবে sns হিসেবে আমদানি করুন ৭টি import
```

```
sys
```

```
৮
```

ফাংশন আপডেট

৯-২৭ নং লাইনে FuncAnimation যে আপডেট ফাংশনটি প্রতি অ্যানিমেশন ফ্রেমে একবার কল করে তা সংজ্ঞায়িত করা হয়েছে। এই ফাংশনটিতে কমপক্ষে একটি আর্গুমেন্ট থাকতে হবে। ৯-১০ নং লাইনে ফাংশনের সংজ্ঞার শুরু দেখানো হয়েছে। প্যারামিটারগুলি হল:

- frame_number—FuncAnimation এর frames আর্গুমেন্ট থেকে পরবর্তী মান, যা আমরা কিছুক্ষণের মধ্যে আলোচনা করব। যদিও FuncAnimation এর জন্য এই প্যারামিটারটি থাকার জন্য আপডেট ফাংশন প্রয়োজন, আমরা এই আপডেট ফাংশনে এটি ব্যবহার করি না।
- রোলস—প্রতি অ্যানিমেশন ফ্রেমে ডাই ৱোলের সংখ্যা।
- মুখগুলি—গ্রাফের অক্ষ বরাবর লেবেল হিসেবে ব্যবহৃত ডাই ফেস মানগুলি।
- ফ্রিকোয়েলি—যে তালিকায় আমরা ডাই ফ্রিকোয়েলিগুলির সারসংক্ষেপ তুলে ধরছি।

আমরা পরবর্তী কয়েকটি উপধারায় ফাংশনের বাকি অংশ নিয়ে আলোচনা করব।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
৯ ডিফল্ট আপডেট (ফ্রেম_নম্বর, রোলস, ১০)
```

মুখ, ফ্রিকোয়েলি):

```
"""প্রতিটি অ্যানিমেশন ফ্রেমের জন্য বার প্লটের বিষয়বস্তু কনফিগার করে।"""

```

ফাংশন আপডেট: ডাই ৱোল করা এবং ফ্রিকোয়েলি তালিকা আপডেট করা

১২-১৩ নম্বর লাইনে ডাই ৱোলের সময় ৱোল করা হয় এবং প্রতিটি ৱোলের জন্য উপযুক্ত ফ্রিকোয়েলি উপাদান বৃদ্ধি করা হয়।

মনে রাখবেন যে আমরা সংশ্লিষ্ট ফ্রিকোয়েলি উপাদান বৃদ্ধি করার আগে ডাই মান (১ থেকে ৬) থেকে ১ বিয়োগ করি - যেমন আপনি দেখতে পাবেন, ফ্রিকোয়েলি হল একটি ছয়টি উপাদানের তালিকা (৩৬ নম্বর লাইনে সংজ্ঞায়িত), তাই এর সূচকগুলি ০ থেকে ৫।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
১১ # ৱোল ডাই এবং আপডেট ফ্রিকোয়েলি
```

```
১২ রেঞ্জে | এর জন্য (রোলস):
```

১৩

ফ্রিকোয়েলি [random.randrange(1, 7) 1] += 1

১৪

ফাংশন আপডেট: বার প্লট এবং টেক্সট কনফিগার করা

ফাংশন আপডেটের লাইন ১৬ matplotlib.pyplot মডিউলের `cla` (স্পষ্ট অক্ষ) কে কল করে।

নতুন বার প্লট এলিমেন্ট আঁকার আগে বিদ্যমান বার প্লট এলিমেন্টগুলো মুছে ফেলার ফাংশন

বর্তমান অ্যানিমেশন ফ্রেম। আমরা আগের লাইন 17-27 এ কোডটি নিয়ে আলোচনা করেছি

অধ্যায়ের ডেটা সায়েন্সের ভূমিকা অংশ। ১৭-২০ নম্বর লাইনে বার তৈরি করা হয়, বার প্লটের সেট করা হয়

শিরোনাম, `x` এবং `yaxis` লেবেল সেট করুন এবং ফ্রিকোয়েলির জন্য জায়গা তৈরি করার জন্য প্লটটি স্কেল করুন এবং

প্রতিটি বারের উপরে শতাংশের লেখা। ২৩-২৭ নম্বর লাইনে ফ্রিকোয়েলি এবং শতাংশের লেখা প্রদর্শিত হয়।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```

১৫      # আপডেট করা ভাই ফ্রিকোয়েলিগুলির জন্য প্লট পুনরায় কনফিগার করুন
১৬      plt.cla() # বর্তমান চিত্রের পুরাতন বিষয়বস্তু পরিষ্কার করুন
১৭      অক্ষ = sns.barplot(মুখ, ফ্রিকোয়েলি, প্যালেট = 'উজ্জ্বল')                                # বাকি আছে
১৮      axes.set_title( {sum(frequencies):,} রোলসের জন্য f'Die ফ্রিকোয়েলি' )
১৯      axes.set(xlabel='Die Value', ylabel='ফ্রিকোয়েলি')
২০      axes.set_ylim(top=max(frequencies) * 1.10) # 0% দ্বারা yaxis স্কেল করুন
২১
২২      # প্রতিটি প্যাচের উপরে প্রদর্শন ফ্রিকোয়েলি এবং শতাংশ (বার)
২৩      বারের জন্য , জিপে ফ্রিকোয়েলি (axes.patches, ফ্রিকোয়েলি):
২৪          টেক্সট_এক্স = বার.গেট_এক্স() + বার.গেট_উইথ() / ২.০
২৫          টেক্সট_ওয়াই = বার.গেট_উচ্চতা()
২৬          টেক্সট = f'{ফ্রিকোয়েলি:,}\n{ফ্রিকোয়েলি / যোগফল(ফ্রিকোয়েলি):.3%}'
২৭          axes.text(text_x, text_y, text, ha='center', va='bottom')
২৮

```

গ্রাফ কনফিগার করতে এবং অবস্থা বজায় রাখতে ব্যবহৃত ভেরিয়েবল

লাইন ৩০ এবং ৩১ স্ক্রিপ্টের কমান্ড লাইন পেতে sys মডিউলের argv তালিকা ব্যবহার করে।

যুক্তি। লাইন ৩৩ সির্বর 'হোয়াইটগ্রিড' স্টাইল নির্দিষ্ট করে। লাইন ৩৪ কল করে

matplotlib.pyplot মডিউলের `ফিগার` ফাংশনটি ফিগার অবজেক্টটি পেতে ব্যবহার করা হয় যেখানে

FuncAnimation অ্যানিমেশনটি প্রদর্শন করে। ফাংশনের আর্গুমেন্ট হল উইন্ডোর শিরোনাম।

আপনি শীত্রাই দেখতে পাবেন, এটি FuncAnimation এর প্রয়োজনীয় আর্গুমেন্টগুলির মধ্যে একটি। লাইন 35 তৈরি করে

প্লটের `x` অক্ষে প্রদর্শিত ভাই ফেস মান ১-৬ ধারণকারী একটি তালিকা। লাইন ৩৬ তৈরি করে

প্রতিটি উপাদান ০ তে শুরু করে ছয়টি উপাদানের ফ্রিকোয়েলি তালিকা—আমরা এটি আপডেট করি

প্রতিটি ভাই রোলের সাথে তালিকার গণনা।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
৩০ টি ফ্রেমের_সংখ্যা = int(sys.argv[1]) ৩১ টি রোলস_পার_ফ্রেম = int(sys.argv[2]) ৩২ টি
```

```
৩৩ sns.set_style('whitegrid') # ধূসর গ্রিড লাইন সহ সাদা পটভূমি ৩৪ চিত্র = plt.figure('একটি ছবি পার্শ্বযুক্ত ডাই ঘূর্ণায়মান') # অ্যানিমেশনের জন্য চিত্র  
৩৫ মান = তালিকা (পরিসীমা (1, 7)) # x অক্ষে প্রদর্শনের জন্য ডাই ফেস ৩৬ ফ্রিকোয়েলি =
```

```
[0] * 6 # ডাই ফ্রিকোয়েলির ছয়টি উপাদানের তালিকা
```

৩৭

অ্যানিমেশন মডিউলের FuncAnimation ফাংশনের সাথে সব মিলিয়ে

লাইন ৩৯-৪১ বার চার্টটি গতিশীলভাবে আপডেট করার জন্য Matplotlib অ্যানিমেশন মডিউলের FuncAnimation ফাংশনকে কল করে। ফাংশনটি অ্যানিমেশনের প্রতিনিধিত্বকারী একটি বস্তু ফেরত পাঠায়। যদিও এটি স্পষ্টভাবে ব্যবহার করা হয় না, আপনাকে অ্যানিমেশনের রেফারেন্স সংরক্ষণ করতে হবে; অন্যথায়, পাইথন তাংক্ষণিকভাবে অ্যানিমেশনটি বন্ধ করে দেয় এবং সিস্টেমে এর মেমোরি ফিরিয়ে দেয়।

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
৩৮ # ফাংশন আপডেট কল করে এমন অ্যানিমেশন কনফিগার করুন এবং শুরু করুন ৩৯ die_animation = animation.FuncAnimation( 40
```

```
চিত্র, আপডেট, পুনরাবৃত্তি = মিথ্যা, ফ্রেম = ফ্রেমের_সংখ্যা, ইন্টারভা = দূরত্ব = (প্রতি_ফ্রেমে_রোলস, মান, ফ্রিকোয়েলি))
```

৪১

৪২

```
৪৩ plt.show() # ডিসপ্লে উইন্ডো
```

FuncAnimation-এর দুটি প্রয়োজনীয় আঙ্গমেন্ট রয়েছে:

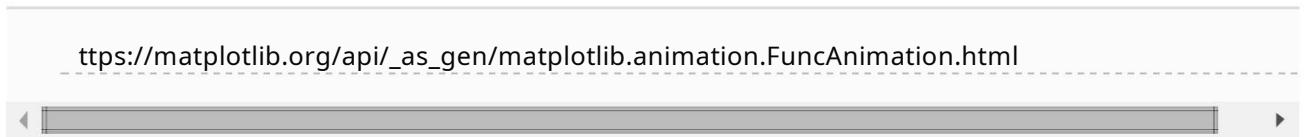
- চিত্র—চিত্র বস্তু যেখানে অ্যানিমেশনটি প্রদর্শন করা হবে, এবং
- আপডেট—প্রতি অ্যানিমেশন ফ্রেমে একবার কল করার ফাংশন।

এই ক্ষেত্রে, আমরা নিম্নলিখিত ঐচ্ছিক কীওয়ার্ড আঙ্গমেন্টগুলিও পাস করি:

- repeat—False নির্দিষ্ট সংখ্যক ফ্রেমের পরে অ্যানিমেশনটি বন্ধ করে দেয়। যদি True (ডিফল্ট) হয়, তাহলে অ্যানিমেশনটি সম্পূর্ণ হলে এটি শুরু থেকে পুনরায় চালু হয়।
- ফ্রেম—মোট অ্যানিমেশন ফ্রেমের সংখ্যা, যা FuncAnimation করবার আপডেট কল করবে তা নিয়ন্ত্রণ করে। একটি পূর্ণসংখ্যা পাস করা একটি রেঞ্জ পাস করার সমতুল্য—উদাহরণস্বরূপ, 600 মানে range(600)। আপডেটের জন্য প্রতিটি কলে প্রথম আঙ্গমেন্ট হিসেবে FuncAnimation এই রেঞ্জ থেকে একটি মান পাস করে।

- **ব্যবধান**— অ্যানিমেশনের মধ্যে মিলিসেকেন্ডের সংখ্যা (এই ক্ষেত্রে 33) ফ্রেম (ডিফল্ট ২০০)। প্রতিটি কল আপডেট করার পর, FuncAnimation পরবর্তী কল করার আগে ৩৩ মিলিসেকেন্ড অপেক্ষা করে।
- **fargs** ("ফাংশন আর্গুমেন্ট" এর সংক্ষিপ্ত রূপ)—FuncAnimation এর দ্বিতীয় আর্গুমেন্টে আপনার নির্দিষ্ট করা ফাংশনে পাস করার জন্য অন্যান্য আর্গুমেন্টের একটি টুপল। fargs tuple এ আপনার নির্দিষ্ট করা আর্গুমেন্টগুলি আপডেটের প্যারামিটার রোল, ফেস এবং ফ্রিকোয়েন্সি (লাইন ৯) এর সাথে সামঞ্জস্যপূর্ণ।

