



# ACHTE ÜBUNG

ZUR EINFÜHRUNG IN DIE PROGRAMMIERUNG FÜR COMPUTERLINGUISTEN

■ [Imu.twbk.de](http://Imu.twbk.de)

■ Lesson ID: **GZZ**



Audience

Speaker

## Participate in a lecture

To participate, please enter the Lesson-ID provided by your docent.

GZZ



PARTICIPATE

# WIEDERHOLUNG : ENCODINGS

GZZ

- Was ist hier passiert?

## Anti terror Änderung

- a) ISO-Latin als ASCII angezeigt
- b) ASCII als ISO-Latin angezeigt
- c) UTF-8 als ISO-Latin angezeigt
- d) UTF-8 als ASCII angezeigt



# WIEDERHOLUNG : ENCODINGS

GZZ

- Was ist hier passiert?

## Anti terror Änderung

- a) ISO-Latin als ASCII angezeigt
- b) ASCII als ISO-Latin angezeigt
- c) UTF-8 als ISO-Latin angezeigt
- d) UTF-8 als ASCII angezeigt



# QUIZ

GZZ

- Aus  
„Das ist nicht mein Traumjob“ ist  
„𐄂𐄃𐄄𐄅𐄆𐄇𐄈𐄉𐄊𐄋𐄌𐄍𐄎𐄏𐄐𐄑𐄒𐄓𐄔𐄕𐄖𐄗𐄘𐄙𐄚𐄛𐄜𐄝𐄞𐄟𐄠𐄡𐄢𐄣𐄤𐄥𐄦𐄧𐄨𐄩𐄪𐄫𐄬𐄭𐄮𐄯𐄰𐄱𐄲𐄳𐄴𐄵𐄶𐄷𐄸𐄹𐄺𐄻𐄼𐄽𐄾𐄿𐅀𐅁𐅂𐅃𐅄𐅅𐅆𐅇𐅈𐅉𐅊𐅋𐅌𐅍𐅎𐅏𐅐𐅑𐅒𐅓𐅔𐅕𐅖𐅗𐅘𐅙𐅚𐅛𐅜𐅝𐅞𐅟𐅠𐅡𐅢𐅣𐅤𐅥𐅦𐅧𐅨𐅩𐅪𐅫𐅬𐅭𐅮𐅯𐅰𐅱𐅲𐅳𐅴𐅵𐅶𐅷𐅸𐅹𐅺𐅻𐅼𐅽𐅾𐅿𐆀𐆁𐆂𐆃𐆄𐆅𐆆𐆇𐆈𐆉𐆊𐆋𐆌𐆍𐆎𐆏𐆐𐆑𐆒𐆓𐆔𐆕𐆖𐆗𐆘𐆙𐆚𐆛𐆜𐆝𐆞𐆟𐆠𐆡𐆢𐆣𐆤𐆥𐆦𐆧𐆨𐆩𐆪𐆫𐆬𐆭𐆮𐆯𐆰𐆱𐆲𐆳𐆴𐆵𐆶𐆷𐆸𐆹𐆺𐆻𐆼𐆽𐆾𐆿𐇀𐇁𐇂𐇃𐇄𐇅𐇆𐇇𐇈𐇉𐇊𐇋𐇌𐇍𐇎𐇏𐇐𐇑𐇒𐇓𐇔𐇕𐇖𐇗𐇘𐇙𐇚𐇛𐇜𐇝𐇞𐇟𐇠𐇡𐇢𐇣𐇤𐇥𐇦𐇧𐇨𐇩𐇪𐇫𐇬𐇭𐇮𐇯𐇰𐇱𐇲𐇳𐇴𐇵𐇶𐇷𐇸𐇹𐇺𐇻𐇼𐇽𐇾𐇿𐈀𐈁𐈂𐈃𐈄𐈅𐈆𐈇𐈈𐈉𐈊𐈋𐈌𐈍𐈎𐈏𐈐𐈑𐈒𐈓𐈔𐈕𐈖𐈗𐈘𐈙𐈚𐈛𐈜𐈝𐈞𐈟𐈠𐈡𐈢𐈣𐈤𐈥𐈦𐈧𐈨𐈩𐈪𐈫𐈬𐈭𐈮𐈯𐈰𐈱𐈲𐈳𐈴𐈵𐈶𐈷𐈸𐈹𐈺𐈻𐈼𐈽𐈾𐈿𐉀𐉁𐉂𐉃𐉄𐉅𐉆𐉇𐉈𐉉𐉊𐉋𐉌𐉍𐉎𐉏𐉐𐉑𐉒𐉓𐉔𐉕𐉖𐉗𐉘𐉙𐉚𐉛𐉜𐉝𐉞𐉟𐉠𐉡𐉢𐉣𐉤𐉥𐉦𐉧𐉨𐉩𐉪𐉫𐉬𐉭𐉮𐉯𐉰𐉱𐉲𐉳𐉴𐉵𐉶𐉷𐉸𐉹𐉺𐉻𐉼𐉽𐉾𐉿𐊀𐊁𐊂𐊃𐊄𐊅𐊆𐊇𐊈𐊉𐊊𐊋𐊌𐊍𐊎𐊏𐊐𐊑𐊒𐊓𐊔𐊕𐊖𐊗𐊘𐊙𐊚𐊛𐊜𐊝𐊞𐊟𐊠𐊡𐊢𐊣𐊤𐊥𐊦𐊧𐊨𐊩𐊪𐊫𐊬𐊭𐊮𐊯𐊰𐊱𐊲𐊳𐊴𐊵𐊶𐊷𐊸𐊹𐊺𐊻𐊼𐊽𐊾𐊿𐋀𐋁𐋂𐋃𐋄𐋅𐋆𐋇𐋈𐋉𐋊𐋋𐋌𐋍𐋎𐋏𐋐𐋑𐋒𐋓𐋔𐋕𐋖𐋗𐋘𐋙𐋚𐋛𐋜𐋝𐋞𐋟𐋠𐋡𐋢𐋣𐋤𐋥𐋦𐋧𐋨𐋩𐋪𐋫𐋬𐋭𐋮𐋯𐋰𐋱𐋲𐋳𐋴𐋵𐋶𐋷𐋸𐋹𐋺𐋻𐋼𐋽𐋾𐋿𐌀𐌁𐌂𐌃𐌄𐌅𐌆𐌇𐌈𐌉𐌊𐌋𐌌𐌍𐌎𐌏𐌐𐌑𐌒𐌓𐌔𐌕𐌖𐌗𐌘𐌙𐌚𐌛𐌜𐌝𐌞𐌟𐌠𐌡𐌢𐌣𐌤𐌥𐌦𐌧𐌨𐌩𐌪𐌫𐌬𐌭𐌮𐌯𐌰𐌱𐌲𐌳𐌴𐌵𐌶𐌷𐌸𐌹𐌺𐌻𐌼𐌽𐌾𐌿𐍀𐍁𐍂𐍃𐍄𐍅𐍆𐍇𐍈𐍉𐍊𐍋𐍌𐍍𐍎𐍏𐍐𐍑𐍒𐍓𐍔𐍕𐍖𐍗𐍘𐍙𐍚𐍛𐍜𐍝𐍞𐍟𐍠𐍡𐍢𐍣𐍤𐍥𐍦𐍧𐍨𐍩𐍪𐍫𐍬𐍭𐍮𐍯𐍰𐍱𐍲𐍳𐍴𐍵𐍶𐍷𐍸𐍹𐍺𐍻𐍼𐍽𐍾𐍿𐎀𐎁𐎂𐎃𐎄𐎅𐎆𐎇𐎈𐎉𐎊𐎋𐎌𐎍𐎎𐎏𐎐𐎑𐎒𐎓𐎔𐎕𐎖𐎗𐎘𐎙𐎚𐎛𐎜𐎝𐎞𐎟𐎠𐎡𐎢𐎣𐎤𐎥𐎦𐎧𐎨𐎩𐎪𐎫𐎬𐎭𐎮𐎯𐎰𐎱𐎲𐎳𐎴𐎵𐎶𐎷𐎸𐎹𐎺𐎻𐎼𐎽𐎾𐎿𐏀𐏁𐏂𐏃𐏄𐏅𐏆𐏇𐏈𐏉𐏊𐏋𐏌𐏍𐏎𐏏𐏐𐏑𐏒𐏓𐏔𐏕𐏖𐏗𐏘𐏙𐏚𐏛𐏜𐏝𐏞𐏟𐏠𐏡𐏢𐏣𐏤𐏥𐏦𐏧𐏨𐏩𐏪𐏫𐏬𐏭𐏮𐏯𐏰𐏱𐏲𐏳𐏴𐏵𐏶𐏷𐏸𐏹𐏺𐏻𐏼𐏽𐏾𐏿𐐀𐐁𐐂𐐃𐐄𐐅𐐆𐐇𐐈𐐉𐐊𐐋𐐌𐐍𐐎𐐏𐐐𐐑𐐒𐐓𐐔𐐕𐐖𐐗𐐘𐐙𐐚𐐛𐐜𐐝𐐞𐐟𐐠𐐡𐐢𐐣𐐤𐐥𐐦𐐧𐐨𐐩𐐪𐐫𐐬𐐭𐐮𐐯𐐰𐐱𐐲𐐳𐐴𐐵𐐶𐐷𐐸𐐹𐐺𐐻𐐼𐐽𐐾𐐿𐑀𐑁𐑂𐑃𐑄𐑅𐑆𐑇𐑈𐑉𐑊𐑋𐑌𐑍𐑎𐑏𐑐𐑑𐑒𐑓𐑔𐑕𐑖𐑗𐑘𐑙𐑚𐑛𐑜𐑝𐑞𐑟𐑠𐑡𐑢𐑣𐑤𐑥𐑦𐑧𐑨𐑩𐑪𐑫𐑬𐑭𐑮𐑯𐑰𐑱𐑲𐑳𐑴𐑵𐑶𐑷𐑸𐑹𐑺𐑻𐑼𐑽𐑾𐑿𐒀𐒁𐒂𐒃𐒄𐒅𐒆𐒇𐒈𐒉𐒊𐒋𐒌𐒍𐒎𐒏𐒐𐒑𐒒𐒓𐒔𐒕𐒖𐒗𐒘𐒙𐒚𐒛𐒜𐒝𐒞𐒟𐒠𐒡𐒢𐒣𐒤𐒥𐒦𐒧𐒨𐒩𐒪𐒫𐒬𐒭𐒮𐒯𐒰𐒱𐒲𐒳𐒴𐒵𐒶𐒷𐒸𐒹𐒺𐒻𐒼𐒽𐒾𐒿𐓀𐓁𐓂𐓃𐓄𐓅𐓆𐓇𐓈𐓉𐓊𐓋𐓌𐓍𐓎𐓏𐓐𐓑𐓒𐓓𐓔𐓕𐓖𐓗𐓘𐓙𐓚𐓛𐓜𐓝𐓞𐓟𐓠𐓡𐓢𐓣𐓤𐓥𐓦𐓧𐓨𐓩𐓪𐓫𐓬𐓭𐓮𐓯𐓰𐓱𐓲𐓳𐓴𐓵𐓶𐓷𐓸𐓹𐓺𐓻𐓼𐓽𐓾𐓿𐔀𐔁𐔂𐔃𐔄𐔅𐔆𐔇𐔈𐔉𐔊𐔋𐔌𐔍𐔎𐔏𐔐𐔑𐔒𐔓𐔔𐔕𐔖𐔗𐔘𐔙𐔚𐔛𐔜𐔝𐔞𐔟𐔠𐔡𐔢𐔣𐔤𐔥𐔦𐔧𐔨𐔩𐔪𐔫𐔬𐔭𐔮𐔯𐔰𐔱𐔲𐔳𐔴𐔵𐔶𐔷𐔸𐔹𐔺𐔻𐔼𐔽𐔾𐔿𐕀𐕁𐕂𐕃𐕄𐕅𐕆𐕇𐕈𐕉𐕊𐕋𐕌𐕍𐕎𐕏𐕐𐕑𐕒𐕓𐕔𐕕𐕖𐕗𐕘𐕙𐕚𐕛𐕜𐕝𐕞𐕟𐕠𐕡𐕢𐕣𐕤𐕥𐕦𐕧𐕨𐕩𐕪𐕫𐕬𐕭𐕮𐕯𐕰𐕱𐕲𐕳𐕴𐕵𐕶𐕷𐕸𐕹𐕺𐕻𐕼𐕽𐕾𐕿𐖀𐖁𐖂𐖃𐖄𐖅𐖆𐖇𐖈𐖉𐖊𐖋𐖌𐖍𐖎𐖏𐖐𐖑𐖒𐖓𐖔𐖕𐖖𐖗𐖘𐖙𐖚𐖛𐖜𐖝𐖞𐖟𐖠𐖡𐖢𐖣𐖤𐖥𐖦𐖧𐖨𐖩𐖪𐖫𐖬𐖭𐖮𐖯𐖰𐖱𐖲𐖳𐖴𐖵𐖶𐖷𐖸𐖹𐖺𐖻𐖼𐖽𐖾𐖿𐗀𐗁𐗂𐗃𐗄𐗅𐗆𐗇𐗈𐗉𐗊𐗋𐗌𐗍𐗎𐗏𐗐𐗑𐗒𐗓𐗔𐗕𐗖𐗗𐗘𐗙𐗚𐗛𐗜𐗝𐗞𐗟𐗠𐗡𐗢𐗣𐗤𐗥𐗦𐗧𐗨𐗩𐗪𐗫𐗬𐗭𐗮𐗯𐗰𐗱𐗲𐗳𐗴𐗵𐗶𐗷𐗸𐗹𐗺𐗻𐗼𐗽𐗾𐗿𐘀𐘁𐘂𐘃𐘄𐘅𐘆𐘇𐘈𐘉𐘊𐘋𐘌𐘍𐘎𐘏𐘐𐘑𐘒𐘓𐘔𐘕𐘖𐘗𐘘𐘙𐘚𐘛𐘜𐘝𐘞𐘟𐘠𐘡𐘢𐘣𐘤𐘥𐘦𐘧𐘨𐘩𐘪𐘫𐘬𐘭𐘮𐘯𐘰𐘱𐘲𐘳𐘴𐘵𐘶𐘷𐘸𐘹𐘺𐘻𐘼𐘽𐘾𐘿𐙀𐙁𐙂𐙃𐙄𐙅𐙆𐙇𐙈𐙉𐙊𐙋𐙌𐙍𐙎𐙏𐙐𐙑𐙒𐙓𐙔𐙕𐙖𐙗𐙘𐙙𐙚𐙛𐙜𐙝𐙞𐙟𐙠𐙡𐙢𐙣𐙤𐙥𐙦𐙧𐙨𐙩𐙪𐙫𐙬𐙭𐙮𐙯𐙰𐙱𐙲𐙳𐙴𐙵𐙶𐙷𐙸𐙹𐙺𐙻𐙼𐙽𐙾𐙿𐚀𐚁𐚂𐚃𐚄𐚅𐚆𐚇𐚈𐚉𐚊𐚋𐚌𐚍𐚎𐚏𐚐𐚑𐚒𐚓𐚔𐚕𐚖𐚗𐚘𐚙𐚚𐚛𐚜𐚝𐚞𐚟𐚠𐚡𐚢𐚣𐚤𐚥𐚦𐚧𐚨𐚩𐚪𐚫𐚬𐚭𐚮𐚯𐚰𐚱𐚲𐚳𐚴𐚵𐚶𐚷𐚸𐚹𐚺𐚻𐚼𐚽𐚾𐚿𐛀𐛁𐛂𐛃𐛄𐛅𐛆𐛇𐛈𐛉𐛊𐛋𐛌𐛍𐛎𐛏𐛐𐛑𐛒𐛓𐛔𐛕𐛖𐛗𐛘𐛙𐛚𐛛𐛜𐛝𐛞𐛟𐛠𐛡𐛢𐛣𐛤𐛥𐛦𐛧𐛨𐛩𐛪𐛫𐛬𐛭𐛮𐛯𐛰𐛱𐛲𐛳𐛴𐛵𐛶𐛷𐛸𐛹𐛺𐛻𐛼𐛽𐛾𐛿𐜀𐜁𐜂𐜃𐜄𐜅𐜆𐜇𐜈𐜉𐜊𐜋𐜌𐜍𐜎𐜏𐜐𐜑𐜒𐜓𐜔𐜕𐜖𐜗𐜘𐜙𐜚𐜛𐜜𐜝𐜞𐜟𐜠𐜡𐜢𐜣𐜤𐜥𐜦𐜧𐜨𐜩𐜪𐜫𐜬𐜭𐜮𐜯𐜰𐜱𐜲𐜳𐜴𐜵𐜶𐜷𐜸𐜹𐜺𐜻𐜼𐜽𐜾𐜿𐝀𐝁𐝂𐝃𐝄𐝅𐝆𐝇𐝈𐝉𐝊𐝋𐝌𐝍𐝎𐝏𐝐𐝑𐝒𐝓𐝔𐝕𐝖𐝗𐝘𐝙𐝚𐝛𐝜𐝝𐝞𐝟𐝠𐝡𐝢𐝣𐝤𐝥𐝦𐝧𐝨𐝩𐝪𐝫𐝬𐝭𐝮𐝯𐝰𐝱𐝲𐝳𐝴𐝵𐝶𐝷𐝸𐝹𐝺𐝻𐝼𐝽𐝾𐝿𐞀𐞁𐞂𐞃𐞄𐞅𐞆𐞇𐞈𐞉𐞊𐞋𐞌𐞍𐞎𐞏𐞐𐞑𐞒𐞓𐞔𐞕𐞖𐞗𐞘𐞙𐞚𐞛𐞜𐞝𐞞𐞟𐞠𐞡𐞢𐞣𐞤𐞥𐞦𐞧𐞨𐞩𐞪𐞫𐞬𐞭𐞮𐞯𐞰𐞱𐞲𐞳𐞴𐞵𐞶𐞷𐞸𐞹𐞺𐞻𐞼𐞽𐞾𐞿𐟀𐟁𐟂𐟃𐟄𐟅𐟆𐟇𐟈𐟉𐟊𐟋𐟌𐟍𐟎𐟏𐟐𐟑𐟒𐟓𐟔𐟕𐟖𐟗𐟘𐟙𐟚𐟛𐟜𐟝𐟞𐟟𐟠𐟡𐟢𐟣𐟤𐟥𐟦𐟧𐟨𐟩𐟪𐟫𐟬𐟭𐟮𐟯𐟰𐟱𐟲𐟳𐟴𐟵𐟶𐟷𐟸