FuncAnimation এর অন্যান্য ঐচ্ছিক আর্গুমেন্টের তালিকার জন্য, দেখুন



অবশেষে, লাইন 43 উইন্ডোটি প্রদর্শন করে।

৬.৫ র্যাপ-আপ

এই অধ্যায়ে, আমরা পাইথনের অভিধান এবং সেট সংগ্রহ নিয়ে আলোচনা করেছি। আমরা একটি অভিধান কী তা বলেছি এবং বেশ কয়েকটি উদাহরণ উপস্থাপন করেছি। আমরা কী-মান জোড়ার বাক্য গঠন দেখিয়েছি এবং কোকড়া বন্ধনীতে কমা দ্বারা পৃথক করা কী-মান জোড়ার তালিকা সহ অভিধান তৈরি করতে কীভাবে সেগুলি ব্যবহার করতে হয় তা দেখিয়েছি, {}। আপনি অভিধান বোধগম্যতা সহ অভিধানও তৈরি করেছেন।

আপনি একটি কী-এর সাথে সম্পর্কিত মান পুনরুদ্ধার করতে এবং কী-মান জোড়া সন্নিবেশ এবং আপডেট করতে বর্গাকার বন্ধনী [] ব্যবহার করেছেন। আপনি একটি কী-এর সংশ্লিষ্ট মান পরিবর্তন করতে অভিধান পদ্ধতি আপডেটও ব্যবহার করেছেন। আপনি একটি অভিধানের কী, মান এবং আইটেমগুলির মাধ্যমে পুনরাবৃত্তি করেছেন।

তুমি অনন্য অপরিবর্তনীয় মানের সেট তৈরি করেছ। তুলনামূলক অপারেটরের সাথে সেট তুলনা করেছ, সেট অপারেটর এবং পদ্ধতির সাথে সেট একত্রিত করেছ, পরিবর্তনযোগ্য সেট অপারেশনের সাথে সেটের মান পরিবর্তন করেছ এবং সেট বোধগম্যতার সাথে সেট তৈরি করেছ। তুমি দেখেছ যে সেটগুলি পরিবর্তনযোগ্য। ফ্রেজেনসেটগুলি অপরিবর্তনীয়, তাই এগুলি সেট এবং ফ্রেজেনসেট উপাদান হিসাবে ব্যবহার করা যেতে পারে।

"ইন্ট্রো টু ডেটা সায়েন্স" বিভাগে, আমরা বৃহৎ সংখ্যার সূত্রকে "জীবন্ত" করে তোলার জন্য একটি গতিশীল বার প্লট সহ ডাইরোলিং সিমুলেশন উপস্থাপন করে আমাদের ডিজ্যুয়ালাইজেশন ভূমিকা অব্যাহত রেখেছি। এছাড়াও, পূর্ববর্তী অধ্যায়ের "ইন্ট্রো টু ডেটা সায়েন্স" বিভাগে দেখানো Seaborn এবং Matplotlib বৈশিষ্ট্যগুলির পাশাপাশি, আমরা একটি ফ্রেমবাইফ্রেম অ্যানিমেশন নিয়ন্ত্রণ করতে Matplotlib এর FuncAnimation ফাংশন ব্যবহার করেছি। FuncAnimation এমন একটি ফাংশনকে বলে যা আমরা সংজ্ঞায়িত করি যা প্রতিটি অ্যানিমেশন ফ্রেমে কী প্রদর্শন করতে হবে তা নির্দিষ্ট করে।

পরবর্তী অধ্যায়ে, আমরা জনপ্রিয় NumPy লাইব্রেরির সাহায্যে অ্যারে-ওরিয়েন্টেড প্রোগ্রামিং নিয়ে আলোচনা করব। আপনি দেখতে পাবেন, NumPy-এর ndarray সংগ্রহ Python-এর বিল্ট-ইন তালিকার সাহায্যে একই ধরণের অনেক ক্রিয়াকলাপ সম্পাদনের চেয়ে দুই ক্রম পর্যন্ত দ্রুততর হতে পারে। আজকের বিগ ডেটা অ্যাপ্লিকেশনগুলির জন্য এই ক্ষমতাটি কাজে আসবে।

প্লেলিস্ট

গল্প . NumPy সহ অ্যারে-ওরিয়েন্টেড প্রোগ্রামিং

উদ্দেশ্যমূলক মতামত

এই অধ্যায়ে আপনি পাবেন:

- তালিকা থেকে অ্যারে কীভাবে আলাদা তা জানুন।
অফার্স এবং ডিল
- numpy মডিউলের উচ্চ কর্মক্ষমতা সম্পর্ক ndarrays ব্যবহার করুন। highlights
- IPython %timeit ম্যাজিকের সাথে তালিকা এবং ndarray এর পারফরম্যান্সের তুলনা করুন। ettings দক্ষতার সাথে ডেটা সংরক্ষণ এবং
- পুনরুদ্ধার করতে ndarray ব্যবহার করুন।
সমর্থন
- ndarray তৈরি করুন এবং আরম্ভ করুন।
সাইন আউট
- পৃথক ndarray উপাদানগুলি পড়ুন।
ndarrays এর মাধ্যমে পুনরাবৃত্তি করুন।
- বহুমাত্রিক এন্ডারে তৈরি এবং পরিচালনা করুন।
সাধারণ ndarray ম্যানিপুলেশনগুলি সম্পাদন করুন।
- পান্তদের এক-মাত্রিক সিরিজ এবং দ্বি-মাত্রিক তৈরি এবং পরিচালনা করুন
ডেটাফ্রেম।
সিরিজ এবং ডেটাফ্রেম সূচকগুলি কাস্টমাইজ করুন।
- একটি সিরিজ এবং একটি ডেটাফ্রেমের ডেটার জন্য মৌলিক বর্ণনামূলক পরিসংখ্যান গণনা করুন।
পান্তাস আউটপুট ফর্ম্যাটিংয়ে ফ্রোটিংপয়েন্ট নম্বর নির্ভুলতা কাস্টমাইজ করুন।

রূপরেখা

.1 ভূমিকা

.2 বিদ্যমান ডেটা থেকে অ্যারে তৈরি করা

.3 অ্যারে বৈশিষ্ট্য

.4 নির্দিষ্ট মান দিয়ে অ্যারে পূরণ করা

.5 রেঞ্জ থেকে অ্যারে তৈরি করা

.6 তালিকা বনাম অ্যারে পারফরম্যান্স: %timeit প্রবর্তন করা হচ্ছে

.7 অ্যারে অপারেটর

.8 NumPy গণনা পদ্ধতি

.9 সার্বজনীন ফাংশন

.10 ইনডেক্সিং এবং স্লাইসিং

.11 বার দেখা হয়েছে: অগভীর কপি

.12টি গভীর কপি

.13 পুনঃআকৃতিকরণ এবং স্থানান্তর

.14 ডেটা সায়েন্সের ভূমিকা: পান্ডাস সিরিজ এবং ডেটাফ্রেম

.14.1 পান্ডাস সিরিজ

.14.2 ডেটাফ্রেম

.15 সারসংক্ষেপ

৭.১ ভূমিকা

NumPy (**Numerical Python**) লাইব্রেরিটি প্রথম ২০০৬ সালে আবির্ভূত হয় এবং এটি পাইথন অ্যারে বাস্তবায়নের জন্য পছন্দের। এটি একটি উচ্চ-কার্যক্ষমতাসম্পন্ন, সমৃদ্ধভাবে কার্যকরী n-মাত্রিক অ্যারে টাইপ অফার করে যার নাম **ndarray**, যা এখন থেকে আমরা এর সমার্থক, অ্যারে দ্বারা উল্লেখ করব। NumPy হল অনেকগুলি ওপেনসোর্স লাইব্রেরির মধ্যে একটি যা Anaconda Python ডিস্ট্রিবিউশন ইনস্টল করে। অ্যারেতে অপারেশনগুলি দুই ক্রম পর্যন্ত হয়

তালিকার তুলনায় দ্রুত। বিগড়েটা জগতে যেখানে অ্যাপ্লিকেশনগুলি বিপুল পরিমাণে অ্যারে-ভিত্তিক ডেটার উপর প্রচুর পরিমাণে প্রক্রিয়াকরণ করতে পারে, সেখানে এই কর্মক্ষমতা সুবিধাটি গুরুত্বপূর্ণ হতে পারে। libraries.io অনুসারে, 450 টিরও বেশি পাইথন লাইব্রেরি NumPy-এর উপর নির্ভরশীল। অনেক জনপ্রিয় ডেটা সায়েন্স লাইব্রেরি যেমন Pandas, SciPy (বৈজ্ঞানিক পাইথন) এবং Keras (গভীর শিক্ষার জন্য) NumPy-এর উপর নির্মিত বা নির্ভরশীল।

এই অধ্যায়ে, আমরা অ্যারের মৌলিক ক্ষমতাগুলি অন্বেষণ করব। তালিকাগুলির একাধিক মাত্রা থাকতে পারে। আপনি সাধারণত নেস্টেড লুপ দিয়ে বহুমাত্রিক তালিকা বা একাধিক for clause দিয়ে তালিকা বোঝার প্রক্রিয়া করেন। NumPy এর একটি শক্তি হল "অ্যারে-ওরিয়েন্টেড প্রোগ্রামিং", যা অ্যারে ম্যানিপুলেশনগুলিকে সংক্ষিপ্ত এবং সহজ করে তুলতে অভ্যন্তরীণ পুনরাবৃত্তি সহ ফাংশনাল স্টাইল প্রোগ্রামিং ব্যবহার করে, স্পষ্টভাবে প্রোগ্রাম করা লুপগুলির বাহ্যিক পুনরাবৃত্তির সাথে ঘটতে পারে এমন ধরণের বাগগুলি দূর করে।

এই অধ্যায়ের ডেটা সায়েন্সের ভূমিকা বিভাগে, আমরা পান্ডাস লাইব্রেরির আমদারের বহু-বিভাগীয় ভূমিকা শুরু করব যা আপনি অনেক ডেটা সায়েন্স কেস স্টাডি অধ্যায়ে ব্যবহার করবেন। বড় ডেটা অ্যাপ্লিকেশনগুলির প্রায়শই NumPy এর অ্যারের তুলনায় আরও নমনীয় সংগ্রহের প্রয়োজন হয় — এমন সংগ্রহ যা মিশ্র ডেটা টাইপ, কাস্টম ইনডেক্সিং, অনুপস্থিত ডেটা, এমন ডেটা যা ধারাবাহিকভাবে কাঠামোগত নয় এবং এমন ডেটা যা আপনার ব্যবহৃত ডেটাবেস এবং ডেটা বিশ্লেষণ প্যাকেজগুলির জন্য উপযুক্ত ফর্মগুলিতে ম্যানিপুলেট করা প্রয়োজন। আমরা পান্ডাস অ্যারের মতো এক-মাত্রিক সিরিজ এবং দ্বি-মাত্রিক ডেটাফ্রেমগুলি পরিচয় করিয়ে দেব এবং তাদের শক্তিশালী ক্ষমতা প্রদর্শন শুরু করব। এই অধ্যায়টি পড়ার পরে, আপনি চারটি অ্যারের মতো সংগ্রহের সাথে পরিচিত হবেন — তালিকা, অ্যারে, সিরিজ এবং ডেটাফ্রেম।

আমরা "গভীর শিক্ষা" অধ্যায়ে পঞ্চমটি—টেলর—যোগ করব।

৭.২ বিদ্যমান তথ্য থেকে অ্যারে তৈরি করা

NumPy ডকুমেন্টেশন [numpy মডিউলটিকে](#) np হিসেবে আমদানি করার পরামর্শ দেয় যাতে আপনি "np" দিয়ে এর সদস্যদের অ্যাক্সেস করতে পারেন:

```
[1] তে: np হিসেবে numpy আমদানি করুন
```

numpy মডিউল অ্যারে তৈরির জন্য বিভিন্ন ফাংশন প্রদান করে। এখানে আমরা [অ্যারে](#) ফাংশন ব্যবহার করি, যা একটি অ্যারে বা অন্যান্য উপাদানের সংগ্রহকে আর্গুমেন্ট হিসেবে গ্রহণ করে এবং আর্গুমেন্টের উপাদানগুলি ধারণকারী একটি নতুন অ্যারে প্রদান করে। আসুন একটি পাস করি

তালিকা:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

```
[2] তে: সংখ্যা = np.array([2, 3, 5, 7, 11])
```

অ্যারে ফাংশনটি তার আর্গুমেন্টের বিষয়বস্তু অ্যারেতে কপি করে। আসুন দেখি কোন ধরণের অবজেক্ট ফাংশন
অ্যারে ফেরত দেয় এবং এর বিষয়বস্তু প্রদর্শন করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[3] তে: টাইপ(সংখ্যা)

আউট[3]: numpy.ndarray

[4] তে: সংখ্যা

আউট[4]: অ্যারে([2, 3, 5,

৭, ১১])

মনে রাখবেন যে টাইপটি হল numpy.ndarray, কিন্তু সমস্ত অ্যারে "অ্যারে" হিসাবে আউটপুট হয়। একটি অ্যারে আউটপুট
করার সময়, NumPy প্রতিটি মানকে একটি কমা এবং একটি স্পেস দিয়ে পরবর্তী থেকে আলাদা করে এবং একই ফিল্ড প্রস্তু
ব্যবহার করে সমস্ত মানকে ডানদিকে সারিবদ্ধ করে। এটি সর্বাধিক সংখ্যক অক্ষর অবস্থান দখল করে এমন মানের উপর ভিত্তি করে
ক্ষেত্রের প্রস্তু নির্ধারণ করে। এই ক্ষেত্রে, মান 11 দুটি অক্ষর অবস্থান দখল করে, তাই সমস্ত মান দুটি অক্ষর ক্ষেত্রে ফর্ম্যাট করা
হয়। এই কারণেই [এবং

২.