𐟹𐟺𐟻𐟼𐟽𐟾𐟿𐠀𐠁𐠂𐠃𐠄𐠅𐠆𐠇𐠈𐠉𐠊𐠋𐠌𐠍𐠎𐠏𐠐𐠑𐠒𐠓𐠔𐠕𐠖𐠗𐠘𐠙𐠚𐠛𐠜𐠝𐠞𐠟𐠠𐠡𐠢𐠣𐠤𐠥𐠦𐠧𐠨𐠩𐠪𐠫𐠬𐠭𐠮𐠯𐠰𐠱𐠲𐠳𐠴𐠵𐠶𐠷𐠸𐠹𐠺𐠻𐠼𐠽𐠾𐠿𐡀𐡁𐡂𐡃𐡄𐡅𐡆𐡇𐡈𐡉𐡊𐡋𐡌𐡍𐡎𐡏𐡐𐡑𐡒𐡓𐡔𐡕𐡖𐡗𐡘𐡙𐡚𐡛𐡜𐡝𐡞𐡟𐡠𐡡𐡢𐡣𐡤𐡥𐡦𐡧𐡨𐡩𐡪𐡫𐡬𐡭𐡮𐡯𐡰𐡱𐡲𐡳𐡴𐡵𐡶𐡷𐡸𐡹𐡺𐡻𐡼𐡽𐡾𐡿𐢀𐢁𐢂𐢃𐢄𐢅𐢆𐢇𐢈𐢉𐢊𐢋𐢌𐢍𐢎𐢏𐢐𐢑𐢒𐢓𐢔𐢕𐢖𐢗𐢘𐢙𐢚𐢛𐢜𐢝𐢞𐢟𐢠𐢡𐢢𐢣𐢤𐢥𐢦𐢧𐢨𐢩𐢪𐢫𐢬𐢭𐢮𐢯𐢰𐢱𐢲𐢳𐢴𐢵𐢶𐢷𐢸𐢹𐢺𐢻𐢼𐢽𐢾𐢿𐣀𐣁𐣂𐣃𐣄𐣅𐣆𐣇𐣈𐣉𐣊𐣋𐣌𐣍𐣎𐣏𐣐𐣑𐣒𐣓𐣔𐣕𐣖𐣗𐣘𐣙𐣚𐣛𐣜𐣝𐣞𐣟𐣠𐣡𐣢𐣣𐣤𐣥𐣦𐣧𐣨𐣩𐣪𐣫𐣬𐣭𐣮𐣯𐣰𐣱𐣲𐣳𐣴𐣵𐣶𐣷𐣸𐣹𐣺𐣻𐣼𐣽𐣾𐣿𐤀𐤁𐤂𐤃𐤄𐤅𐤆𐤇𐤈𐤉𐤊𐤋𐤌𐤍𐤎𐤏𐤐𐤑𐤒𐤓𐤔𐤕𐤖𐤗𐤘𐤙𐤚𐤛𐤜𐤝𐤞𐤟𐤠𐤡𐤢𐤣𐤤𐤥𐤦𐤧𐤨𐤩𐤪𐤫𐤬𐤭𐤮𐤯𐤰𐤱𐤲𐤳𐤴𐤵𐤶𐤷𐤸𐤹𐤺𐤻𐤼𐤽𐤾𐤿𐥀𐥁𐥂𐥃𐥄𐥅𐥆𐥇𐥈𐥉𐥊𐥋𐥌𐥍𐥎𐥏𐥐𐥑𐥒𐥓𐥔𐥕𐥖𐥗𐥘𐥙𐥚𐥛𐥜𐥝𐥞𐥟𐥠𐥡𐥢𐥣𐥤𐥥𐥦𐥧𐥨𐥩𐥪𐥫𐥬𐥭𐥮𐥯𐥰𐥱𐥲𐥳𐥴𐥵𐥶𐥷𐥸𐥹𐥺𐥻𐥼𐥽𐥾𐥿𐦀𐦁𐦂𐦃𐦄𐦅𐦆𐦇𐦈𐦉𐦊𐦋𐦌𐦍𐦎𐦏𐦐𐦑𐦒𐦓𐦔𐦕𐦖𐦗𐦘𐦙𐦚𐦛𐦜𐦝𐦞𐦟𐦠𐦡𐦢𐦣𐦤𐦥𐦦𐦧𐦨𐦩𐦪𐦫𐦬𐦭𐦮𐦯𐦰𐦱𐦲𐦳𐦴𐦵𐦶𐦷𐦸𐦹𐦺𐦻𐦼𐦽𐦾𐦿𐧀𐧁𐧂𐧃𐧄𐧅𐧆𐧇𐧈𐧉𐧊𐧋𐧌𐧍𐧎𐧏𐧐𐧑𐧒𐧓𐧔𐧕𐧖𐧗𐧘𐧙𐧚𐧛𐧜𐧝𐧞𐧟𐧠𐧡𐧢𐧣𐧤𐧥𐧦𐧧𐧨𐧩𐧪𐧫𐧬𐧭𐧮𐧯𐧰𐧱𐧲𐧳𐧴𐧵𐧶𐧷𐧸𐧹𐧺𐧻𐧼𐧽𐧾𐧿𐨀𐨁𐨂𐨃𐨄𐨅𐨆𐨇𐨈𐨉𐨊𐨋𐨌𐨍𐨎𐨏𐨐𐨑𐨒𐨓𐨔𐨕𐨖𐨗𐨘𐨙𐨚𐨛𐨜𐨝𐨞𐨟𐨠𐨡𐨢𐨣𐨤𐨥𐨦𐨧𐨨𐨩𐨪𐨫𐨬𐨭𐨮𐨯𐨰𐨱𐨲𐨳𐨴𐨵𐨶𐨷𐨹𐨺𐨸𐨻𐨼𐨽𐨾𐨿𐩀𐩁𐩂𐩃𐩄𐩅𐩆𐩇𐩈𐩉𐩊𐩋𐩌𐩍𐩎𐩏𐩐𐩑𐩒𐩓𐩔𐩕𐩖𐩗𐩘𐩙𐩚𐩛𐩜𐩝𐩞𐩟𐩠𐩡𐩢𐩣𐩤𐩥𐩦𐩧𐩨𐩩𐩪𐩫𐩬𐩭𐩮𐩯𐩰𐩱𐩲𐩳𐩴𐩵𐩶𐩷𐩸𐩹𐩺𐩻𐩼𐩽𐩾𐩿𐪀𐪁𐪂𐪃𐪄𐪅𐪆𐪇𐪈𐪉𐪊𐪋𐪌𐪍𐪎𐪏𐪐𐪑𐪒𐪓𐪔𐪕𐪖𐪗𐪘𐪙𐪚𐪛𐪜𐪝𐪞𐪟𐪠𐪡𐪢𐪣𐪤𐪥𐪦𐪧𐪨𐪩𐪪𐪫𐪬𐪭𐪮𐪯𐪰𐪱𐪲𐪳𐪴𐪵𐪶𐪷𐪸𐪹𐪺𐪻𐪼𐪽𐪾𐪿𐫀𐫁𐫂𐫃𐫄𐫅𐫆𐫇𐫈𐫉𐫊𐫋𐫌𐫍𐫎𐫏𐫐𐫑𐫒𐫓𐫔𐫕𐫖𐫗𐫘𐫙𐫚𐫛𐫜𐫝𐫞𐫟𐫠𐫡𐫢𐫣𐫤𐫦𐫥𐫧𐫨𐫩𐫪𐫫𐫬𐫭𐫮𐫯𐫰𐫱𐫲𐫳𐫴𐫵𐫶𐫷𐫸𐫹𐫺𐫻𐫼𐫽𐫾𐫿𐬀𐬁𐬂𐬃𐬄𐬅𐬆𐬇𐬈𐬉𐬊𐬋𐬌𐬍𐬎𐬏𐬐𐬑𐬒𐬓𐬔𐬕𐬖𐬗𐬘𐬙𐬚𐬛𐬜𐬝𐬞𐬟𐬠𐬡𐬢𐬣𐬤𐬥𐬦𐬧𐬨𐬩𐬪𐬫𐬬𐬭𐬮𐬯𐬰𐬱𐬲𐬳𐬴𐬵𐬶𐬷𐬸𐬹𐬺𐬻𐬼𐬽𐬾𐬿𐭀𐭁𐭂𐭃𐭄𐭅𐭆𐭇𐭈𐭉𐭊𐭋𐭌𐭍𐭎𐭏𐭐𐭑𐭒𐭓𐭔𐭕𐭖𐭗𐭘𐭙𐭚𐭛𐭜𐭝𐭞𐭟𐭠𐭡𐭢𐭣𐭤𐭥𐭦𐭧𐭨𐭩𐭪𐭫𐭬𐭭𐭮𐭯𐭰𐭱𐭲𐭳𐭴𐭵𐭶𐭷𐭸𐭹𐭺𐭻𐭼𐭽𐭾𐭿𐮀𐮁𐮂𐮃𐮄𐮅𐮆𐮇𐮈𐮉𐮊𐮋𐮌𐮍𐮎𐮏𐮐𐮑𐮒𐮓𐮔𐮕𐮖𐮗𐮘𐮙𐮚𐮛𐮜𐮝𐮞𐮟𐮠𐮡𐮢𐮣𐮤𐮥𐮦𐮧𐮨𐮩𐮪𐮫𐮬𐮭𐮮𐮯𐮰𐮱𐮲𐮳𐮴𐮵𐮶𐮷𐮸𐮹𐮺𐮻𐮼𐮽𐮾𐮿𐯀𐯁𐯂𐯃𐯄𐯅𐯆𐯇𐯈𐯉𐯊𐯋𐯌𐯍𐯎𐯏𐯐𐯑𐯒𐯓𐯔𐯕𐯖𐯗𐯘𐯙𐯚𐯛𐯜𐯝𐯞𐯟𐯠𐯡𐯢𐯣𐯤𐯥𐯦𐯧𐯨𐯩𐯪𐯫𐯬𐯭𐯮𐯯𐯰𐯱𐯲𐯳𐯴𐯵𐯶𐯷𐯸𐯹𐯺𐯻𐯼𐯽𐯾𐯿𐰀𐰁𐰂𐰃𐰄𐰅𐰆𐰇𐰈𐰉𐰊𐰋𐰌𐰍𐰎𐰏𐰐𐰑𐰒𐰓𐰔𐰕𐰖𐰗𐰘𐰙𐰚𐰛𐰜𐰝𐰞𐰟𐰠𐰡𐰢𐰣𐰤𐰥𐰦𐰧𐰨𐰩𐰪𐰫𐰬𐰭𐰮𐰯𐰰𐰱𐰲𐰳𐰴𐰵𐰶𐰷𐰸𐰹𐰺𐰻𐰼𐰽𐰾𐰿𐱀𐱁𐱂𐱃𐱄𐱅𐱆𐱇𐱈𐱉𐱊𐱋𐱌𐱍𐱎𐱏𐱐𐱑𐱒𐱓𐱔𐱕𐱖𐱗𐱘𐱙𐱚𐱛𐱜𐱝𐱞𐱟𐱠𐱡𐱢𐱣𐱤𐱥𐱦𐱧𐱨𐱩𐱪𐱫𐱬𐱭𐱮𐱯𐱰𐱱𐱲𐱳𐱴𐱵𐱶𐱷𐱸𐱹𐱺𐱻𐱼𐱽𐱾𐱿𐲀𐲁𐲂𐲃𐲄𐲅𐲆𐲇𐲈𐲉𐲊𐲋𐲌𐲍𐲎𐲏𐲐𐲑𐲒𐲓𐲔𐲕𐲖𐲗𐲘𐲙𐲚𐲛𐲜𐲝𐲞𐲟𐲠𐲡𐲢𐲣𐲤𐲥𐲦𐲧𐲨𐲩𐲪𐲫𐲬𐲭𐲮𐲯𐲰𐲱𐲲𐲳𐲴𐲵𐲶𐲷𐲸𐲹𐲺𐲻𐲼𐲽𐲾𐲿𐳀𐳁𐳂𐳃𐳄𐳅𐳆𐳇𐳈𐳉𐳊𐳋𐳌𐳍𐳎𐳏𐳐𐳑𐳒𐳓𐳔𐳕𐳖𐳗𐳘𐳙𐳚𐳛𐳜𐳝𐳞𐳟𐳠𐳡𐳢𐳣𐳤𐳥𐳦𐳧𐳨𐳩𐳪𐳫𐳬𐳭𐳮𐳯𐳰𐳱𐳲𐳳𐳴𐳵𐳶𐳷𐳸