বহুমাত্রিক যুক্তি

অ্যারে ফাংশনটি তার আর্গুমেন্টের মাত্রা কপি করে। আসুন tworowbythreecolumn তালিকা থেকে একটি অ্যারে তৈরি করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[5] তে: np.array([[1, 2, 3], [4, 5, 6]])

আউট[5]:

অ্যারে([[1, 2, 3], [4, 5, 6]])

NumPy অ্যারেগুলিকে অটোফরম্যাট করে, তাদের মাত্রার সংখ্যার উপর ভিত্তি করে, সারিবদ্ধ করে
প্রতিটি সারির মধ্যে কলাম।

৭.৩ অ্যারে অ্যাট্রিবিউটস

একটি অ্যারে অবজেক্ট এমন বৈশিষ্ট্য প্রদান করে যা আপনাকে এর গঠন এবং বিষয়বস্তু সম্পর্কে তথ্য আবিষ্কার করতে সক্ষম করে।
এই বিভাগে আমরা নিম্নলিখিত অ্যারেগুলি ব্যবহার করব:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: np হিসেবে numpy আমদানি করুন

[2] তে: পূর্ণসংখ্যা = np.array([[1, 2, 3], [4, 5, 6]])

[3] তে: পূর্ণসংখ্যা

আউট[3]:

অ্যারে([[1, 2, 3], [4, 5, 6]])

[4] তে: floats = np.array([0.0, 0.1, 0.2, 0.3, 0.4])

[5] তে: ভাসমান

আউট[5]: অ্যারে([0.

, 0.1, 0.2, 0.3, 0.8])

NumPy ফ্লোটিংপয়েন্ট মানের দশমিক বিল্ডুর ডানদিকে পরবর্তী 0s প্রদর্শন করে না।

একটি অ্যারের এলিমেন্টের ধরণ নির্ধারণ করা

অ্যারে ফাংশনটি একটি অ্যারের এলিমেন্টের ধরণ তার আঙ্গমেন্টের এলিমেন্ট থেকে নির্ধারণ করে।

আপনি একটি অ্যারের **dtype** অ্যাট্রিবিউট দিয়ে এলিমেন্টের ধরণ পরীক্ষা করতে পারেন:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

[6]: integers.dtype Out[6]: dtype('int64')

int32 কিছু প্ল্যাটফর্ম

[7] তে: floats.dtype

আউট[7]: dtype('float64')

পরবর্তী অংশে আপনি দেখতে পাবেন, বিভিন্ন arraycreation ফাংশন একটি **dtype** কীওয়ার্ড আঙ্গমেন্ট গ্রহণ করে যাতে আপনি একটি অ্যারের উপাদানের ধরণ নির্দিষ্ট করতে পারেন।

পারফরম্যান্সের কারণে, NumPy C প্রোগ্রামিং ভাষায় লেখা হয় এবং C এর ডেটা টাইপ ব্যবহার করে। ডিফল্টক্লাপে, NumPy পূর্ণসংখ্যাগুলিকে NumPy টাইপ int64 মান হিসাবে সংরক্ষণ করে—যা C তে 64bit (8byte) পূর্ণসংখ্যার সাথে সঙ্গতিপূর্ণ—এবং ফ্লোটিংপয়েন্ট সংখ্যাগুলিকে NumPy টাইপ float64 মান হিসাবে সংরক্ষণ করে—যা C তে 64bit (8byte) ফ্লোটিং-পয়েন্ট মানের সাথে সঙ্গতিপূর্ণ। আমাদের উদাহরণগুলিতে, সাধারণত আপনি int64, float64, bool (বুলিয়ানের জন্য) এবং অ-সংখ্যাসূচক ডেটার জন্য (যেমন স্ট্রিং) object টাইপগুলি দেখতে পাবেন। সমর্থিত প্রকারের সম্পূর্ণ তালিকা এখানে রয়েছে

<https://docs.scipy.org/doc/numpy/user/basics.types.html>

একটি অ্যারের মাত্রা নির্ধারণ করা

ndim অ্যাট্রিবিউটে একটি অ্যারের মাত্রার সংখ্যা থাকে এবং অ্যাট্রিবিউটের আকৃতিতে একটি টুপল থাকে যা একটি অ্যারের মাত্রা নির্দিষ্ট করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[8] তে: integers.ndim

আউট[8]: 2

[9] তে: floats.ndim

আউট[9]: 1

[10] তে: integers.shape Out[10]: (2, 3)

[11] তে: floats.shape

আউট[11]: (5,)

এখানে, পূর্ণসংখ্যার 2টি সারি এবং 3টি কলাম (6টি উপাদান) রয়েছে এবং floats এক-মাত্রিক, তাই স্লিপেট [11] একটি oneelement tuple (কমা দ্বারা নির্দেশিত) দেখায় যাতে floats-এর উপাদানের সংখ্যা (5) থাকে।

একটি অ্যারের উপাদান সংখ্যা এবং উপাদানের আকার নির্ধারণ করা

আপনি একটি অ্যারের মোট উপাদানের সংখ্যা, অ্যাট্রিবিউটের আকার এবং **itemsize** সহ প্রতিটি উপাদান সংরক্ষণের জন্য প্রয়োজনীয় বাইটের সংখ্যা দেখতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[12] তে: integers.size Out[12]: 6

সি কম্পাইলার ৩২ বিট ইন্ট ব্যবহার করে

আউট[13]: 8

[14] তে: floats.size

আউট[14]: 5

[15] তে: floats.itemsize

আউট[15]: 8

মনে রাখবেন যে পূর্ণসংখ্যার আকার হল আকৃতির টুপলের মানের গুণফল—তিনটি উপাদানের দুটি সারি, মোট ছয়টি উপাদানের জন্য। প্রতিটি ক্ষেত্রে, আইটেমসাইজ 8 হয় কারণ পূর্ণসংখ্যায় int64 মান থাকে এবং floats-এ float64 মান থাকে, যা প্রতিটিতে 8 বাইট থাকে।

একটি বহুমাত্রিক অ্যারের উপাদানগুলির মাধ্যমে পুনরাবৃত্তি করা

আপনি সাধারণত সংক্ষিপ্ত ফাংশনালস্টাইল প্রোগ্রামিং কৌশল ব্যবহার করে অ্যারেগুলি পরিচালনা করবেন।

তবে, যেহেতু অ্যারেগুলি পুনরাবৃত্তিযোগ্য, আপনি যদি চান তবে বাহ্যিক পুনরাবৃত্তি ব্যবহার করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[16] তে: পূর্ণসংখ্যার সারিয়ের জন্য :

```
...:           সারিতে কলামের জন্য :  
...:           মুদ্রণ (কলাম, শেষ = '  
...:           মুদ্রণ()  
...:  
১ ২ ৩  
৪ ৫ ৬
```

আপনি একটি বহুমাত্রিক অ্যারেকে onedimensional এর মতো করে পুনরাবৃত্তি করতে পারেন, এর flat অ্যাট্রিবিউট ব্যবহার করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[17] তে: integers.flat তে i এর জন্য :

```
...:           মুদ্রণ (i, শেষ = ' ')  
...:  
১ ২ ৩ ৪ ৫ ৬
```

৭.৪ নির্দিষ্ট মূল্য সহ ফিলিং অ্যারে

NumPy যথাক্রমে 0, 1 অথবা একটি নির্দিষ্ট মান ধারণকারী অ্যারে তৈরি করার জন্য **শূন্য**, এক এবং **পূর্ণ** ফাংশন প্রদান করে। ডিফল্টরূপে, শূন্য এবং এক float64 মান ধারণকারী অ্যারে তৈরি করে। আমরা দেখাবো কিভাবে এলিমেন্ট টাইপটি মুহূর্তের জন্য কাস্টমাইজ করতে হয়। এই ফাংশনগুলির প্রথম আঙ্গমেন্টটি অবশ্যই একটি পূর্ণসংখ্যা বা পূর্ণসংখ্যার একটি টুপল হতে হবে যা পছন্দসই মাত্রা নির্দিষ্ট করে। একটি পূর্ণসংখ্যার জন্য, প্রতিটি ফাংশন নির্দিষ্ট সংখ্যক উপাদান সহ একটি এক-মাত্রিক অ্যারে প্রদান করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: np হিসেবে numpy আমদানি করুন

[2] তে: np.zeros(5)
আউট[2]: অ্যারে([0., 0., 0., 0., 0.])

একগুচ্ছ পূর্ণসংখ্যার জন্য, এই ফাংশনগুলি নির্দিষ্ট মাত্রা সহ একটি বহুমাত্রিক অ্যারে প্রদান করে। আপনি শুন্য এবং এক ফাংশনের `dtype` কীওয়ার্ড আর্গুমেন্ট ব্যবহার করে অ্যারের উপাদানের ধরণ নির্দিষ্ট করতে পারেন:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[3] তে: np.ones((2, 4), dtype=int)
```

আউট[3]:

```
অ্যারে([[1, 1, 1, 1],
```

```
    [1, 1, 1, 1]])
```

পূর্ণ দ্বারা ফেরত দেওয়া অ্যারেতে দ্বিতীয় আর্গুমেন্টের মান সহ উপাদান রয়েছে এবং ধরণ:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[4] তে: np.full((3, 5), 13)
```

আউট[4]:

```
অ্যারে([[13, 13, 13, 13, 13], [13, 13, 13, 13, 13],
```

```
    [13, 13, 13, 13, 13]])
```

৭.৫ বিভিন্ন রেঞ্জ থেকে অ্যারে তৈরি করা

NumPy রেঞ্জ থেকে অ্যারে তৈরির জন্য অপিটমাইজড ফাংশন প্রদান করে। আমরা সহজ সমানভাবে ব্যবধানযুক্ত পূর্ণসংখ্যা এবং ফ্লোটিংপয়েন্ট রেঞ্জের উপর ফোকাস করি, তবে NumPy নন-লিনিয়ার রেঞ্জও সমর্থন করে।

১

১ <https://docs.scipy.org/doc/numpy/reference/routines.array-creation.html>.

arange দিয়ে পূর্ণসংখ্যার পরিসর তৈরি করা

NumPy এর `arange` ফাংশন ব্যবহার করে পূর্ণসংখ্যার রেঞ্জ তৈরি করা যাক —বিল্টইন ফাংশন রেঞ্জ ব্যবহারের অনুরূপ। প্রতিটি ক্ষেত্রে, `arange` প্রথমে ফলাফল প্রাপ্ত অ্যারের উপাদানের সংখ্যা নির্ধারণ করে, মেমোরি বরাদ্দ করে, তারপর নির্দিষ্ট পরিসরের মান সংরক্ষণ করে

অ্যারে:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

```
[1] তে: np হিসেবে numpy আমদানি করুন
```

```
[2] তে: np.arange(5)
```

```
আউট[2]: অ্যারে([0, 1, 2, 3, 4])
```

```
[3] তে: np.arange(5, 10)
```

```
আউট[3]: অ্যারে([5, 6, 7, 8, 9])
```

```
[4] তে: np.arange(10, 1, 2)
```

```
আউট[4]: অ্যারে([10, 8, 6, 4, 2])
```

যদিও আপনি আর্গুমেন্ট হিসেবে রেঞ্জ পাস করে অ্যারে তৈরি করতে পারেন, সর্বদা arange ব্যবহার করুন কারণ এটি অ্যারের জন্য অপ্টিমাইজ করা হয়েছে। শীঘ্ৰই আমরা দেখাবো কিভাবে বিভিন্ন অপারেশনের এক্সিকিউশন সময় নির্ধারণ করতে হয় যাতে আপনি তাদের কর্মক্ষমতা তুলনা করতে পারেন।

লিনস্পেস ব্যবহার করে ফ্লোটিং-পয়েন্ট রেঞ্জ তৈরি করা

NumPy এর `linspace` ফাংশন ব্যবহার করে আপনি সমানভাবে ব্যবধানযুক্ত ফ্লোটিংপয়েন্ট রেঞ্জ তৈরি করতে পারেন। ফাংশনের প্রথম দুটি আর্গুমেন্ট রেঞ্জের শুরু এবং শেষের মান নির্দিষ্ট করে এবং শেষের মানটি অ্যারেতে অন্তর্ভুক্ত করা হয়। এক্ষেত্রে কীওয়ার্ড আর্গুমেন্ট `num` সমানভাবে ব্যবধানযুক্ত মানের সংখ্যা নির্দিষ্ট করে - এই আর্গুমেন্টের ডিফল্ট মান হল 50:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[5] তে: np.linspace(0.0, 1.0, num=5)
```