# QUIZ

GZZ

- Aus  
„Das ist nicht mein Traumjob“ ist  
„𐄂𐄃𐄄𐄅𐄆𐄇𐄈𐄉𐄊𐄋𐄌𐄍𐄎𐄏𐄐𐄑𐄒𐄓𐄔𐄕𐄖𐄗𐄘𐄙𐄚𐄛𐄜𐄝𐄞𐄟𐄠𐄡𐄢𐄣𐄤𐄥𐄦𐄧𐄨𐄩𐄪𐄫𐄬𐄭𐄮𐄯𐄰𐄱𐄲𐄳𐄴𐄵𐄶𐄷𐄸𐄹𐄺𐄻𐄼𐄽𐄾𐄿𐅀𐅁𐅂𐅃𐅄𐅅𐅆𐅇𐅈𐅉𐅊𐅋𐅌𐅍𐅎𐅏𐅐𐅑𐅒𐅓𐅔𐅕𐅖𐅗𐅘𐅙𐅚𐅛𐅜𐅝𐅞𐅟𐅠𐅡𐅢𐅣𐅤𐅥𐅦𐅧𐅨𐅩𐅪𐅫𐅬𐅭𐅮𐅯𐅰𐅱𐅲𐅳𐅴𐅵𐅶𐅷𐅸𐅹𐅺𐅻𐅼𐅽𐅾𐅿𐆀𐆁𐆂𐆃𐆄𐆅𐆆𐆇𐆈𐆉𐆊𐆋𐆌𐆍𐆎𐆏𐆐𐆑𐆒𐆓𐆔𐆕𐆖𐆗𐆘𐆙𐆚𐆛𐆜𐆝𐆞𐆟𐆠𐆡𐆢𐆣𐆤𐆥𐆦𐆧𐆨𐆩𐆪𐆫𐆬𐆭𐆮𐆯𐆰𐆱𐆲𐆳𐆴𐆵𐆶𐆷𐆸𐆹𐆺𐆻𐆼𐆽𐆾𐆿𐇀𐇁𐇂𐇃𐇄𐇅𐇆𐇇𐇈𐇉𐇊𐇋𐇌𐇍𐇎𐇏𐇐𐇑𐇒𐇓𐇔𐇕𐇖𐇗𐇘𐇙𐇚𐇛𐇜𐇝𐇞𐇟𐇠𐇡𐇢𐇣𐇤𐇥𐇦𐇧𐇨𐇩𐇪𐇫𐇬𐇭𐇮𐇯𐇰𐇱𐇲𐇳𐇴𐇵𐇶𐇷𐇸𐇹𐇺𐇻𐇼𐇽𐇾𐇿𐈀𐈁𐈂𐈃𐈄𐈅𐈆𐈇𐈈𐈉𐈊𐈋𐈌𐈍𐈎𐈏𐈐𐈑𐈒𐈓𐈔𐈕𐈖𐈗𐈘𐈙𐈚𐈛𐈜𐈝𐈞𐈟𐈠𐈡𐈢𐈣𐈤𐈥𐈦𐈧𐈨𐈩𐈪𐈫𐈬𐈭𐈮𐈯𐈰𐈱𐈲𐈳𐈴𐈵𐈶𐈷𐈸𐈹𐈺𐈻𐈼𐈽𐈾𐈿𐉀𐉁𐉂𐉃𐉄𐉅𐉆𐉇𐉈𐉉𐉊𐉋𐉌𐉍𐉎𐉏𐉐𐉑𐉒𐉓𐉔𐉕𐉖𐉗𐉘𐉙𐉚𐉛𐉜𐉝𐉞𐉟𐉠𐉡𐉢𐉣𐉤𐉥𐉦𐉧𐉨𐉩𐉪𐉫𐉬𐉭𐉮𐉯𐉰𐉱𐉲𐉳𐉴𐉵𐉶𐉷𐉸𐉹𐉺𐉻𐉼𐉽𐉾𐉿𐊀𐊁𐊂𐊃𐊄𐊅𐊆𐊇𐊈𐊉𐊊𐊋𐊌𐊍𐊎𐊏𐊐𐊑𐊒𐊓𐊔𐊕𐊖𐊗𐊘𐊙𐊚𐊛𐊜𐊝𐊞𐊟𐊠𐊡𐊢𐊣𐊤𐊥𐊦𐊧𐊨𐊩𐊪𐊫𐊬𐊭𐊮𐊯𐊰𐊱𐊲𐊳𐊴𐊵𐊶𐊷𐊸𐊹𐊺𐊻𐊼𐊽𐊾𐊿𐋀𐋁𐋂𐋃𐋄𐋅𐋆𐋇𐋈𐋉𐋊𐋋𐋌𐋍𐋎𐋏𐋐𐋑𐋒𐋓𐋔𐋕𐋖𐋗𐋘𐋙𐋚𐋛𐋜𐋝𐋞𐋟𐋠𐋡𐋢𐋣𐋤𐋥𐋦𐋧𐋨𐋩𐋪𐋫𐋬𐋭𐋮𐋯𐋰𐋱𐋲𐋳𐋴𐋵𐋶𐋷𐋸𐋹𐋺𐋻𐋼𐋽𐋾𐋿𐌀𐌁𐌂𐌃𐌄𐌅𐌆𐌇𐌈𐌉𐌊𐌋𐌌𐌍𐌎𐌏𐌐𐌑𐌒𐌓𐌔𐌕𐌖𐌗𐌘𐌙𐌚𐌛𐌜𐌝𐌞𐌟𐌠𐌡𐌢𐌣𐌤𐌥𐌦𐌧𐌨𐌩𐌪𐌫𐌬𐌭𐌮𐌯𐌰𐌱𐌲𐌳𐌴𐌵𐌶𐌷𐌸𐌹𐌺𐌻𐌼𐌽𐌾𐌿𐍀𐍁𐍂𐍃𐍄𐍅𐍆𐍇𐍈𐍉𐍊𐍋𐍌𐍍𐍎𐍏𐍐𐍑𐍒𐍓𐍔𐍕𐍖𐍗𐍘𐍙𐍚𐍛𐍜𐍝𐍞𐍟𐍠𐍡𐍢𐍣𐍤𐍥𐍦𐍧𐍨𐍩𐍪𐍫𐍬𐍭𐍮𐍯𐍰𐍱𐍲𐍳𐍴𐍵𐍶𐍷𐍸𐍹𐍺𐍻𐍼𐍽𐍾𐍿𐎀𐎁𐎂𐎃𐎄𐎅𐎆𐎇𐎈𐎉𐎊𐎋𐎌𐎍𐎎𐎏𐎐𐎑𐎒𐎓𐎔𐎕𐎖𐎗𐎘𐎙𐎚𐎛𐎜𐎝𐎞𐎟𐎠𐎡𐎢𐎣𐎤𐎥𐎦𐎧𐎨𐎩𐎪𐎫𐎬𐎭𐎮𐎯𐎰𐎱𐎲𐎳𐎴𐎵𐎶𐎷𐎸𐎹𐎺𐎻𐎼𐎽𐎾𐎿𐏀𐏁𐏂𐏃𐏄𐏅𐏆𐏇𐏈𐏉𐏊𐏋𐏌𐏍𐏎𐏏𐏐𐏑𐏒𐏓𐏔𐏕𐏖𐏗𐏘𐏙𐏚𐏛𐏜𐏝𐏞𐏟𐏠𐏡𐏢𐏣𐏤𐏥𐏦𐏧𐏨𐏩𐏪𐏫𐏬𐏭𐏮𐏯𐏰𐏱𐏲𐏳𐏴𐏵𐏶𐏷𐏸𐏹𐏺𐏻𐏼𐏽𐏾𐏿𐐀𐐁𐐂𐐃𐐄𐐅𐐆𐐇𐐈𐐉𐐊𐐋𐐌𐐍𐐎𐐏𐐐𐐑𐐒𐐓𐐔𐐕𐐖𐐗𐐘𐐙𐐚𐐛𐐜𐐝𐐞𐐟𐐠𐐡𐐢𐐣𐐤𐐥𐐦𐐧𐐨𐐩𐐪𐐫𐐬𐐭𐐮𐐯𐐰𐐱𐐲𐐳𐐴𐐵𐐶𐐷𐐸𐐹𐐺𐐻𐐼𐐽𐐾𐐿𐑀𐑁𐑂𐑃𐑄𐑅𐑆𐑇𐑈𐑉𐑊𐑋𐑌𐑍𐑎𐑏𐑐𐑑𐑒𐑓𐑔𐑕𐑖𐑗𐑘𐑙𐑚𐑛𐑜𐑝𐑞𐑟𐑠𐑡𐑢𐑣𐑤𐑥𐑦𐑧𐑨𐑩𐑪𐑫𐑬𐑭𐑮𐑯𐑰𐑱𐑲𐑳𐑴𐑵𐑶𐑷𐑸𐑹𐑺𐑻𐑼𐑽𐑾𐑿𐒀𐒁𐒂𐒃𐒄𐒅𐒆𐒇𐒈𐒉𐒊𐒋𐒌𐒍𐒎𐒏𐒐𐒑𐒒𐒓𐒔𐒕𐒖𐒗𐒘𐒙𐒚𐒛𐒜𐒝𐒞𐒟𐒠𐒡𐒢𐒣𐒤𐒥𐒦𐒧𐒨𐒩𐒪𐒫𐒬𐒭𐒮𐒯𐒰𐒱𐒲𐒳𐒴𐒵𐒶𐒷𐒸𐒹𐒺𐒻𐒼𐒽𐒾𐒿𐓀𐓁𐓂𐓃𐓄𐓅𐓆𐓇𐓈𐓉𐓊𐓋𐓌𐓍𐓎𐓏𐓐𐓑𐓒𐓓𐓔𐓕𐓖𐓗𐓘𐓙𐓚𐓛𐓜𐓝𐓞𐓟𐓠𐓡𐓢𐓣𐓤𐓥𐓦𐓧𐓨𐓩𐓪𐓫𐓬𐓭𐓮𐓯𐓰𐓱𐓲𐓳𐓴𐓵𐓶𐓷𐓸𐓹𐓺𐓻𐓼𐓽𐓾𐓿𐔀𐔁𐔂𐔃𐔄𐔅𐔆𐔇𐔈𐔉𐔊𐔋𐔌𐔍𐔎𐔏𐔐𐔑𐔒𐔓𐔔𐔕𐔖𐔗𐔘𐔙𐔚𐔛𐔜𐔝𐔞𐔟𐔠𐔡𐔢𐔣𐔤𐔥𐔦𐔧𐔨𐔩𐔪𐔫𐔬𐔭𐔮𐔯𐔰𐔱𐔲𐔳𐔴𐔵𐔶𐔷𐔸𐔹𐔺𐔻𐔼𐔽𐔾𐔿𐕀𐕁𐕂𐕃𐕄𐕅𐕆𐕇𐕈𐕉𐕊𐕋𐕌𐕍𐕎𐕏𐕐𐕑𐕒𐕓𐕔𐕕𐕖𐕗𐕘𐕙𐕚𐕛𐕜𐕝𐕞𐕟𐕠𐕡𐕢𐕣𐕤𐕥𐕦𐕧𐕨𐕩𐕪𐕫𐕬𐕭𐕮𐕯𐕰𐕱𐕲𐕳𐕴𐕵𐕶𐕷𐕸𐕹𐕺𐕻𐕼𐕽𐕾𐕿𐖀𐖁𐖂𐖃𐖄𐖅𐖆𐖇𐖈𐖉𐖊𐖋𐖌𐖍𐖎𐖏𐖐𐖑𐖒𐖓𐖔𐖕𐖖𐖗𐖘𐖙𐖚𐖛𐖜𐖝𐖞𐖟𐖠𐖡𐖢𐖣𐖤𐖥𐖦𐖧𐖨𐖩𐖪𐖫𐖬𐖭𐖮𐖯𐖰𐖱𐖲𐖳𐖴𐖵𐖶𐖷𐖸𐖹𐖺𐖻𐖼𐖽𐖾𐖿𐗀𐗁𐗂𐗃𐗄𐗅𐗆𐗇𐗈𐗉𐗊𐗋𐗌𐗍𐗎𐗏𐗐𐗑𐗒𐗓𐗔𐗕𐗖𐗗𐗘𐗙𐗚𐗛𐗜𐗝𐗞𐗟𐗠𐗡𐗢𐗣𐗤𐗥𐗦𐗧𐗨𐗩𐗪𐗫𐗬𐗭𐗮𐗯𐗰𐗱𐗲𐗳𐗴𐗵𐗶𐗷𐗸𐗹𐗺𐗻𐗼𐗽𐗾𐗿𐘀𐘁𐘂𐘃𐘄𐘅𐘆𐘇𐘈𐘉𐘊𐘋𐘌𐘍𐘎𐘏𐘐𐘑𐘒𐘓𐘔𐘕𐘖𐘗𐘘𐘙𐘚𐘛𐘜𐘝𐘞𐘟𐘠𐘡𐘢𐘣𐘤𐘥𐘦𐘧𐘨𐘩𐘪𐘫𐘬𐘭𐘮𐘯𐘰𐘱𐘲𐘳𐘴𐘵𐘶𐘷𐘸𐘹𐘺𐘻𐘼𐘽𐘾𐘿𐙀𐙁𐙂𐙃𐙄𐙅𐙆𐙇𐙈𐙉𐙊𐙋𐙌𐙍𐙎𐙏𐙐𐙑𐙒𐙓𐙔𐙕𐙖𐙗𐙘𐙙𐙚𐙛𐙜𐙝𐙞𐙟𐙠𐙡𐙢𐙣𐙤𐙥𐙦𐙧𐙨𐙩𐙪𐙫𐙬𐙭𐙮𐙯𐙰𐙱𐙲𐙳𐙴𐙵𐙶𐙷𐙸𐙹𐙺𐙻𐙼𐙽𐙾𐙿𐚀𐚁𐚂𐚃𐚄𐚅𐚆𐚇𐚈𐚉𐚊𐚋𐚌𐚍𐚎𐚏𐚐𐚑𐚒𐚓𐚔𐚕𐚖𐚗𐚘𐚙𐚚𐚛𐚜𐚝𐚞𐚟𐚠𐚡𐚢𐚣𐚤𐚥𐚦𐚧𐚨𐚩𐚪𐚫𐚬𐚭𐚮𐚯𐚰𐚱𐚲𐚳𐚴𐚵𐚶𐚷𐚸𐚹𐚺𐚻𐚼𐚽𐚾𐚿𐛀𐛁𐛂𐛃𐛄𐛅𐛆𐛇𐛈𐛉𐛊𐛋𐛌𐛍𐛎𐛏𐛐𐛑𐛒𐛓𐛔𐛕𐛖𐛗𐛘𐛙𐛚𐛛𐛜𐛝𐛞𐛟𐛠𐛡