```
আউট[5]: অ্যারে([ 0. , 0.25, 0.5 , 0.75, 1. ])
```

একটি অ্যারে পুনঃআকৃতিকরণ

আপনি বিভিন্ন উপাদান থেকে একটি অ্যারে তৈরি করতে পারেন, তারপর অ্যারে পদ্ধতি `reshape` ব্যবহার করে onedimensional অ্যারেটিকে multidimensional অ্যারেতে রূপান্তর করতে পারেন। আসুন 1 থেকে 20 পর্যন্ত মান ধারণকারী একটি অ্যারে তৈরি করি, তারপর এটিকে চারটি সারিতে পাঁচটি কলামে পুনরায় আকার দেই:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[6] তে: np.arange(1, 21).reshape(4, 5)
```

```
আউট[6]:
```

```
অ্যারে([[ 1, 2, 3, 4, 5],
           [ 6, 7, 8, 9, 10],
           [11, 12, 13, 14, 15],
           [16, 17, 18, 19, 20]])
```

পূর্ববর্তী স্লিপেটে চেইনড মেথড কলগুলি লক্ষ্য করুন। প্রথমে, arange একটি তৈরি করে

১-২০ মান সম্পর্কিত অ্যারে। তারপর আমরা ৪ পেতে সেই অ্যারেতে reshape কর করি
by5 অ্যারে প্রদর্শিত হয়েছিল।

আপনি যেকোনো অ্যারে পুনরায় আকার দিতে পারেন, তবে শর্ত থাকে যে নতুন আকৃতিতে একই সংখ্যক
মূল উপাদান হিসেবে। সুতরাং একটি ছয় উপাদানের একমাত্রিক অ্যারে 3by2 হতে পারে
অথবা 2by3 অ্যারে, এবং তদ্বিপরীত, কিন্তু একটি 15element অ্যারেকে একটিতে পুনরায় আকার দেওয়ার চেষ্টা করছে
4by4 অ্যারে (16 টি উপাদান) একটি ValueError সৃষ্টি করে।

বড় অ্যারে প্রদর্শন করা হচ্ছে

একটি অ্যারে প্রদর্শনের সময়, যদি ১০০০ বা তার বেশি আইটেম থাকে, তাহলে NumPy মাঝখানে ফেলে দেয়
আউটপুট থেকে সারি, কলাম অথবা উভয়ই। নিম্নলিখিত স্লিপেটগুলি ১০০,০০০ উৎপন্ন করে
উপাদান। প্রথম ক্ষেত্রে চারটি সারিই দেখানো হয়েছে কিন্তু শুধুমাত্র প্রথম এবং শেষ তিনটি
২৫,০০০ কলাম। স্বরলিপি ... অনুপস্থিত তথ্য প্রতিনিধিত্ব করে। দ্বিতীয় ক্ষেত্রে দেখা যাচ্ছে
১০০টি সারির প্রথম এবং শেষ তিনটি এবং ১০০০টি কলামের প্রথম এবং শেষ তিনটি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[7] তে: np.arange(1, 100001).reshape(4, 25000)

আউট[7]:

```
অ্যারে([[  ১,      ২,      ৩, ...,  ২৪৯৯৮,  ২৪৯৯৯,      ২৫০০০],
           [ ২৫০০১,  ২৫০০২,  ২৫০০৩, ...,  ৪৯৯৯৮,  ৪৯৯৯৯,      ৫০০০০],
           [ ৫০০০১,  ৫০০০২,  ৫০০০৩, ...,  ৭৪৯৯৮,  ৭৪৯৯৯,      ৭৫০০০],
           [ ৭৫০০১,  ৭৫০০২,  ৭৫০০৩, ...,  ৯৯৯৯৮,  ৯৯৯৯৯,  ১০০০০০]])
```

[8] তে: np.arange(1, 100001).reshape(100, 1000)

আউট[8]:

```
অ্যারে([[  ১,      ২,      ৩, ...,      ৯৯৮,      ৯৯৯,      ১০০০],
           [ ১০০১,      ১০০২,  ১০০৩, ...,      ১৯৯৮,      ১৯৯৯,      ২০০০],
           [ ২০০১,      ২০০২,  ২০০৩, ...,  ২৯৯৮,  ২৯৯৯,      ৩০০০],
           ...,
           [ ৯৭০০১,  ৯৭০০২,  ৯৭০০৩, ...,  ৯৭৯৯৮,  ৯৭৯৯৯,      ৯৮০০০],
           [ ৯৮০০১,  ৯৮০০২,  ৯৮০০৩, ...,  ৯৮৯৯৮,  ৯৮৯৯৯,      ৯৯০০০],
           [ ৯৯০০১,  ৯৯০০২,  ৯৯০০৩, ...,  ৯৯৯৯৮,  ৯৯৯৯৯,  ১০০০০০]])
```

৭.৬ লিস্ট বনাম অ্যারে পারফরম্যান্স: ভূমিকা %বার

বেশিরভাগ অ্যারে অপারেশন সংশ্লিষ্ট তালিকা অপারেশনের তুলনায় উল্লেখযোগ্যভাবে দ্রুত কার্যকর হয়।

প্রদর্শনের জন্য, আমরা IPython %timeit ম্যাজিক কমান্ড ব্যবহার করব, যা

অপারেশনের গড় সময়কাল। মনে রাখবেন যে আপনার সিস্টেমে প্রদর্শিত সময়গুলি পরিবর্তিত হতে পারে
আমরা এখানে যা দেখাচ্ছি তা থেকে।

৬০,০০,০০০ ডাই রোলের ফলাফল সম্বলিত একটি তালিকা তৈরির সময় নির্ধারণ

আমরা ষষ্ঠ পার্শ্বযুক্ত ডাই রোল করার পদ্ধতি ৬০,০০০,০০০ বার দেখিয়েছি। এখানে, আসুন র্যান্ডম মডিউলের র্যান্ডেজ ফাংশন ব্যবহার করে একটি তালিকা বোঝার মাধ্যমে ছয় মিলিয়ন ডাই রোলের একটি তালিকা তৈরি করি এবং %timeit ব্যবহার করে অপারেশনের সময় নির্ধারণ করি। মনে রাখবেন যে আমরা স্লিপেট [2] এর স্টেটমেন্টটিকে দুটি লাইনে বিভক্ত করার জন্য লাইন-কন্টিনিউয়েশন অক্ষর (\) ব্যবহার করেছি:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

[1] তে: র্যান্ডম আমদানি করুন

```
[2] তে: %timeit rolls_list = \
...:     [ range(0, 6_000_000) তে i এর জন্য random.randrange(1, 7) ]
প্রতি লুপে ৬.২৯ সেকেন্ড ± ১১৯ মিলিসেকেন্ড (গড় ± ৭ রানের স্ট্যান্ডার্ড ডেভেলপমেন্ট, প্রতিটিতে ১টি লুপ)
```

ডিফল্টরূপে, %timeit একটি লুপে একটি স্টেটমেন্ট এক্সিকিউট করে এবং এটি লুপটি সাতবার চালায়। যদি আপনি লুপের সংখ্যা নির্দেশ না করেন, %timeit একটি উপযুক্ত মান নির্বাচন করে। আমাদের পরীক্ষায়, গড়ে ৫০০ মিলিসেকেন্ডের বেশি সময় লেগেছে এমন অপারেশনগুলি কেবল একবার পুনরাবৃত্তি হয়েছিল এবং ৫০০ মিলিসেকেন্ডের কম সময় লেগেছে এমন অপারেশনগুলি ১০ বার বা তার বেশি পুনরাবৃত্তি হয়েছিল।

স্টেটমেন্টটি কার্যকর করার পর, %timeit স্টেটমেন্টের গড় এক্সিকিউশন সময়, সেইসাথে সমস্ত এক্সিকিউশনের স্ট্যান্ডার্ড ডেভিয়েশন প্রদর্শন করে। গড়ে, %timeit নির্দেশ করে যে ১১৯ মিলিসেকেন্ড (ms) এর স্ট্যান্ডার্ড ডেভিয়েশন সহ তালিকাটি তৈরি করতে ৬.২৯ সেকেন্ড সময় লেগেছে। মোট, পূর্ববর্তী স্লিপেটটি সাতবার স্লিপেটটি চালাতে প্রায় ৪৪ সেকেন্ড সময় নিয়েছে।

৬,০০০,০০০ ডাই এর ফলাফল ধারণকারী একটি অ্যারে তৈরির সময় নির্ধারণ রোলস

এবার, `numpy.random` মডিউল থেকে `randint` ফাংশন ব্যবহার করে ৬,০০০,০০০ ডাই রোলের একটি অ্যারে তৈরি করা যাক।

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

[3] তে: np হিসেবে numpy আমদানি করুন

```
[4] তে: %timeit rolls_array = np.random.randint(1, 7, 6_000_000) প্রতি লুপে 72.4 ms ± 635 μs (গড় ± ৭ রানের স্ট্যান্ডার্ড ডেভেলপমেন্ট, প্রতিটি 10টি লুপ)
```

গড়ে, %timeit নির্দেশ করে যে অ্যারে তৈরি করতে মাত্র ৭২.৪ মিলিসেকেন্ড সময় লেগেছে, যার স্ট্যান্ডার্ড ডেভিয়েশন ৬৩৫ মাইক্রোসেকেন্ড (μs)। মোট, পূর্ববর্তী স্লিপেটটি আমাদের কম্পিউটারে কার্যকর করতে মাত্র আধা সেকেন্ডেরও কম সময় লেগেছে—প্রায় ১/১০০ ভাগ সময়।

নিপেট [2] এক্সিকিউট করতে লেগেছে। অ্যারের সাহায্যে অপারেশনটি দুই ধাপ দ্রুততর!

৬০,০০০,০০০ এবং ৬০০,০০০,০০০ ডাই রোলস

এবার, ৬০,০০০,০০০ ডাই রোলের একটি অ্যারে তৈরি করা যাক:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[5] ক্ষেত্র: %timeit rolls_array = np.random.randint(1, 7, 60_000_000) প্রতি লুপে 873 ms ± 29.4 ms (গড় ± 7 রানের স্ট্যান্ডার্ড ডেভেলপমেন্ট, প্রতিটি  
লুপে 1 টি)
```

গড়ে, অ্যারেটি তৈরি করতে মাত্র ৮৭৩ মিলিসেকেন্ড সময় লেগেছে।

অবশ্যে, আসুন ৬০০,০০০,০০০ মিলিয়ন ডাই রোল করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[6] ক্ষেত্র: %timeit rolls_array = np.random.randint(1, 7, 600_000_000) প্রতি লুপে 10.1 সেকেন্ড ± 232 মিলিসেকেন্ড (গড় ± 7 রানের স্ট্যান্ডার্ড ডেভেলপমেন্ট,  
প্রতিটি লুপে 1 টি)
```

NumPy ব্যবহার করে ৬০০,০০০,০০০ উপাদান তৈরি করতে প্রায় ১০ সেকেন্ড সময় লেগেছে, যেখানে তালিকা বোঝার মাধ্যমে মাত্র ৬,০০০,০০০ উপাদান তৈরি করতে প্রায় ৬ সেকেন্ড সময় লেগেছে।

এই সময় সংক্রান্ত গবেষণার উপর ভিত্তি করে, আপনি স্পষ্টভাবে দেখতে পাবেন কেন গণনা-নিবিড় ক্রিয়াকলাপের জন্য তালিকার চেয়ে অ্যারেগুলিকে অগ্রাধিকার দেওয়া হয়। ডেটা সায়েন্স কেস স্টাডিতে, আমরা বিগ ডেটা এবং এআই-এর কর্মক্ষমতা-নিবিড় জগতে প্রবেশ করব। আমরা দেখব কিভাবে চতুর হার্ডওয়্যার, সফ্টওয়্যার, যোগাযোগ এবং অ্যালগরিদম ডিজাইন আজকের অ্যাপ্লিকেশনগুলির প্রায়শই বিশাল কম্পিউটিং চ্যালেঞ্জগুলি মোকাবেলা করার জন্য একত্রিত হয়।

%timeit পুনরাবৃত্তি কাস্টমাইজ করা হচ্ছে

প্রতিটি %timeit লুপের মধ্যে পুনরাবৃত্তির সংখ্যা এবং লুপের সংখ্যা n এবং r বিকল্পগুলির সাহায্যে কাস্টমাইজ করা যায়।
নিম্নলিখিতটি প্রতি লুপে তিনবার নিপেট [4] এর স্টেটমেন্ট কার্যকর করে এবং লুপটি দুবার চালায়:

২

^২ বেশিরভাগ পাঠকের জন্য, %timeits ডিফল্ট সেটিংস ব্যবহার করা ঠিক হবে।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[7] ক্ষেত্র: %timeit n3 r2 rolls_array = np.random.randint(1, 7, 6_000_000) প্রতি লুপে 85.5 ms ± 5.32 ms (গড় ± 2 রানের স্ট্যান্ডার্ড ডেভেলপমেন্ট, প্রতিটি 3টি লুপ) www.EBooksWorld.ir
```



অন্যান্য আইপিথন ম্যাজিক

আইপিথন বিভিন্ন কাজের জন্য ডজন ডজন জাদু সরবরাহ করে—সম্পূর্ণ তালিকার জন্য, আইপিথন ম্যাজিক্স ডকুমেন্টেশন দেখুন। এখানে কয়েকটি সহায়ক বিষয় রয়েছে:

৩

<http://ipython.readthedocs.io/en/stable/interactive/magics.html>.

- স্থানীয় ফাইল বা URL থেকে IPython-এ কোড পড়ার জন্য `%load`।
- `%save` একটি ফাইলে স্লিপেট সংরক্ষণ করতে।
- IPython থেকে একটি .py ফাইল চালানোর জন্য `%run` ব্যবহার করুন।
- IPython আউটপুটগুলির জন্য ডিফল্ট ফ্রোটিংপয়েন্ট নির্ভুলতা পরিবর্তন করতে `%precision` ব্যবহার করুন।
- `%cd` ব্যবহার করে ডিরেক্টরি পরিবর্তন করা যাবে, প্রথমে IPython থেকে বেরিয়ে না এসেই।
- `%edit` ব্যবহার করে একটি বহিরাগত সম্পাদক চালু করুন—যদি আপনার আরও জটিল স্লিপেট পরিবর্তন করার প্রয়োজন হয় তবে এটি কার্যকর।
- বর্তমান IPython সেশনে আপনার দ্বারা সম্পাদিত সমস্ত স্লিপেট এবং কমান্ডের তালিকা দেখতে `%history` টিপুন।

৭.৭ অ্যারে অপারেটর

NumPy অনেক অপারেটর প্রদান করে যা আপনাকে সহজ এক্সপ্রেশন লিখতে সক্ষম করে যা সম্পূর্ণ অ্যারেতে ক্রিয়াকলাপ সম্পাদন করে। এখানে, আমরা অ্যারে এবং সংখ্যাসূচক মানের মধ্যে এবং একই আকারের অ্যারের মধ্যে পার্টিগণিত প্রদর্শন করি।

অ্যারে এবং স্বতন্ত্র সংখ্যাসূচক মান সহ গাণিতিক ক্রিয়াকলাপ

প্রথমে, আসুন আমরা অ্যারে এবং সংখ্যাসূচক মান সহ অ্যারে এবং গাণিতিক অপারেটর এবং অগমেন্টেড অ্যাসাইনমেন্ট ব্যবহার করে উপাদানভিত্তিক গাণিতিক সম্পাদন করি। প্রতিটি উপাদানে এলিমেন্টভিত্তিক ক্রিয়াকলাপ প্রয়োগ করা হয়, তাই স্লিপেট [4] প্রতিটি উপাদানকে 2 দিয়ে গুণ করে এবং স্লিপেট [5] প্রতিটি উপাদানকে কিউব করে। প্রতিটি ফলাফল ধারণকারী একটি নতুন অ্যারে প্রদান করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[2] তে: সংখ্যা = np.arange(1, 6)

[3] তে: সংখ্যা

আউট[3]: অ্যারে([1, 2, 3, 4, 5])

[4] তে: সংখ্যা * 2

আউট[4]: অ্যারে([2, 4,

৬, ৮, ১০])

[5] তে: সংখ্যা ** 3

আউট[5]: অ্যারে([1,

৮, ২৭, ৬৪, ১২৫])

[6]: সংখ্যা # সংখ্যাগুলি গাণিতিক অপারেটরদের দ্বারা অপরিবর্তিত রয়েছে

আউট[6]: অ্যারে([1, 2, 3, 4, 5])

স্লিপেট [6] দেখায় যে গাণিতিক অপারেটররা সংখ্যা পরিবর্তন করেনি। অপারেটর +

এবং * পরিবর্তনীয়, তাই স্লিপেট [4] কে $2 * \text{সংখ্যা}$ হিসেবেও লেখা যেতে পারে।

বর্ধিত অ্যাসাইনমেন্টগুলি বাম অপারেন্ডের প্রতিটি উপাদানকে পরিবর্তন করে।

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[7] তে: সংখ্যা += 10

[8] তে: সংখ্যা

আউট[8]: অ্যারে([11, 12, 13, 14, 15])

সম্প্রচার

সাধারণত, গাণিতিক ক্রিয়াকলাপের জন্য একই আকার এবং আকৃতির দুটি অ্যারে অপারেন্ডের প্রয়োজন হয়। যখন একটি অপারেন্ড একটি একক মান হয়, যাকে [স্কেলার বলা হয়](#), তখন NumPy উপাদানভিত্তিক গণনা সম্পাদন করে যেন স্কেলারটি অন্য অপারেন্ডের মতো একই আকৃতির একটি অ্যারে, কিন্তু এর সমস্ত উপাদানে স্কেলার মান সহ। একে [ব্রডকাস্টিং](#) বলা হয়।

স্লিপেট [4], [5] এবং [7] প্রতিটি এই ক্ষমতা ব্যবহার করে। উদাহরণস্বরূপ, স্লিপেট [4] এর সমতুল্য:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

সংখ্যা * [২, ২, ২, ২, ২]

বিভিন্ন আকার এবং আকৃতির অ্যারের মধ্যেও ব্রডকাস্টিং প্রয়োগ করা যেতে পারে, যা কিছু সংক্ষিপ্ত এবং শক্তিশালী ম্যানিপুলেশন সক্ষম করে। আমরা অধ্যায়ের পরে NumPy এর সার্বজনীন ফাংশনগুলি পরিচয় করিয়ে দেওয়ার সময় ব্রডকাস্টিংয়ের আরও উদাহরণ দেখাব।

অ্যারের মধ্যে পাটিগণিতের ক্রিয়াকলাপ

আপনি একই আকৃতির অ্যারের মধ্যে গাণিতিক ক্রিয়াকলাপ এবং বর্ধিত অ্যাসাইনমেন্ট সম্পাদন করতে পারেন।

আসুন একমাত্রিক অ্যারের সংখ্যাগুলিকে গুণ করি এবং

সংখ্যা2 (নীচে তৈরি করা হয়েছে) যার প্রতিটিতে পাঁচটি করে উপাদান রয়েছে:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

[9] তে: numbers2 = np.linspace(1.1, **5.5**, 5)

[10] তে: সংখ্যা 2

আউট[10]: অ্যারে([1.1, 2.2, 3.3, 4.4, 5.5])

[11] তে: সংখ্যা * সংখ্যা2

আউট[11]: অ্যারে([12.1, 26.4, 42.9, 61.6, 82.5])

ফলাফল হল প্রতিটি অপারেন্টে অ্যারেগুলিকে উপাদান অনুসারে গুণ করে একটি নতুন অ্যারে তৈরি করা হয়—১.১ * ১.১,

১.২ * ২.২, ১.৩ এবং ফ্লোটিংপয়েন্ট সংখ্যার ফলে

* ৩.৩, ইত্যাদি। পূর্ণসংখ্যার অ্যারের মধ্যে পাটিগণিত

ফ্লোটিংপয়েন্ট সংখ্যার একটি অ্যারে তৈরি হয়।

অ্যারের তুলনা করা

আপনি পৃথক মান এবং অন্যান্য অ্যারের সাথে অ্যারে তুলনা করতে পারেন। তুলনাগুলি উপাদান অনুসারে করা হয়। এই ধরনের তুলনা বুলিয়ান মানের অ্যারে তৈরি করে যেখানে প্রতিটি উপাদানের সত্য বা মিথ্যা মান তুলনা ফলাফল নির্দেশ করে:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

[12] তে: সংখ্যা

আউট[12]: অ্যারে([11, 12, 13, 14, 15])

[13] তে: সংখ্যা >= **13**

আউট[13]: অ্যারে([মিথ্যা, মিথ্যা, সত্য, সত্য, সত্য])

[14] তে: সংখ্যা 2

আউট[14]: অ্যারে([1.1, 2.2, 3.3, 4.4, 5.5])

[15] তে: সংখ্যা2 < সংখ্যা

আউট[15]: অ্যারে([সত্য, সত্য, সত্য, সত্য, সত্য])

[16] তে: সংখ্যা == সংখ্যা2

আউট[16]: অ্যারে([মিথ্যা, মিথ্যা, মিথ্যা, মিথ্যা, মিথ্যা])

[17] তে: সংখ্যা == সংখ্যা

আউট[17]: অ্যারে([সত্য, সত্য, সত্য, সত্য, সত্য])

স্লিপেট [13] ব্রডকাস্টিং ব্যবহার করে নির্ধারণ করে যে সংখ্যার প্রতিটি উপাদান 13 এর চেয়ে বড় নাকি সমান। বাকি স্লিপেটগুলি প্রতিটি অ্যারে অপারেন্ডের সংশ্লিষ্ট উপাদানগুলির তুলনা করে।

৭.৮ অসম্পূর্ণ গণনা পদ্ধতি

একটি অ্যারের বিভিন্ন পদ্ধতি রয়েছে যা এর বিষয়বস্তু ব্যবহার করে গণনা সম্পাদন করে। ডিফল্টরূপে, এই পদ্ধতিগুলি অ্যারের আকৃতি উপেক্ষা করে এবং গণনার সমস্ত উপাদান ব্যবহার করে।

উদাহরণস্বরূপ, একটি অ্যারের গড় গণনা করলে তার আকৃতি নির্বিশেষে তার সমস্ত উপাদানের সমষ্টি হয়, তারপর উপাদানের মোট সংখ্যা দিয়ে ভাগ করা হয়। আপনি প্রতিটি মাত্রার উপরও এই গণনাগুলি সম্পাদন করতে পারেন। উদাহরণস্বরূপ, একটি দ্বিমাত্রিক অ্যারেতে, আপনি প্রতিটি সারির গড় এবং প্রতিটি কলামের গড় গণনা করতে পারেন।

তিনটি পরীক্ষায় চারজন শিক্ষার্থীর গ্রেড উপস্থাপনকারী একটি অ্যারে বিবেচনা করুন:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

[1] তে: np হিসেবে numpy আমদানি করুন

[2] তে: grades = np.array([[87, 96, 70], [100, 87, 90],
...: [98, 99, 90], [100, 88, 82]]))
...:

[3]: গ্রেড
আউট[3]:
অ্যারে([[87, 96, 70], [100, 87, 90],
.....
[98, 99, 90],
[100, 88, 82]]))

আমরা **যোগফল**, **সর্বনিম্ন**, **সর্বোচ্চ**, **গড়**, **স্ট্যান্ডার্ড** (স্ট্যান্ডার্ড ডেভিয়েশন) গণনা করার জন্য পদ্ধতি ব্যবহার করতে পারি এবং **var** (ভেরিয়েল) —প্রতিটি একটি কার্যকরী স্টাইল প্রোগ্রামিং হ্রাস:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

[4] তে: grades.sum()
আউট[4]: ১০৫৪

[5] তে: grades.min()
আউট[5]: 70

[6] তে: grades.max()
আউট[6]: 100

[7] তে: grades.mean()

[8] তে: grades.std()

আউট[8]: 8.792357792739987

[9] তে: grades.var()

আউট[9]: 77.305555555555556

সারি বা কলাম অনুসারে গণনা

অনেক গণনা পদ্ধতি নির্দিষ্ট অ্যারের মাত্রার উপর সংশ্লিষ্ট হতে পারে, যা অ্যারের অক্ষ নামে পরিচিত। এই পদ্ধতিগুলিতে একটি অক্ষ কীওয়ার্ড আর্গুমেন্ট থাকে যা গণনায় কোন মাত্রা ব্যবহার করতে হবে তা নির্দিষ্ট করে, যা আপনাকে দ্বিমাত্রিক অ্যারেতে সারি বা কলাম অনুসারে গণনা করার দ্রুত উপায় দেয়।

ধরে নিন যে আপনি প্রতিটি পরীক্ষার গড় গ্রেড গণনা করতে চান, যা গ্রেডের কলাম দ্বারা প্রতিনিধিত্ব করা হয়। axis=0 উল্লেখ করলে সমস্ত সারির মানের গণনা করা হয়।

প্রতিটি কলামের মধ্যে:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

[10] তে: grades.mean(axis=0)

আউট[10]: অ্যারে([95.25, 85.25, 83.])

তাহলে উপরের ৯৫.২৫ হল প্রথম কলামের গ্রেডের গড় (৮৭, ১০০, ৯৪ এবং ১০০), ৮৫.২৫ হল দ্বিতীয় কলামের গ্রেডের গড় (৯৬, ৮৭, ৭৭ এবং ৮১) এবং ৮৩ হল তৃতীয় কলামের গ্রেডের গড় (৭০, ৯০, ৯০ এবং ৮২)। আবার, NumPy '৮৩.'-এ দশমিক বিন্দুর ডানদিকে ০s প্রদর্শন করে না। আরও মনে রাখবেন যে এটি একই ক্ষেত্রের প্রস্তুত সমস্ত উপাদানের মান প্রদর্শন করে, যার কারণে '৮৩.' এর পরে

দুটি স্থান।

একইভাবে, axis=1 উল্লেখ করলে প্রতিটি সারির মধ্যে সমস্ত কলামের মানের গণনা করা হয়। সমস্ত পরীক্ষার জন্য প্রতিটি শিক্ষার্থীর গড় গণনা করতে, আমরা পারি

ব্যবহার:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

[11] তে: grades.mean(axis=1)

আউট[11]: অ্যারে([84.33333333, 92.33333333, 87.

, ৮৭.৬৬৬৬৬৬৬৭])

এটি চারটি গড় তৈরি করে—প্রতিটি সারির মানগুলির জন্য একটি করে। সুতরাং 84.33333333 হল

NumPy অ্যারেতে আরও অনেক গণনা পদ্ধতি রয়েছে। সম্পূর্ণ তালিকার জন্য, দেখুন

<https://docs.scipy.org/doc/numpy/reference/arrays.ndarray.html>

৭.৯ সর্বজনীন ফাংশন

NumPy ডজন ডজন স্বতন্ত্র **ইউনিভার্সাল ফাংশন** (বা ufuncs) অফার করে যা বিভিন্ন উপাদানভিত্তিক ক্রিয়াকলাপ সম্পাদন করে। প্রতিটি এক বা দুটি অ্যারে বা অ্যারের মতো (যেমন তালিকা) আর্গুমেন্ট ব্যবহার করে তার কাজ সম্পাদন করে। এই ফাংশনগুলির মধ্যে কিছু অ্যারেতে + এবং * এর মতো অপারেটর ব্যবহার করার সময় ডাকা হয়। প্রতিটি ফলাফল ধারণকারী একটি নতুন অ্যারে প্রদান করে।

আসুন একটি অ্যারে তৈরি করি এবং **sqrt** ব্যবহার করে এর মানের বর্গমূল গণনা করি।

সর্বজনীন ফাংশন:

কোড ইমেজ দেখতে এখানে [ক্লিক করুন](#)

[1] তে: np হিসেবে numpy আমদানি করুন

[2] তে: সংখ্যা = np.array([1, 4, 9, **16, 25**, 36])

[3] তে: np.sqrt(সংখ্যা)

আউট[3]: অ্যারে([1., 2., 3., 4., 5., 6.])

add universal ফাংশন ব্যবহার করে একই আকৃতির দুটি অ্যারে যোগ করা যাক :

কোড ইমেজ দেখতে এখানে [ক্লিক করুন](#)

[4] তে: numbers2 = np.arange(1, 7) * **10**

[5] তে: সংখ্যা 2

আউট[5]: অ্যারে([10, 20, 30, 40, 50, 60])

[6] তে: np.add(সংখ্যা, সংখ্যা2)

আউট[6]: অ্যারে([11, 24, 39, 56, 75, 96])

np.add(numbers, numbers2) রাশিটি এর সমতুল্য:

সংখ্যা + সংখ্যা2

ইউনিভার্সাল ফাংশন সহ সম্প্রচার

চলুন, multiply universal ফাংশন ব্যবহার করে number2 এর প্রতিটি উপাদানকে ক্ষেত্রের মান 5 দিয়ে গুণ করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[7] তে: np.multiply(numbers2, 5)
আউট[7]: অ্যারে([ 50, 100, 150, 200, 250, 300])
```