𐛢𐛣𐛤𐛥𐛦𐛧𐛨𐛩𐛪𐛫𐛬𐛭𐛮𐛯𐛰𐛱𐛲𐛳𐛴𐛵𐛶𐛷𐛸𐛹𐛺𐛻𐛼𐛽𐛾𐛿𐜀𐜁𐜂𐜃𐜄𐜅𐜆𐜇𐜈𐜉𐜊𐜋𐜌𐜍𐜎𐜏𐜐𐜑𐜒𐜓𐜔𐜕𐜖𐜗𐜘𐜙𐜚𐜛𐜜𐜝𐜞𐜟𐜠𐜡𐜢𐜣𐜤𐜥𐜦𐜧𐜨𐜩𐜪𐜫𐜬𐜭𐜮𐜯𐜰𐜱𐜲𐜳𐜴𐜵𐜶𐜷𐜸𐜹𐜺𐜻𐜼𐜽𐜾𐜿𐝀𐝁𐝂𐝃𐝄𐝅𐝆𐝇𐝈𐝉𐝊𐝋𐝌𐝍𐝎𐝏𐝐𐝑𐝒𐝓𐝔𐝕𐝖𐝗𐝘𐝙𐝚𐝛𐝜𐝝𐝞𐝟𐝠𐝡𐝢𐝣𐝤𐝥𐝦𐝧𐝨𐝩𐝪𐝫𐝬𐝭𐝮𐝯𐝰𐝱𐝲𐝳𐝴𐝵𐝶𐝷𐝸𐝹𐝺𐝻𐝼𐝽𐝾𐝿𐞀𐞁𐞂𐞃𐞄𐞅𐞆𐞇𐞈𐞉𐞊𐞋𐞌𐞍𐞎𐞏𐞐𐞑𐞒𐞓𐞔𐞕𐞖𐞗𐞘𐞙𐞚𐞛𐞜𐞝𐞞𐞟𐞠𐞡𐞢𐞣𐞤𐞥𐞦𐞧𐞨𐞩𐞪𐞫𐞬𐞭𐞮𐞯𐞰𐞱𐞲𐞳𐞴𐞵𐞶𐞷𐞸𐞹𐞺𐞻𐞼𐞽𐞾𐞿𐟀𐟁𐟂𐟃𐟄𐟅𐟆𐟇𐟈𐟉𐟊𐟋𐟌𐟍𐟎𐟏𐟐𐟑𐟒𐟓𐟔𐟕𐟖𐟗𐟘𐟙𐟚𐟛𐟜𐟝𐟞𐟟𐟠𐟡𐟢𐟣𐟤𐟥𐟦𐟧𐟨𐟩𐟪𐟫𐟬𐟭𐟮𐟯𐟰𐟱𐟲𐟳𐟴𐟵𐟶𐟷𐟸𐟹𐟺𐟻𐟼𐟽𐟾𐟿𐠀𐠁𐠂𐠃𐠄𐠅𐠆𐠇𐠈𐠉𐠊𐠋𐠌𐠍𐠎𐠏𐠐𐠑𐠒𐠓𐠔𐠕𐠖𐠗𐠘𐠙𐠚𐠛𐠜𐠝𐠞𐠟𐠠𐠡𐠢𐠣𐠤𐠥𐠦𐠧𐠨𐠩𐠪𐠫𐠬𐠭𐠮𐠯𐠰𐠱𐠲𐠳𐠴𐠵𐠶𐠷𐠸𐠹𐠺𐠻𐠼𐠽𐠾𐠿𐡀𐡁𐡂𐡃𐡄𐡅𐡆𐡇𐡈𐡉𐡊𐡋𐡌𐡍𐡎𐡏𐡐𐡑𐡒𐡓𐡔𐡕𐡖𐡗𐡘𐡙𐡚𐡛𐡜𐡝𐡞𐡟𐡠𐡡𐡢𐡣𐡤𐡥𐡦𐡧𐡨𐡩𐡪𐡫𐡬𐡭𐡮𐡯𐡰𐡱𐡲𐡳𐡴𐡵𐡶𐡷𐡸𐡹𐡺𐡻𐡼𐡽𐡾𐡿𐢀𐢁𐢂𐢃𐢄𐢅𐢆𐢇𐢈𐢉𐢊𐢋𐢌𐢍𐢎𐢏𐢐𐢑𐢒𐢓𐢔𐢕𐢖𐢗𐢘𐢙𐢚𐢛𐢜𐢝𐢞𐢟𐢠𐢡𐢢𐢣𐢤𐢥𐢦𐢧𐢨𐢩𐢪𐢫𐢬𐢭𐢮𐢯𐢰𐢱𐢲𐢳𐢴𐢵𐢶𐢷𐢸𐢹𐢺𐢻𐢼𐢽𐢾𐢿𐣀𐣁𐣂𐣃𐣄𐣅𐣆𐣇𐣈𐣉𐣊𐣋𐣌𐣍𐣎𐣏𐣐𐣑𐣒𐣓𐣔𐣕𐣖𐣗𐣘𐣙𐣚𐣛𐣜𐣝𐣞𐣟𐣠𐣡𐣢𐣣𐣤𐣥𐣦𐣧𐣨𐣩𐣪𐣫𐣬𐣭𐣮𐣯𐣰𐣱𐣲𐣳𐣴𐣵𐣶𐣷𐣸𐣹𐣺𐣻𐣼𐣽𐣾𐣿𐤀𐤁𐤂𐤃𐤄𐤅𐤆𐤇𐤈𐤉𐤊𐤋𐤌𐤍𐤎𐤏𐤐𐤑𐤒𐤓𐤔𐤕𐤖𐤗𐤘𐤙𐤚𐤛𐤜𐤝𐤞𐤟𐤠𐤡𐤢𐤣𐤤𐤥𐤦𐤧𐤨𐤩𐤪𐤫𐤬𐤭𐤮𐤯𐤰𐤱𐤲𐤳𐤴𐤵𐤶𐤷𐤸𐤹𐤺𐤻𐤼𐤽𐤾𐤿𐥀𐥁𐥂𐥃𐥄𐥅𐥆𐥇𐥈𐥉𐥊𐥋𐥌𐥍𐥎𐥏𐥐𐥑𐥒𐥓𐥔𐥕𐥖𐥗𐥘𐥙𐥚𐥛𐥜𐥝𐥞𐥟𐥠𐥡𐥢𐥣𐥤𐥥𐥦𐥧𐥨𐥩𐥪𐥫𐥬𐥭𐥮𐥯𐥰𐥱𐥲𐥳𐥴𐥵𐥶𐥷𐥸𐥹𐥺𐥻𐥼𐥽𐥾𐥿𐦀𐦁𐦂𐦃𐦄𐦅𐦆𐦇𐦈𐦉𐦊𐦋𐦌𐦍𐦎𐦏𐦐𐦑𐦒𐦓𐦔𐦕𐦖𐦗𐦘𐦙𐦚𐦛𐦜𐦝𐦞𐦟𐦠𐦡𐦢𐦣𐦤𐦥𐦦𐦧𐦨𐦩𐦪𐦫𐦬𐦭𐦮𐦯𐦰𐦱𐦲𐦳𐦴𐦵𐦶𐦷𐦸𐦹𐦺𐦻𐦼𐦽𐦾𐦿𐧀𐧁𐧂𐧃𐧄𐧅𐧆𐧇𐧈𐧉𐧊𐧋𐧌𐧍𐧎𐧏𐧐𐧑𐧒𐧓𐧔𐧕𐧖𐧗𐧘𐧙𐧚𐧛𐧜𐧝𐧞𐧟𐧠𐧡𐧢𐧣𐧤𐧥𐧦𐧧𐧨𐧩𐧪𐧫𐧬𐧭𐧮𐧯𐧰𐧱𐧲𐧳𐧴𐧵𐧶𐧷𐧸𐧹𐧺𐧻𐧼𐧽𐧾𐧿𐨀𐨁𐨂𐨃𐨄𐨅𐨆𐨇𐨈𐨉𐨊𐨋𐨌𐨍𐨎𐨏𐨐𐨑𐨒𐨓𐨔𐨕𐨖𐨗𐨘𐨙𐨚𐨛𐨜𐨝𐨞𐨟𐨠𐨡𐨢𐨣𐨤𐨥𐨦𐨧𐨨𐨩𐨪𐨫𐨬𐨭𐨮𐨯𐨰𐨱𐨲𐨳𐨴𐨵𐨶𐨷𐨹𐨺𐨸𐨻𐨼𐨽𐨾𐨿𐩀𐩁𐩂𐩃𐩄𐩅𐩆𐩇𐩈𐩉𐩊𐩋𐩌𐩍𐩎𐩏𐩐𐩑𐩒𐩓𐩔𐩕𐩖𐩗𐩘𐩙𐩚𐩛𐩜𐩝𐩞𐩟𐩠𐩡𐩢𐩣𐩤𐩥𐩦𐩧𐩨𐩩𐩪𐩫𐩬𐩭𐩮𐩯𐩰𐩱𐩲𐩳𐩴𐩵𐩶𐩷𐩸𐩹𐩺𐩻𐩼𐩽𐩾𐩿𐪀𐪁𐪂𐪃𐪄𐪅𐪆𐪇𐪈𐪉𐪊𐪋𐪌𐪍𐪎𐪏𐪐𐪑𐪒𐪓𐪔𐪕𐪖𐪗𐪘𐪙𐪚𐪛𐪜𐪝𐪞𐪟𐪠𐪡𐪢𐪣𐪤𐪥𐪦𐪧𐪨𐪩𐪪𐪫𐪬𐪭𐪮𐪯𐪰𐪱𐪲𐪳𐪴𐪵𐪶𐪷𐪸𐪹𐪺𐪻𐪼𐪽𐪾𐪿𐫀𐫁𐫂𐫃𐫄𐫅𐫆𐫇𐫈𐫉𐫊𐫋𐫌𐫍𐫎𐫏𐫐𐫑𐫒𐫓𐫔𐫕𐫖𐫗𐫘𐫙𐫚𐫛𐫜𐫝𐫞𐫟𐫠𐫡𐫢𐫣𐫤𐫦𐫥𐫧𐫨𐫩𐫪𐫫𐫬𐫭𐫮𐫯𐫰𐫱𐫲𐫳𐫴𐫵𐫶𐫷𐫸𐫹𐫺𐫻𐫼𐫽𐫾𐫿𐬀𐬁𐬂𐬃𐬄𐬅𐬆𐬇𐬈𐬉𐬊𐬋𐬌𐬍𐬎𐬏𐬐𐬑𐬒𐬓𐬔𐬕𐬖𐬗𐬘𐬙𐬚𐬛𐬜𐬝𐬞𐬟𐬠𐬡𐬢𐬣𐬤𐬥𐬦𐬧𐬨𐬩𐬪𐬫𐬬𐬭𐬮𐬯𐬰𐬱𐬲𐬳𐬴𐬵𐬶𐬷𐬸𐬹𐬺𐬻𐬼𐬽𐬾𐬿𐭀𐭁𐭂𐭃𐭄𐭅𐭆𐭇𐭈𐭉𐭊𐭋𐭌𐭍𐭎𐭏𐭐𐭑𐭒𐭓𐭔𐭕𐭖𐭗𐭘𐭙𐭚𐭛𐭜𐭝𐭞𐭟𐭠𐭡𐭢𐭣𐭤𐭥𐭦𐭧𐭨𐭩𐭪𐭫𐭬𐭭𐭮𐭯𐭰𐭱𐭲𐭳𐭴𐭵𐭶𐭷𐭸𐭹𐭺𐭻𐭼𐭽𐭾𐭿𐮀𐮁𐮂𐮃𐮄𐮅𐮆𐮇𐮈𐮉𐮊𐮋𐮌𐮍𐮎𐮏𐮐𐮑𐮒𐮓𐮔𐮕𐮖𐮗𐮘𐮙𐮚𐮛𐮜𐮝𐮞𐮟𐮠𐮡𐮢𐮣𐮤𐮥𐮦𐮧𐮨𐮩𐮪𐮫𐮬𐮭𐮮𐮯𐮰𐮱𐮲𐮳𐮴𐮵𐮶𐮷𐮸𐮹𐮺𐮻𐮼𐮽𐮾𐮿𐯀𐯁𐯂𐯃𐯄𐯅𐯆𐯇𐯈𐯉𐯊𐯋𐯌𐯍𐯎𐯏𐯐𐯑𐯒𐯓𐯔𐯕𐯖𐯗𐯘𐯙𐯚𐯛𐯜𐯝𐯞𐯟𐯠𐯡𐯢𐯣𐯤𐯥𐯦𐯧𐯨𐯩𐯪𐯫𐯬𐯭𐯮𐯯𐯰𐯱𐯲𐯳𐯴𐯵𐯶𐯷𐯸𐯹𐯺𐯻𐯼𐯽𐯾𐯿𐰀𐰁𐰂𐰃𐰄𐰅𐰆𐰇𐰈𐰉𐰊𐰋𐰌𐰍𐰎𐰏𐰐𐰑𐰒𐰓𐰔𐰕𐰖𐰗𐰘𐰙𐰚𐰛𐰜𐰝𐰞𐰟𐰠𐰡𐰢𐰣𐰤𐰥𐰦𐰧𐰨𐰩𐰪𐰫𐰬𐰭𐰮𐰯𐰰𐰱𐰲𐰳𐰴𐰵𐰶𐰷𐰸𐰹𐰺𐰻𐰼𐰽𐰾𐰿𐱀𐱁𐱂𐱃𐱄𐱅𐱆𐱇𐱈𐱉𐱊𐱋𐱌𐱍𐱎𐱏𐱐𐱑𐱒𐱓𐱔𐱕𐱖𐱗𐱘𐱙𐱚𐱛𐱜𐱝𐱞𐱟𐱠𐱡𐱢𐱣𐱤𐱥𐱦𐱧𐱨𐱩𐱪𐱫𐱬𐱭𐱮𐱯𐱰𐱱𐱲𐱳𐱴𐱵𐱶𐱷𐱸𐱹𐱺𐱻𐱼𐱽𐱾𐱿𐲀𐲁𐲂𐲃𐲄𐲅𐲆𐲇𐲈𐲉𐲊𐲋𐲌𐲍𐲎𐲏𐲐𐲑𐲒𐲓𐲔𐲕𐲖𐲗𐲘𐲙𐲚𐲛𐲜𐲝𐲞𐲟𐲠𐲡𐲢𐲣𐲤𐲥𐲦𐲧𐲨𐲩𐲪𐲫𐲬𐲭𐲮𐲯𐲰𐲱𐲲𐲳𐲴𐲵𐲶𐲷𐲸𐲹𐲺𐲻𐲼𐲽𐲾𐲿𐳀𐳁𐳂𐳃𐳄𐳅𐳆𐳇𐳈𐳉𐳊𐳋𐳌𐳍𐳎𐳏𐳐𐳑𐳒𐳓𐳔𐳕𐳖𐳗𐳘𐳙𐳚𐳛𐳜𐳝𐳞𐳟𐳠𐳡𐳢𐳣𐳤𐳥𐳦𐳧𐳨𐳩𐳪𐳫𐳬𐳭𐳮𐳯𐳰𐳱𐳲𐳳𐳴𐳵𐳶𐳷𐳸

# QUIZ

GZZ

```
Leonie@Laptop $ hexdump -C datei.txt
00000000  41 6d 20 45 6e 64 65 20 69 73 74 20 65 73 20 61 |Am Ende ist es a|
00000010  62 65 72 20 45 67 61 6c 20 6f 62 20 64 69 65 20 |ber Egal ob die |
00000020  4c 65 75 74 65 20 22 73 69 6e 6e 20 6d 61 63 68 |Leute "sinn mach|
00000030  65 6e 22 20 6f 64 65 72 20 22 73 69 6e 6e 20 65 |en" oder "sinn e|
00000040  72 67 65 62 65 6e 22 20 73 61 67 65 6e 2c 20 64 |rgeben" sagen, d|
00000050  65 6e 6e 20 73 69 65 20 72 65 64 65 6e 20 6d 65 |enn sie reden me|
00000060  69 73 74 65 6e 73 20 73 6f 77 69 65 73 6f 20 6e |istens sowieso n|
00000070  75 72 20 55 6e 73 69 6e 6e 21 0a |ur Unsinn!.|
0000007b
```

■ Welches Encoding hat die Datei?

- a) ISO-Latin
- b) UTF-16LE
- c) UTF-16BE



# QUIZ

GZZ

```
Leonie@Laptop $ hexdump -C datei.txt
00000000  41 6d 20 45 6e 64 65 20  69 73 74 20 65 73 20 61  |Am Ende ist es a|
00000010  62 65 72 20 45 67 61 6c  20 6f 62 20 64 69 65 20  |ber Egal ob die |
00000020  4c 65 75 74 65 20 22 73  69 6e 6e 20 6d 61 63 68  |Leute "sinn mach|
00000030  65 6e 22 20 6f 64 65 72  20 22 73 69 6e 6e 20 65  |en" oder "sinn e|
00000040  72 67 65 62 65 6e 22 20  73 61 67 65 6e 2c 20 64  |rgeben" sagen, d|
00000050  65 6e 6e 20 73 69 65 20  72 65 64 65 6e 20 6d 65  |enn sie reden me|
00000060  69 73 74 65 6e 73 20 73  6f 77 69 65 73 6f 20 6e  |istens sowieso n|
00000070  75 72 20 55 6e 73 69 6e  6e 21 0a                    |ur Unsinn!.|
0000007b
```

■ Welches Encoding hat die Datei?

- a) ISO-Latin
- b) UTF-16LE
- c) UTF-16BE



# WILDCARDS

- Mit einer sog. Wildcard kann man Dateien nach ihrem Namen auswählen
- `ls`  
`abc.txt`  
`defgh.txt`  
`xyz.py`
- `ls *.txt`  
`abc.txt`  
`defgh.txt`
- `ls ???.*`  
`abc.txt`  
`xyz.txt`



- In Python kann man Text ähnlich wie mit Wildcards durchsuchen
- RegExes (RegularExpression) können noch mehr als Wildcards
- Dies sind die möglichen Zeichen
  - . (Punkt) = ein beliebiges Zeichen
  - a = ein kleines a
  - [a-z] = ein kleiner Buchstabe
  - [A-Z] = ein großer Buchstabe
  - [äöüÄÖÜ] = ein Umlaut
  - [^z] = ein beliebiges Zeichen das kein kleines z ist



# QUIZ

GZZ

- Was matcht der folgende Regex?
- '.o.....'
- a) Goethe
- b) Schiller
- c) Kant
- d) Lipschitz



# QUIZ

GZZ

- Was matcht der folgende Regex?
- `' .o..... '`
- a) Goethe
- b) Schiller
- c) Kant
- d) Lipschitz



# QUIZ

GZZ

■ Was matcht der folgende Regex?

■ `'..[m-z][^n]'`

- a) Seit
- b) Sein
- c) Kann
- d) Muss



# QUIZ

GZZ

■ Was matcht der folgende Regex?

■ `'..[m-z][^n]'`

- a) Seit
- b) Sein
- c) Kann
- d) Muss



# QUIZ

GZZ

- Was matcht der folgende Regex?
- '[A-Z][äöü]st'
- a) Übst
- b) Obst
- c) Löst
- d) Dust



# QUIZ

GZZ

- Was matcht der folgende Regex?
- '[A-Z][äöü]st'
- a) Übst
- b) Obst
- c) Löst
- d) Dust



- Man kann für Zeichen auch eine Anzahl festlegen

- \* = beliebig viele (0-∞)
- ? = eins oder keins (0-1)
- + = ein oder mehr (1-∞)
- {1, 3} = eins bis drei (1-3)
- {5, } = fünf bis beliebig (5-∞)
- {, 2} = keine bis zwei (0-2)



# QUIZ

GZZ

- Welche der folgenden Strings werden von diesem Regex gematcht: „test“, „tttt“, „eeee“, „a“
- `'[a-z]{2,4}'`
- a) „a“
- b) „tttt“, „eeee“
- c) „test“
- d) „test“, „tttt“, „eeee“



# QUIZ

GZZ

- Welche der folgenden Strings werden von diesem Regex gematcht: „test“, „tttt“, „eeee“, „a“
- `'[a-z]{2,4}'`
- a) „a“
- b) „tttt“, „eeee“
- c) „test“
- d) „test“, „tttt“, „eeee“



# QUIZ

GZZ

- Was matcht der folgende Regex?
- `'.{1,3}'`
- a) Das
- b) Dies
- c) Sonst
- d) Selbst



# QUIZ

GZZ

- Was matcht der folgende Regex?
- `'.{1,3}'`
- a) Das
- b) Dies
- c) Sonst
- d) Selbst



# QUIZ

GZZ

■ Was matcht der folgende Regex?

■ 'a+.\*'

a) alle

b) Aal

c) Egaaal

d) Alles



# QUIZ

GZZ

■ Was matcht der folgende Regex?

■ 'a+.\*'

a) alle

b) Aal

c) Egaaal

d) Alles



# QUIZ

GZZ

- Was matcht der folgende Regex?
- 'x?B+.\*'
- a) Xenial
- b) Bauen
- c) Xylophon
- d) basteln



# QUIZ

GZZ

■ Was matcht der folgende Regex?

■ 'x?B+.\*'

a) Xenial

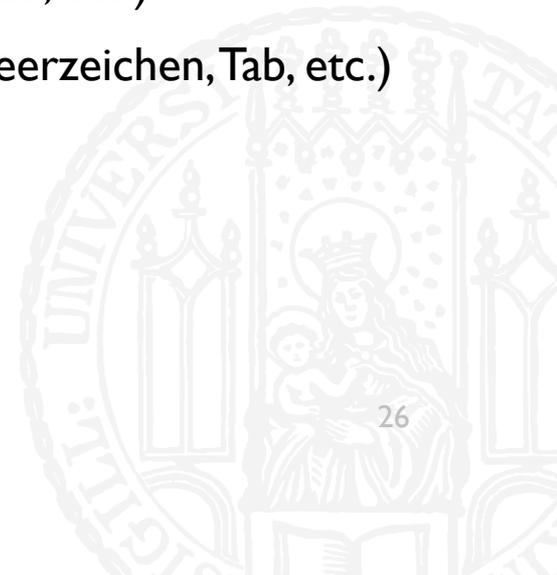
b) Bauen

c) Xylophon

d) basteln



- Es gibt besondere Zeichen für bestimmte Zeichen
  - `^` = Anfang des Strings
  - `$` = Ende des Strings
  - `\w` = ein „Word Character“ (Buchstabe)
  - `\s` = ein „Whitespace Character“ (Leerzeichen, Tab, etc.)
  - `\S` = kein „Whitespace Character“ (alles außer Leerzeichen, Tab, etc.)
  - `\d` = ein „digit“ (Zahl)
  - `\\` = ein tatsächliches „\“
  - `\.` = ein tatsächlicher „.“



# QUIZ

GZZ

- Was matcht der folgende Regex in diesem Text?
  - `'^\w+'`
  - "Wir verkaufen nicht nur Obst!  
Auch Döner!"
- a) „Wir“
  - b) „Döner!“
  - c) „Wir“ „verkaufen“ „nicht“ „nur“ „Auch“ „Döner“
  - d) „Wir“ „verkaufen“ „nicht“ „nur“ Obst“ „Auch“ Döner“