np.multiply(numbers2, 5) রাশিটি এর সমতুল্য:

সংখ্যা ২ * ৫

আসুন সংখ্যা 2 কে 2by3 অ্যারেতে পুনঃআকার দেই, তারপর এর মানগুলিকে তিনটি উপাদানের এক-মাত্রিক
অ্যারে দিয়ে গুণ করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[8] তে: numbers3 = numbers2.reshape(2, 3)
[9] তে: সংখ্যা 3
আউট[9]:
অ্যারে([[ 10, 20, 30],
           [ 80, 60, 40]])
```

```
[10] তে: numbers4 = np.array([2, 4, 6])
```

```
[11] তে: np.multiply(numbers3, numbers4)
আউট[11]:
অ্যারে([[ 20, 80, 180], [ 80, 200, 360]])
```

এটি কাজ করে কারণ numbers4 এর দৈর্ঘ্য number3 এর প্রতিটি সারির সমান, তাই
NumPy number4 কে নিম্নলিখিত অ্যারে হিসেবে বিবেচনা করে গুণন প্রক্রিয়া প্রয়োগ করতে পারে:

অ্যারে([[2, 8, 6], [2, 8, 6]])

যদি একটি সার্বজনীন ফাংশন দুটি ভিন্ন আকৃতির অ্যারে গ্রহণ করে যা সম্প্রচার সমর্থন করে না, তাহলে একটি
ValueError ঘটে। আপনি সম্প্রচারের নিয়মগুলি এখানে দেখতে পারেন:

অন্যান্য সার্বজনীন ফাংশন

NumPy ডকুমেন্টেশনে সর্বজনীন ফাংশনগুলিকে পাঁচটি বিভাগে তালিকাভুক্ত করা হয়েছে—গণিত, ত্রিকোণমিতি, বিট ম্যানিপুলেশন, তুলনা এবং ভাসমান বিল্ড। নিম্নলিখিত টেবিলে প্রতিটি বিভাগের কিছু ফাংশন তালিকাভুক্ত করা হয়েছে। আপনি সম্পূর্ণ তালিকা, তাদের বিবরণ এবং সর্বজনীন ফাংশন সম্পর্কে আরও তথ্য এখানে দেখতে পারেন:

NumPy সার্বজনীন ফাংশন

গণিত—যোগ, বিয়োগ, গুণ, ভাগ, তাগশেষ, এক্সপ, লগ, বর্গমূল, ঘাত এবং আরও অনেক কিছু!

ত্রিকোণমিতি- সিন, কস, ট্যান, হাইপোট, আর্কসিন, আর্কোস, আর্কটান এবং
আরও।

বিট ম্যানিপুলেশন—বিটওয়াইজ_এবং, বিটওয়াইজ_অর, বিটওয়াইজ_এক্সওর, ইনভার্ট, বাম_শিফট এবং ডান_শিফট।

তুলনা—বৃহত্তর, বৃহত্তর_সমান, কম, কম_সমান, সমান, সমান নয়, যৌক্তিক_এবং, যৌক্তিক_অথবা, যৌক্তিক_এক্সর,
যৌক্তিক_নট, সর্বনিম্ন, সর্বোচ্চ, এবং আরও অনেক কিছু।

ভাসমান বিল্ড—তল, সিল, আইএসআইএনএফ, ইসনান, ফ্যাবস, ট্রাঙ্ক এবং আরও অনেক কিছু।

৭.১০ সূচক এবং স্লাইসিং

"সিকোয়েল্জ: তালিকা এবং টুপলস" অধ্যায়ে আমরা যে একই সিনট্যাক্স এবং কৌশলগুলি দেখিয়েছি তা ব্যবহার করে এক-মাত্রিক
অ্যারেগুলিকে সূচীবদ্ধ এবং স্লাইস করা যেতে পারে। এখানে, আমরা অ্যারে-নির্দিষ্ট সূচীবদ্ধকরণ এবং স্লাইসিং ক্ষমতার উপর ফোকাস করব।

দ্বি-মাত্রিক অ্যারে দিয়ে ইনডেক্সিং

দ্বিমাত্রিক অ্যারেতে একটি উপাদান নির্বাচন করতে, বর্গাকার বন্ধনীতে উপাদানের সারি এবং কলাম সূচক ধারণকারী একটি টিপল নির্দিষ্ট করুন (যেমন স্লিপেট [4] তে):

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: np হিসেবে numpy আমদানি করুন

[2] তে: grades = np.array([[87, 96, 70], [100, 87, 90],
...: [98, 99, 90], [100, 88, 82]]])
...:

[3]: গ্রেড

আউট[3]:

অ্যারে([[87, 96, 70],
[100, 87, 90],
[98, 99, 90],
[100, 88, 82]])

[4] তে: গ্রেড[0, 1] # সারি 0, কলাম 1
আউট[4]: 96

দ্বি-মাত্রিক অ্যারের সারির একটি উপসেট নির্বাচন করা

একটি একক সারি নির্বাচন করতে, বর্গাকার বন্ধনীতে শুধুমাত্র একটি সূচক উল্লেখ করুন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[5]: গ্রেড[1]
আউট[5]: অ্যারে([100, 87, 90])

একাধিক ক্রমিক সারি নির্বাচন করতে, স্লাইস নোটেশন ব্যবহার করুন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6]: গ্রেড[0:2]
আউট[6]:
অ্যারে([[87, 96, 70],
[100, 87, 90]])

একাধিক অ-ক্রমিক সারি নির্বাচন করতে, সারি সূচকের একটি তালিকা ব্যবহার করুন:

[7]: গ্রেড[[1, 3]]
আউট[7]:
অ্যারে([[100, 87, 90], [100, 88, 82]])

দ্বি-মাত্রিক অ্যারের কলামের একটি উপসেট নির্বাচন করা

আপনি কলামের উপসেটগুলি নির্বাচন করতে পারেন একটি টিপল প্রদান করে যেখানে সারি(গুলি) এবং কলাম(গুলি) নির্দিষ্ট করা হবে। প্রতিটি একটি নির্দিষ্ট সূচক, একটি স্লাইস অথবা একটি তালিকা হতে পারে। আসুন শুধুমাত্র প্রথম কলামের উপাদানগুলি নির্বাচন করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[8]: গ্রেড[:, 0]

আউট[8]: অ্যারে([87, 100, 94, 100])

কমার পরে ০ ইঙ্গিত করে যে আমরা কেবল ০ কলাম নির্বাচন করছি। কমার আগে : নির্দেশ করে যে কলামের মধ্যে কোন সারি নির্বাচন করতে হবে। এই ক্ষেত্রে, : হল একটি স্লাইস যা সমস্ত সারিকে প্রতিনিধিত্ব করে। এটি একটি নির্দিষ্ট সারি সংখ্যা, সারির একটি উপসেট প্রতিনিধিত্বকারী একটি স্লাইস বা নির্বাচন করার জন্য নির্দিষ্ট সারি সূচকের একটি তালিকাও হতে পারে, যেমন স্লিপেট [5]-[7]।

আপনি একটি স্লাইস ব্যবহার করে পরপর কলাম নির্বাচন করতে পারেন:

[9]: গ্রেড[:, 1:3]

আউট[9]:

অ্যারে([[96, 70], [87, 90],

[৭৭, ৯০],

[81, 82]))

অথবা কলাম সূচকের তালিকা ব্যবহার করে নির্দিষ্ট কলাম:

[10]: গ্রেড[:, [0, 2]]

আউট[10]:

অ্যারে([[87, 70], [100, 90],

[94, 90],

[১০০, ৮২]))

৭.১১ ভিউ: অগভীর কপি

পূর্ববর্তী অধ্যায়ে ভিউ অবজেক্টের সাথে পরিচয় করিয়ে দেওয়া হয়েছিল - অর্থাৎ, এমন অবজেক্ট যা অন্যান্য অবজেক্টের ডেটা "দেখে", তাদের নিজস্ব ডেটার কপি না থাকে। ভিউ হল অগভীর কপি।

বিভিন্ন অ্যারে পদ্ধতি এবং স্লাইসিং অপারেশন একটি অ্যারের ডেটার ভিউ তৈরি করে।

অ্যারে মেথড ভিউ মূল অ্যারের ভিউ সহ একটি নতুন অ্যারে অবজেক্ট ফেরত দেয়।

বস্তুর তথ্য। প্রথমে, একটি অ্যারে এবং সেই অ্যারের একটি ভিউ তৈরি করা যাক:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

[1] তে: np হিসেবে numpy আমদানি করুন

[2] তে: সংখ্যা = np.arange(1, 6)

[3] তে: সংখ্যা

আউট[3]: অ্যারে([1, 2, 3, 4, 5])

[4] তে: numbers2 = numbers.view()

[5] তে: সংখ্যা 2

আউট[5]: অ্যারে([1, 2, 3, 4, 5])

আমরা বিল্টইন id ফাংশন ব্যবহার করে দেখতে পারি যে সংখ্যা এবং numbers2 ডি঱ বস্তু:

[6] তে: আইডি(সংখ্যা)

আউট[6]: 4462958592

[7] তে: id(numbers2)

আউট[7]: 4590846240

numbers2 সংখ্যার মতো একই ডেটা দেখায় তা প্রমাণ করার জন্য, আসুন সংখ্যার একটি উপাদান পরিবর্তন করি, তারপর উভয় অ্যারে প্রদর্শন করি:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

[8] তে: সংখ্যা [1] *= 10

[9] তে: সংখ্যা 2

আউট[9]: অ্যারে([1, 20, 3,

8, 4])

[10] তে: সংখ্যা

আউট[10]: অ্যারে([1, 20, 3, 4, 5])

একইভাবে, ভিউতে একটি মান পরিবর্তন করলে মূল অ্যারের মানও পরিবর্তিত হয়:

[11] তে: সংখ্যা2[1] /= 10

[12] তে: সংখ্যা

আউট[12]: অ্যারে([1, 2, 3, 4, 5])

[13] তে: সংখ্যা 2

আউট[13]: অ্যারে([1, 2, 3, 4, 5])

স্লাইস ভিউ

স্লাইসগুলিও ভিউ তৈরি করে। আসুন number2 কে এমন একটি স্লাইস বানাই যেখানে কেবল প্রথম তিনটি দেখা যায়।

সংখ্যার উপাদান:

[14] তে: সংখ্যা2 = সংখ্যা[0:3]

[15] তে: সংখ্যা 2

আউট[15]: অ্যারে([1, 2, 3])

আবার, আমরা নিশ্চিত করতে পারি যে সংখ্যা এবং numbers2 id সহ ভিন্ন বস্তু:

[16] তে: id(numbers)

আউট[16]: 4462958592

[17] তে: id(numbers2)

আউট[17]: 4590848000

আমরা নিশ্চিত করতে পারি যে numbers2 হল শুধুমাত্র প্রথম তিনটি সংখ্যার উপাদানের একটি দৃশ্য, numbers2[3] অ্যাক্সেস করার চেষ্টা করে, যা একটি IndexError তৈরি করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[18] তে: সংখ্যা2[3]

সূচক অটি

ট্রেসব্যাক (সর্বশেষ কলটি শেষ

<module>() এ ipythoninput18582053f52daa>

> ১টি সংখ্যা২[৩]