# QUIZ

GZZ

- Was matcht der folgende Regex in diesem Text?
  - `'^\w+'`
  - "Wir verkaufen nicht nur Obst!  
Auch Döner!"
- a) „Wir“
  - b) „Döner!“
  - c) „Wir“ „verkaufen“ „nicht“ „nur“ „Auch“ „Döner“
  - d) „Wir“ „verkaufen“ „nicht“ „nur“ Obst“ „Auch“ Döner“



# QUIZ

GZZ

- Was matcht der folgende Regex in diesem Text?
  - `'\w+@\w+\.\w{2,3}'`
  - “Ihr könnt Fragen zum Tutorium, Aufgaben, Python, RegExs etc. jederzeit an [leonie.weissweiler@campus.lmu.de](mailto:leonie.weissweiler@campus.lmu.de), oder an [kaiser@gmail.de](mailto:kaiser@gmail.de) (insbesondere Fragen zur Korrektur der Hausaufgaben) stellen.“
- a) nichts
  - b) „leonie.weissweiler@campus.lmu.de“
  - c) „kaiser@gmail.de“
  - d) „kaiser@gmail.de“ „leonie.weissweiler@campus.lmu.de“



# QUIZ

GZZ

- Was matcht der folgende Regex in diesem Text?
  - `'\w+@\w+\.\w{2,3}'`
  - “Ihr könnt Fragen zum Tutorium, Aufgaben, Python, RegExs etc. jederzeit an [leonie.weissweiler@campus.lmu.de](mailto:leonie.weissweiler@campus.lmu.de), oder an [kaiser@gmail.de](mailto:kaiser@gmail.de) (insbesondere Fragen zur Korrektur der Hausaufgaben) stellen.“
- a) nichts
  - b) „leonie.weissweiler@campus.lmu.de“
  - c) „kaiser@gmail.de“
  - d) „kaiser@gmail.de“ „leonie.weissweiler@campus.lmu.de“



- Um in Python eine Regex zu benutzen muss man sie zuerst erzeugen
- `meine_regex = re.compile(r'b+.*')`
- Man schreibt ein `r` vor den String um einen raw-string zu erzeugen, um Backslashes verwenden zu können
- Danach kann man sie mit verschiedenen Funktionen anwenden
  - `re.match(meine_regex, "Ein beliebig blöder String")`  
Überprüft ob die Regex am Anfang des Strings matcht
  - `re.search(meine_regex, "Ein beliebig blöder String")`  
Überprüft ob die Regex irgendwo im String matcht
  - `re.findall(meine_regex, "Ein beliebig blöder String")`  
Gibt eine Liste aller Teile des Strings zurück die matchen (ohne Überlappung)

# REGEX ALS SPLIT

Dies ist ein Text. Er enthält, manche, so 12, besondere Zeichen

```
for word in text.split(' ')
```

```
split_regex = re.compile(r'\w+'),  
for word in re.findall(split_regex, text)
```

Dies ist ein Text. Er enthält,  
manche, so 12, besondere  
Zeichen

Dies ist ein Text. Er enthält,  
manche, so 12, besondere  
Zeichen

# ERGÄNZUNG: REVERSE SORT

GZZ

- Man kann beim Sortieren einen „Rückwärtsgang“ einlegen:

- #nach Keys sortieren

```
for key, value in sorted(dict.items(), reverse=True):  
    print(key, value)
```

- #nach Value sortieren

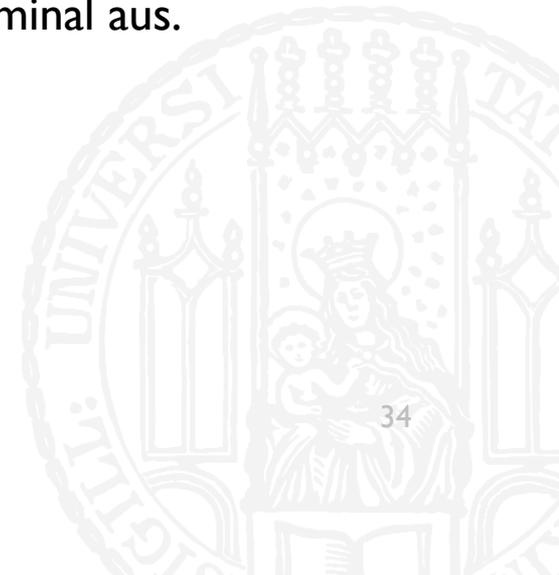
```
for key, value in sorted(dict.items(), key=lambda x: x[1], reverse=True):  
    print(key, value)
```



# MUSTERLÖSUNG 8-I

GZZ

Sortieren Sie die erzeugte Frequenzliste nach der Häufigkeit und geben Sie die nach der aufsteigenden Häufigkeit sortierte Frequenzliste auf dem Terminal aus.



# MUSTERLÖSUNG 8-I

GZZ

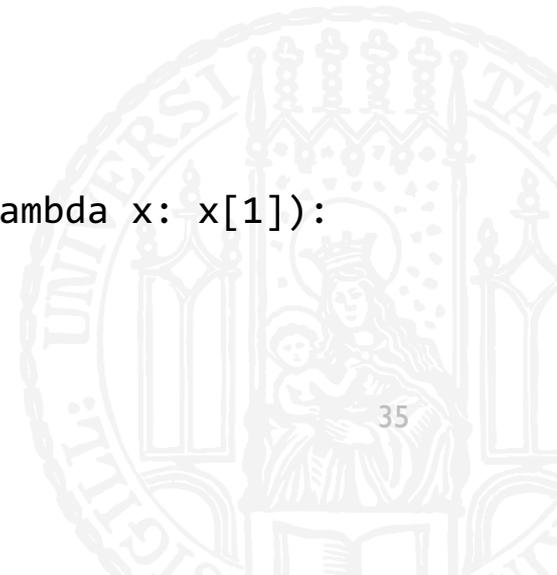
```
#!/usr/bin/python3
#Aufgabe 8-1
#Autorin: Leonie Weißweiler

verbrechen = open('Verbrechen_Strafe.txt', 'r')
frequenzliste = {}

    for word in line.split(' '):
        word = word.strip()
        word = word.lower()
        if (word in frequenzliste):
            frequenzliste[word] = frequenzliste[word] + 1
        else:
            frequenzliste[word] = 1

for wort, frequenz in sorted(frequenzliste.items(), key=lambda x: x[1]):
    print(wort, ':', frequenz)

verbrechen.close()
```



# MUSTERLÖSUNG 8-2

GZZ

Sortieren Sie die erzeugte Frequenzliste nach der Häufigkeit und geben Sie die nach der absteigenden Häufigkeit sortierte Frequenzliste auf dem Terminal aus.



# MUSTERLÖSUNG 8-2

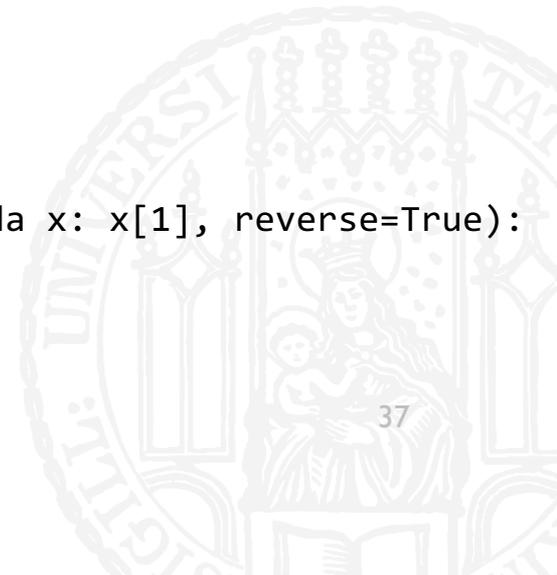
GZZ

```
#!/usr/bin/python3
#Aufgabe 8-2
#Autorin: Leonie Weißweiler

verbrechen = open('Verbrechen_Strafe.txt', 'r')
frequenzliste = {}

for line in verbrechen:
    for word in line.split(' '):
        word = word.strip()
        word = word.lower()
        if (word in frequenzliste):
            frequenzliste[word] = frequenzliste[word] + 1
        else:
            frequenzliste[word] = 1
for wort, frequenz in sorted(frequenzliste.items(), key=lambda x: x[1], reverse=True):
    print(wort, ':', frequenz)

verbrechen.close()
```



# MUSTERLÖSUNG 8-3

GZZ

Geben Sie die zehn Wörter aus, die am häufigsten vorkommen.



# MUSTERLÖSUNG 8-3

GZZ

```
#!/usr/bin/python3
#Aufgabe 8-3
#Autorin: Leonie Weißweiler

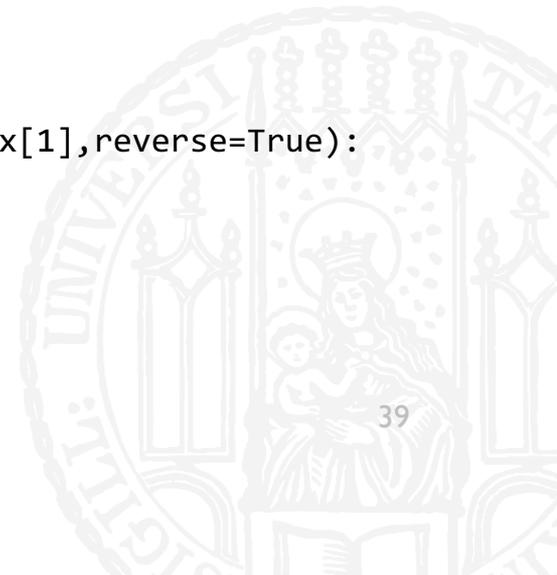
verbrechen = open('Verbrechen_Strafe.txt', 'r')
frequenzliste = {}
allewoerter=[]

for line in verbrechen:
    for word in line.split(' '):
        word = word.strip()
        word = word.lower()
        if (word in frequenzliste):
            frequenzliste[word] = frequenzliste[word] + 1
        else:
            frequenzliste[word] = 1

for wort, frequenz in sorted(frequenzliste.items(), key=lambda x: x[1],reverse=True):
    allewoerter.append(wort)

for x in range(0,10):
    print (allewoerter[x])

verbrechen.close()
```



# MUSTERLÖSUNG 8-4

Verwenden Sie für die nächsten Aufgaben eine Textdatei, die sie aus einer .html Datei extrahieren. Dazu folgende Hinweise:

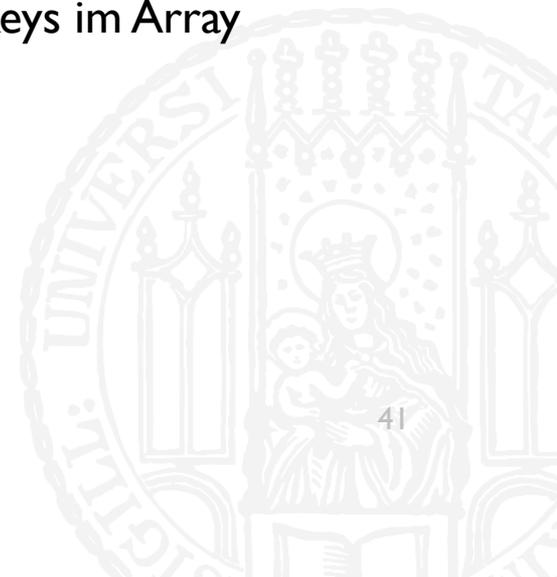
- a) Mit dem UNIX Befehl `wget` : `wget "http://de.wikipedia.org/wiki/Ludwig\_Wittgenstein" -O wittgenstein.html` speichern Sie von der Kommandozeile aus automatisch den Inhalt einer www-Seite in einer lokalen Datei. Mit dem obigen Befehl speichern Sie z.B. den Inhalt der Wikipedia Seite über Ludwig Wittgenstein in der Datei `wittgenstein.html`.
- b) Mit dem UNIX Befehl `lynx` können Sie den Text aus der html Datei extrahieren und in einer Textdatei speichern.

```
lynx -dump -assume\_charset=UTF-8 -hiddenlinks=ignore -nolist -  
verbose wittgenstein.html > wittgenstein.txt
```