IndexError: সূচী 3 অক্ষ 0 এর জন্য সীমার বাইরে, যার আকার 3

এখন, আসুন উভয় অ্যারে ভাগ করে নেওয়া একটি উপাদান পরিবর্তন করি, তারপর সেগুলি প্রদর্শন করি। আবার, আমরা দেখতে পাই যে numbers2 হল সংখ্যার একটি দৃশ্য:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[19] তে: সংখ্যা [1] *= 20

[20] তে: সংখ্যা

আউট[20]: অ্যারে([1, 2, 3, 4, 5])

[21] তে: সংখ্যা 2

আউট[21]: অ্যারে([1, 40, 3])

৭.১২ গভীর কপি

যদিও ডিউগুলি পৃথক অ্যারে অবজেক্ট, তারা অন্যান্য অ্যারে থেকে উপাদান ডেটা ভাগ করে মেমরি সংরক্ষণ করে। তবে, পরিবর্তনযোগ্য মানগুলি ভাগ করার সময়, কখনও কখনও মূল ডেটার স্বাধীন কপি দিয়ে একটি **গভীর অনুলিপি** তৈরি করা প্রয়োজন। এটি মাল্টিকোর প্রোগ্রামিংয়ে বিশেষভাবে গুরুত্বপূর্ণ, যেখানে আপনার প্রোগ্রামের পৃথক অংশ একই সময়ে আপনার ডেটা পরিবর্তন করার চেষ্টা করতে পারে, সম্ভবত এটি দুর্ঘিত করতে পারে।

অ্যারে **মেথড কপি** মূল অ্যারে অবজেক্টের ডেটার একটি ডিপ কপি সহ একটি নতুন অ্যারে অবজেক্ট ফেরত দেয়। প্রথমে, আসুন একটি অ্যারে এবং সেই অ্যারের একটি ডিপ কপি তৈরি করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: np হিসেবে numpy আমদানি করুন

[2] তে: সংখ্যা = np.arange(1, 6)

[3] তে: সংখ্যা

আউট[3]: অ্যারে([1, 2, 3, 4, 5])

[4] তে: numbers2 = numbers.copy()

[5] তে: সংখ্যা 2

আউট[5]: অ্যারে([1, 2, 3, 4, 5])

সংখ্যার তথ্যের একটি পৃথক কপি numbers2-এর কাছে আছে তা প্রমাণ করার জন্য, আসুন সংখ্যার একটি উপাদান পরিবর্তন করি, তারপর উভয় অ্যারে প্রদর্শন করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6] তে: সংখ্যা [1] *= 10

[7] তে: সংখ্যা

আউট[7]: অ্যারে([1, 20, 3,

8, 5])

[8] তে: সংখ্যা 2

আউট[8]: অ্যারে([1, 2, 3, 4, 5])

আপনি দেখতে পাচ্ছেন, পরিবর্তনটি কেবল সংখ্যায় দেখা যাচ্ছে।

পূর্ববর্তী অধ্যায়গুলিতে, আমরা অগভীর কপি করার পদ্ধতিটি আলোচনা করেছি। এই অধ্যায়ে, আমরা অ্যারে অবজেক্টগুলিকে তাদের কপি পদ্ধতি ব্যবহার করে কীভাবে ডিপ কপি করতে হয় তা আলোচনা করেছি। যদি আপনার অন্যান্য ধরণের পাইথন অবজেক্টের ডিপ কপির প্রয়োজন হয়, তাহলে সেগুলি কপি মডিউলের **ডিপকপি** ফাংশনে পাঠান।

৭.১৩ পুনর্নির্মাণ এবং স্থানান্তর

আমরা এক-মাত্রিক রেঞ্জ থেকে দ্বিমাত্রিক অ্যারে তৈরি করতে অ্যারে পদ্ধতি পুনর্নির্মাণ ব্যবহার করেছি। NumPy অ্যারে পুনর্নির্মাণের বিভিন্ন উপায় প্রদান করে।

পুনঃআকৃতি বনাম পুনঃআকার

অ্যারে মেথড রিশেপ এবং রিসাইজ উভয়ই আপনাকে অ্যারের ডাইমেনশন পরিবর্তন করতে সক্ষম করে। মেথড রিশেপ নতুন ডাইমেনশন সহ মূল অ্যারের একটি ডিউ (অগভীর কপি) প্রদান করে। এটি মূল অ্যারে পরিবর্তন করে না:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[1] তে: np হিসেবে numpy আমদানি করুন

[2] তে: গ্রেড = np.array([[87, 96, 70], [100, 87, 90]])

[3]: গ্রেড

আউট[3]:

অ্যারে([[87, 96, 70],
[100, 87, 90]])

[4] তে: grades.reshape(1, 6)

আউট[4]: অ্যারে([[87, 96, 70, 100, 87, 90]])

[5] তে: গ্রেড

আউট[5]:

অ্যারে([[87, 96, 70],
[100, 87, 90]])

মেথড **রিসাইজ** মূল অ্যারের আকৃতি পরিবর্তন করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6] তে: grades.resize(1, 6)

[7] তে: গ্রেড

আউট[7]: অ্যারে([[87, 96, 70, 100, 87, 90]])

আপনি একটি বহুমাত্রিক অ্যারে নিতে পারেন এবং **flatten** and **ravel** পদ্ধতি ব্যবহার করে এটিকে একটি একক মাত্রায় পরিণত করতে পারেন। Method flatten deep মূল অ্যারের ডেটা কপি করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[8] তে: গ্রেড = np.array([[87, 96, 70], [100, 87, 90]])

[9] তে: গ্রেড

আউট[9]:

অ্যারে([[87, 96, 70], [100, 87, 90]])

[10] তে: flattened = grades.flatten()

[11] তে: সমতল

আউট[11]: অ্যারে([87, 96, 70, 100, 87, 90])

[12]: গ্রেড

আউট[12]:

অ্যারে([[87, 96, 70],

[100, 87, 90]])

grades এবং flattened ডেটা শেয়ার করে না তা নিশ্চিত করতে, flattened এর একটি উপাদান পরিবর্তন করা যাক, তারপর উভয় অ্যারে প্রদর্শন করা যাক:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[13] তে: সমতল[0] = 100

[14] তে: সমতল

আউট[14]: অ্যারে([100, 96, 70, 100, 87, 90])

[15]: গ্রেড

আউট[15]:

অ্যারে([[87, 96, 70],

[100, 87, 90]])

মেথড রেভেল মূল অ্যারের একটি ভিউ তৈরি করে, যা গ্রেড অ্যারের ডেটা শেয়ার করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[16] তে: raveled = grades.ravel()

[17] তে: র্যাডেলড

আউট[17]: অ্যারে([87, 96, 70, 100, 87, 90])

[18]: গ্রেড

আউট[18]:

অ্যারে([[87, 96, 70], [100, 87, 90]])

grades এবং raveled একটি ডেটা শেয়ার করে তা নিশ্চিত করতে, আসুন raveled এর একটি উপাদান পরিবর্তন করি, তারপর উভয় অ্যারে প্রদর্শন করি:

কোড ইমেজ দেখতে এখানে লিঙ্ক করুন

[19] তে: র্যাডেলড[0] = 100

[20] তে: র্যাডেলড

আউট[20]: অ্যারে([100, 96, 70, 100, 87, 90])

[21]: গ্রেড

আউট[21]:

অ্যারে([[100, 96, 70],

[100, 87, 90]])

সারি এবং কলাম স্থানান্তর করা

আপনি দ্রুত একটি অ্যারের সারি এবং কলাম স্থানান্তর করতে পারেন —অর্থাৎ অ্যারেটিকে "ফ্লিপ" করতে পারেন, তাই সারিগুলি কলাম হয়ে যায় এবং কলামগুলি সারি হয়ে যায়। **T বৈশিষ্ট্য**

অ্যারের একটি ট্রান্সপোজড ভিউ (অগভীর অনুলিপি) প্রদান করে। মূল গ্রেড অ্যারে তিনটি পরীক্ষার (কলাম) দুটি শিক্ষার্থীর গ্রেড (সারি) উপস্থাপন করে। আসুন তিনটি পরীক্ষার (কলাম) গ্রেড হিসাবে ডেটা দেখতে সারি এবং কলামগুলি ট্রান্সপোজ করি (

সারি) দুইজন শিক্ষার্থীর জন্য (কলাম):

[22] তে: grades.T

আউট[22]:

অ্যারে([[100, 100], [96, 87],

[90, 90]])

ট্রান্সপোজিং মূল অ্যারে পরিবর্তন করে না:

[23]: গ্রেড

আউট[23]:

অ্যারে([[100, 96, 70], [100, 87, 90]])

অনুভূমিক এবং উল্লম্ব স্ট্যাকিং

আপনি আরও কলাম বা আরও সারি যোগ করে অ্যারে একত্রিত করতে পারেন—যা অনুভূমিক স্ট্যাকিং এবং উল্লম্ব স্ট্যাকিং নামে পরিচিত। আসুন গ্রেডের আরেকটি 2by3 অ্যারে তৈরি করি:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[24] তে: grades2 = np.array([[94, 77, 90], [100, 81, 82]])
```

ধরা যাক grades2 হল grades অ্যারেতে থাকা দুই শিক্ষার্থীর জন্য তিনটি অতিরিক্ত পরীক্ষার গ্রেড। আমরা NumPy এর **hstack** (অনুভূমিক স্ট্যাক) ফাংশনের সাথে grades এবং grades2 একত্রিত করতে পারি, অ্যারেগুলিকে একত্রিত করার জন্য একটি টুপল পাস করে।

অতিরিক্ত বন্ধনী প্রয়োজন কারণ hstack একটি যুক্তি আশা করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[25] তে: np.hstack((grades, grades2))
```

আউট[25]:

```
অ্যারে([[100, 96, 90, 98, 97, 90],
           [100, 87, 90, 100, 81, 82]])
```

এরপর, ধরে নেওয়া যাক যে grades2 তিনটি পরীক্ষায় আরও দুই শিক্ষার্থীর গ্রেডকে প্রতিনিধিত্ব করে।

এই ক্ষেত্রে, আমরা NumPy এর **vstack** (উল্লম্ব) এর সাথে grades এবং grades2 একত্রিত করতে পারি।
স্ট্যাক) ফাংশন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[26] তে: np.vstack((grades, grades2))
```

আউট[26]:

```
অ্যারে([[100, 96, 90],
           [100, 87, 90],
           [98, 97, 90],
           [100, 81, 82]])
```

৭.১৪ তথ্য বিজ্ঞানের ভূমিকা: পান্ডাস সিরিজ এবং তথ্যচিত্র

NumPy এর অ্যারে পূর্ণসংখ্যা সূচকের মাধ্যমে অ্যারেস করা সমজাতীয় সংখ্যাসূচক ডেটার জন্য অপ্টিমাইজ করা হয়েছে। ডেটা সায়েল অনন্য চাহিদা উপস্থাপন করে যার জন্য আরও কাস্টমাইজড ডেটা স্ট্রাকচার প্রয়োজন।
বড় ডেটা অ্যাপ্লিকেশনগুলিকে মিশ্র ডেটা টাইপ, কাস্টমাইজড ইনডেক্সিং, অনুপস্থিত ডেটা, ধারাবাহিকভাবে কাঠামোগত নয় এমন ডেটা এবং এমন ডেটা সমর্থন করতে হবে যা

ডাটাবেস এবং ডেটা বিশ্লেষণের জন্য উপযুক্ত ফর্মগুলিতে হেরফের করা উচিত

আপনার ব্যবহৃত প্র্যাকেজগুলি।

এই ধরনের ডেটা নিয়ে কাজ করার জন্য **পান্ডাস** সবচেয়ে জনপ্রিয় লাইব্রেরি। এটি দুটি মূল সংগ্রহ প্রদান করে যা আপনি আমাদের ডেটা সায়েন্সের ভূমিকা বিভাগে এবং ডেটা সায়েন্সের কেস স্টাডি জুড়ে ব্যবহার করবেন - এক-মাত্রিক সংগ্রহের জন্য সিরিজ এবং দ্বি-মাত্রিক সংগ্রহের জন্য **ডেটাফ্রেম**। সিরিজ এবং ডেটাফ্রেমের প্রেক্ষাপটে বহুমাত্রিক ডেটা পরিচালনা করতে আপনি পান্ডাসের **মাল্টিইন্ডেক্স** ব্যবহার করতে পারেন।

ওয়েস ম্যাককিনি ২০০৮ সালে শিল্পে কাজ করার সময় পান্ডা তৈরি করেছিলেন। পান্ডা নামটি "প্যানেল ডেটা" শব্দ থেকে এসেছে, যা সময়ের সাথে সাথে পরিমাপের জন্য ডেটা, যেমন স্টকের দাম বা ঐতিহাসিক তাপমাত্রা রিডিং। ম্যাককিনির এমন একটি লাইব্রেরির প্রয়োজন ছিল যেখানে একই ডেটা স্ট্রাকচার সময় এবং অ-সময় ভিত্তিক উভয় ডেটা পরিচালনা করতে পারে, ডেটা সারিবদ্ধকরণ, অনুপস্থিত ডেটা, সাধারণ ডাটাবেস স্টাইল ডেটা ম্যানিপুলেশন এবং

আরও।⁸

⁸ ম্যাককিনি, ওয়েস। ডেটা বিশ্লেষণের জন্য পাইথন: পান্ডাদের সাথে ডেটা র্যাংলিং, নুমপাই, এবং IPython, pp. 123165. Sebastopol, CA: O'Reilly Media, 2018।

NumPy এবং পান্ডা একে অপরের সাথে ঘনিষ্ঠভাবে সম্পর্কিত। সিরিজ এবং ডেটাফ্রেম "হড়ের নীচে" অ্যারে ব্যবহার করে। সিরিজ এবং ডেটাফ্রেম অনেক NumPy অপারেশনের জন্য বৈধ আর্গুমেন্ট। একইভাবে, অ্যারে অনেক সিরিজ এবং ডেটাফ্রেমের জন্য বৈধ আর্গুমেন্ট।

অপারেশন।

পান্ডাস একটি বিশাল বিষয়—এর ডকুমেন্টেশনের পিডিএফ ফাইল ২০০০ পৃষ্ঠারও বেশি। এই এবং পরবর্তী অধ্যায়ের 'ইনট্রো টু ডেটা সায়েন্স' বিভাগে, আমরা পান্ডাসের একটি ভূমিকা উপস্থাপন করছি। আমরা এর সিরিজ এবং ডেটাফ্রেম সংগ্রহগুলি নিয়ে আলোচনা করব এবং ডেটা প্রস্তুতির সহায়তায় সেগুলি ব্যবহার করব। আপনি দেখতে পাবেন যে সিরিজ এবং ডেটাফ্রেমগুলি আপনার জন্য বিভিন্ন উপায়ে উপাদান নির্বাচন, ফিল্টার/ম্যাপ/রিডুস অপারেশন (কার্যকরী স্টাইল প্রোগ্রামিং এবং বিগ ডেটার কেন্দ্রবিন্দু), গাণিতিক অপারেশন, ডিজুয়ালাইজেশন এবং আরও অনেক কিছুর মতো সাধারণ কাজগুলি সম্পাদন করা সহজ করে তোলে।

⁹ সর্বশেষ পান্ডাস ডকুমেন্টেশনের জন্য, দেখুন

[http://pandas.pydata.org/pandasdocs/stable/।](http://pandas.pydata.org/pandasdocs/stable/)

৭.১৪.১ পান্ডা সিরিজ

সিরিজ হলো একটি উন্নত একমাত্রিক অ্যারে। যেখানে অ্যারেগুলি কেবল শূন্য-ভিত্তিক পূর্ণসংখ্যা সূচক ব্যবহার করে, সিরিজ কাস্টম ইনডেক্সিং সমর্থন করে, এমনকি স্ট্রিংগুলির মতো অ-পূর্ণসংখ্যা সূচকগুলিও অন্তর্ভুক্ত করে। সিরিজ অতিরিক্ত ক্ষমতাও প্রদান করে যা তাদের আরও

অনেক ডেটাসায়েন্স-ভিত্তিক কাজের জন্য সুবিধাজনক। উদাহরণস্বরূপ, সিরিজে ডেটা অনুপস্থিত থাকতে পারে এবং অনেক সিরিজ অপারেশন ডিফল্টরূপে অনুপস্থিত ডেটা উপেক্ষা করে।

ডিফল্ট সূচক সহ একটি সিরিজ তৈরি করা

ডিফল্টরূপে, একটি সিরিজে 0 থেকে ক্রমানুসারে পূর্ণসংখ্যা সূচক থাকে। নিম্নলিখিতটি পূর্ণসংখ্যার তালিকা থেকে শিক্ষার্থীর গ্রেডের একটি সিরিজ তৈরি করে:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

```
[1] তে: pd হিসেবে পান্ডা আমদানি করুন
```

```
[2] তে: গ্রেড = pd.Series([87, 100, 94])
```

ইনিশিয়ালাইজারটি একটি টুপল, একটি অভিধান, একটি অ্যারে, অন্য একটি সিরিজ অথবা একটি একক মানও হতে পারে। আমরা মুহূর্তের মধ্যে একটি একক মান দেখাব।

একটি সিরিজ প্রদর্শন করা হচ্ছে

পান্ডাস দুটি কলাম ফর্ম্যাটে একটি সিরিজ প্রদর্শন করে, যেখানে সূচকগুলি বাম কলামে বাম প্রান্তিককৃত থাকে এবং মানগুলি ডান কলামে ডান প্রান্তিককৃত থাকে। সিরিজ উপাদানগুলি তালিকাভুক্ত করার পরে, পান্ডাস অন্তর্নিহিত অ্যারের উপাদানগুলির ডেটা টাইপ (dtype) দেখায়:

```
[3]: গ্রেড
```

```
আউট[3]:
```

0	৮৭
১	১০০
২	৯৪

```
dtype: int64
```

একই দুটি কলাম ফর্ম্যাটে একটি তালিকা প্রদর্শনের জন্য সংশ্লিষ্ট কোডের তুলনায় এই ফর্ম্যাটে একটি সিরিজ প্রদর্শন করা কতটা সহজ তা লক্ষ্য করুন।

একই মান সম্পন্ন সকল উপাদান দিয়ে একটি সিরিজ তৈরি করা

আপনি এমন একটি সিরিজ উপাদান তৈরি করতে পারেন যার সকলের মান একই:

[কোড ইমেজ দেখতে এখানে ক্লিক করুন](#)

```
[4] তে: pd.Series(98.6, range(3))
```

```
আউট[4]:
```

0	৯৮.৬
১	৯৮.৬

দ্বিতীয় যুক্তি হল একটি একমাত্রিক পুনরাবৃত্তিযোগ্য বস্তু (যেমন একটি তালিকা, একটি অ্যারে বা একটি পরিসর) যাতে সিরিজের সূচক থাকে। সূচকের সংখ্যা উপাদানের সংখ্যা নির্ধারণ করে।

একটি সিরিজের উপাদানগুলিতে অ্যাক্সেস করা

আপনি একটি সিরিজের উপাদানগুলিতে একটি সূচক ধারণকারী বর্গাকার বন্ধনীর মাধ্যমে অ্যাক্সেস করতে পারেন:

[5] তে: গ্রেড[0]

আউট[5]: 87

একটি সিরিজের জন্য বর্ণনামূলক পরিসংখ্যান তৈরি করা

সিরিজ বিভিন্ন বর্ণনামূলক পরিসংখ্যান তৈরি সহ সাধারণ কাজের জন্য অনেক পদ্ধতি প্রদান করে। এখানে আমরা গণনা, গড়, সর্বনিম্ন, সর্বোচ্চ এবং মান (মান) দেখাই।

বিচ্ছিন্নতা):

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[6] তে: grades.count()

আউট[6]: 3

[7] তে: grades.mean()

আউট[7]: 93.66666666666667

[8] তে: grades.min()

আউট[8]: 87

[9] তে: grades.max()

আউট[9]: 100

[10] তে: grades.std()

আউট[10]: 6.506407098647712

এগুলির প্রতিটিই একটি কার্যকরী শ্রেণী ত্রাস। কলিং সিরিজ পদ্ধতি **বর্ণনা** এই সমস্ত পরিসংখ্যান এবং আরও অনেক কিছু তৈরি করে:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

[11] তে: grades.describe()

আউট[11]:

গণনা করা	৩,০০০০০
গড়	৯৩.৬৬৬৬৬৭
স্ট্যান্ডার্ড	৬.৫০৬৪০৭
মিনিট	৮৭,০০০০০
২৫%	৯০,৫০০০০০
৫০%	৯৪,০০০০০
৭৫%	৯৭,০০০০০
সর্বোচ্চ	১০০,০০০০০

dtype: float64

২৫%, ৫০% এবং ৭৫% হলো কোয়ার্টাইল:

- ৫০% সাজানো মানের মধ্যমা প্রতিনিধিত্ব করে।
- ২৫% বাছাই করা মানের প্রথমার্ধের মধ্যমাকে প্রতিনিধিত্ব করে।
- ৭৫% বাছাই করা মানের দ্বিতীয়ার্ধের মধ্যমাকে প্রতিনিধিত্ব করে।

কোয়ার্টাইলের জন্য, যদি দুটি মধ্যম উপাদান থাকে, তাহলে তাদের গড় হবে সেই কোয়ার্টাইলের আমাদের সিরিজে মাত্র তিনটি মান আছে, তাই 25% কোয়ার্টাইল হল গড় ৮৭ এবং ৯৪, এবং ৭৫% কোয়ার্টাইল হল ৯৪ এবং ১০০ এর গড়। একসাথে, ইন্টারকোয়ার্টাইল রেঞ্জ হল ৭৫% কোয়ার্টাইল বিয়োগ করে ২৫% কোয়ার্টাইল, যা আরেকটি বিচ্ছুরণের পরিমাপ, যেমন আদর্শ বিচুতি এবং প্রকরণ। অবশ্যই, কোয়ার্টাইল এবং বৃহত্তর ডেটাসেটে ইন্টারকোয়ার্টাইল রেঞ্জ বেশি কার্যকর।

কাস্টম সূচক সহ একটি সিরিজ তৈরি করা

আপনি index keyword আঙ্গুমেন্ট ব্যবহার করে কাস্টম indexes নির্দিষ্ট করতে পারেন:

কোড ইমেজ দেখতে এখানে ক্লিক করুন

```
[১২] তে: গ্রেড = pd.Series([৮৭, ১০০, ৯৪], সূচক=['ওয়ালি', 'ইভা', 'স্যাম'])
```

n [13]: গ্রেড

আউট[13]:

ওয়ালি	৮৭
ইভা	১০০
একা	৯৪

dtype: int64

এই ক্ষেত্রে, আমরা স্ট্রিং সূচক ব্যবহার করেছি, তবে আপনি অন্যান্য অপরিবর্তনীয় প্রকার ব্যবহার করতে পারেন, যার মধ্যে রয়েছে ০ থেকে শুরু না হওয়া এবং অবিচ্ছিন্ন পূর্ণসংখ্যা। আবার লক্ষ্য করুন, কত সুন্দরভাবে এবং সংক্ষেপে পান্তারা প্রদর্শনের জন্য একটি সিরিজ ফর্ম্যাট করে।