# MUSTERLÖSUNG 8-5

GZZ

Spalten Sie alle Zeilen der Datei wittgenstein.txt in Wörter auf und erzeugen Sie eine Frequenzliste der Wörter. Sortieren Sie die Frequenzliste absteigend nach der Häufigkeit und speichern Sie die nach der Häufigkeit sortierten keys im Array `alle_sortierten_woerter`



# MUSTERLÖSUNG 8-5

GZZ

```
#!/usr/bin/python3
#Aufgabe 8-5
#Autorin: Leonie Weißweiler

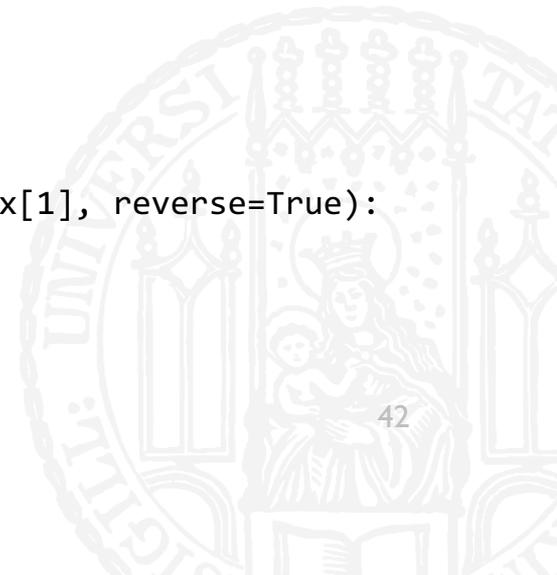
import re

splitregex = re.compile(r'\w+')
wittgenstein = open('wittgenstein.txt', 'r')
frequenzliste = {}
alle_sortierten_woerter = []

for line in wittgenstein:
    for word in re.findall(splitregex,line):
        word = word.strip()
        if (word in frequenzliste):
            frequenzliste[word] = frequenzliste[word] + 1
        else:
            frequenzliste[word] = 1

for wort, frequenz in sorted(frequenzliste.items(), key=lambda x: x[1], reverse=True):
    alle_sortierten_woerter.append(wort)

print(alle_sortierten_woerter)
wittgenstein.close()
```



# MUSTERLÖSUNG 8-6

GZZ

- Erweitern Sie das Programm, sodass es alle großgeschriebenen Wörter im Array `alle_sortierten_woerter` findet und die Wörter zusammen mit Ihrer Häufigkeit in der Datei "grosse\_woerter.txt" speichert.



# MUSTERLÖSUNG 8-6

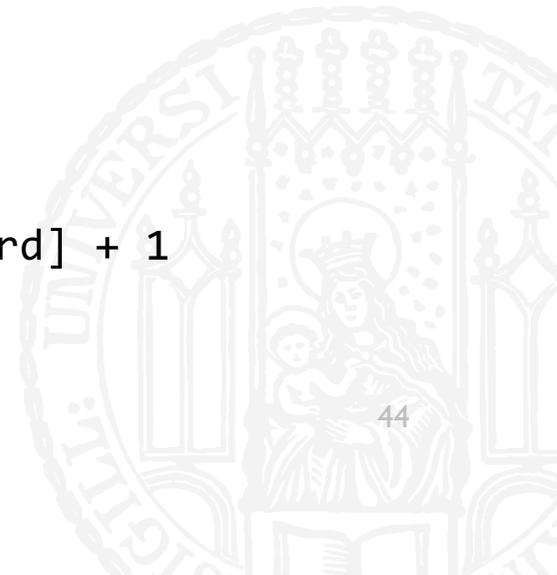
GZZ

```
#!/usr/bin/python3
#Aufgabe 8-6
#Autorin: Leonie Weißweiler

import re

splitregex = re.compile(r'\w+')
grosseregex = re.compile(r'^[A-ZÄÖÜ]')
wittgenstein = open('wittgenstein.txt', 'r')
grossewoerter = open('grosse_woerter.txt', 'w')
frequenzliste = {}
alle_sortierten_woerter = []

for line in wittgenstein:
    for word in re.findall(splitregex,line):
        word = word.strip()
        if (word in frequenzliste):
            frequenzliste[word] = frequenzliste[word] + 1
        else:
            frequenzliste[word] = 1
```



# MUSTERLÖSUNG 8-6

GZZ

```
for wort, frequenz in sorted(frequenzliste.items(), key=lambda
x: x[1], reverse=True):
    alle_sortierten_woerter.append(wort)

for wort in alle_sortierten_woerter:
    if(re.search(grosseregex, wort)):
        grossewoerter.write(wort)
        grossewoerter.write(' : ')
        grossewoerter.write(str(frequenzliste[word]))
        grossewoerter.write('\n')

wittgenstein.close()
grossewoerter.close()
```



# MUSTERLÖSUNG 8-7A

GZZ

Wie lauten die regulären Ausdrücke, die folgende Wörter im Array `alle_sortierten_woerter` finden (geben Sie nur den regulären Ausdruck an!). Testen Sie Ihre Ausdrücke mit einem Programm.

a) alle Wörter mit 3 Buchstaben



# MUSTERLÖSUNG 8-7A

GZZ

```
#!/usr/bin/python3
#Aufgabe 8-7a
#Autorin: Leonie Weißweiler

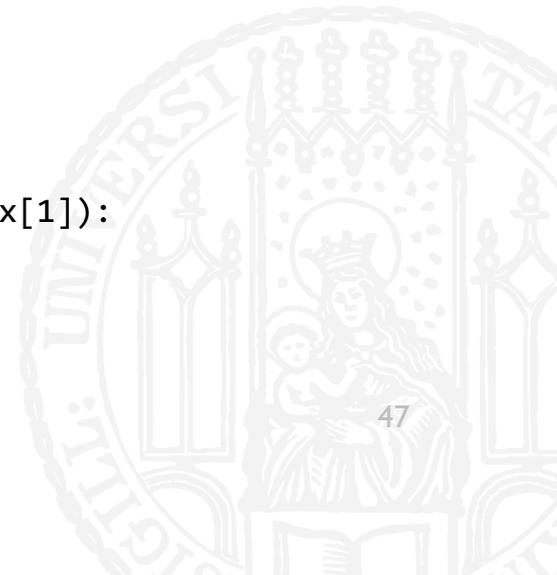
import re

splitregex = re.compile(r'\w+')
kurzregex = re.compile(r'^\w{1,3}$')
wittgenstein = open('wittgenstein.txt', 'r')
frequenzliste = {}

for line in wittgenstein:
    for word in re.findall(splitregex,line):
        word = word.strip()
        if (word in frequenzliste):
            frequenzliste[word] = frequenzliste[word] + 1
        else:
            frequenzliste[word] = 1

for wort, frequenz in sorted(frequenzliste.items(), key=lambda x: x[1]):
    if(re.search(kurzregex, wort)):
        print(wort)

wittgenstein.close()
```



# MUSTERLÖSUNG 8-7A

GZZ

$r'^{\wedge}\{1,3\}$



# MUSTERLÖSUNG 8-7B

GZZ

b) alle Wörter in denen ein Vokal vorkommt

r' [aeiouAEIOU]+'



# MUSTERLÖSUNG 8-7C

GZZ

c) alle Wörter mit Umlauten

r' [ ä ö ü Ä Ö Ü ] + '



# MUSTERLÖSUNG 8-7D

GZZ

d) alle Wörter die mit 'net' enden

$r' \cdot \text{*net\$}$



# MUSTERLÖSUNG 8-7E

GZZ

e) alle Wörter die drei bis fünf Buchstaben lang sind und mit einem Vokal beginnen

$r'^{\wedge}[aeiouAEIOU]\{2,4\}\$'$



# MUSTERLÖSUNG 8-7F

GZZ

f) alle Wörter die mit einem Großbuchstaben beginnen und mit dem Suffix 'ung' enden

$r'^{^}[A-ZÄÖÜ] \cdot *ung\$'$

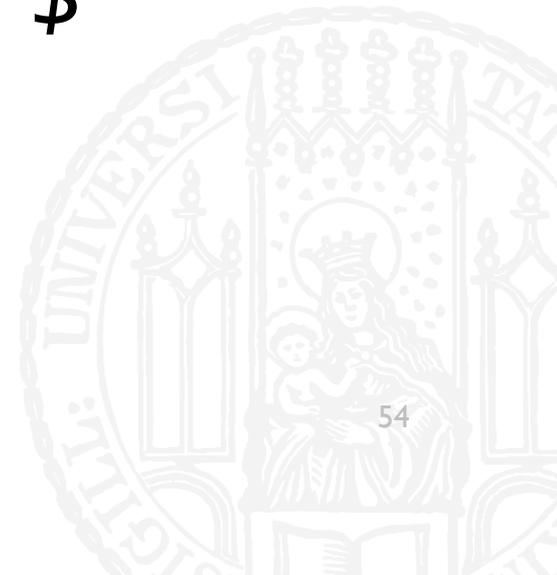


# MUSTERLÖSUNG 8-7H

GZZ

h) alle Wörtern in denen kein Vokal vorkommt

$r'^{\wedge}[\wedge aeiou\ddot{u}AEIOU\ddot{A}\ddot{O}\ddot{U}] * \$ '$



# MUSTERLÖSUNG 8-8

GZZ

Schreiben Sie ein Programm, das die Anzahl aller kleingeschriebenen Wörter in der Datei wittgenstein.txt ermittelt. Geben Sie die Anzahl aus.



# MUSTERLÖSUNG 8-8

GZZ

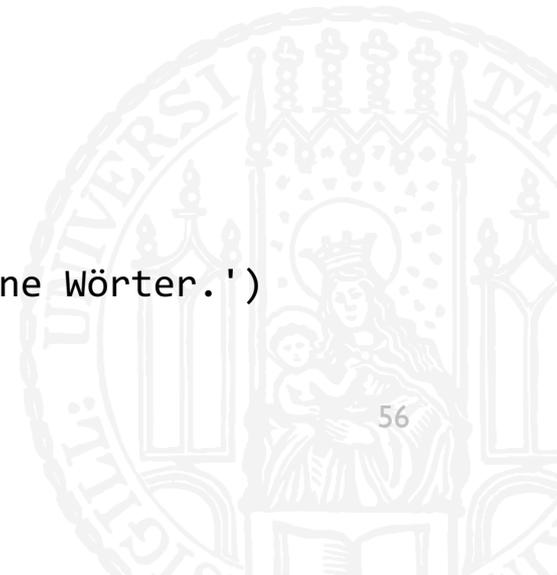
```
#!/usr/bin/python3
#Aufgabe 8-8
#Autorin: Leonie Weißweiler

import re

splitregex = re.compile(r'\w+')
lowerregex = re.compile(r'^[a-zäöü]+$')
wittgenstein = open('wittgenstein.txt', 'r')
anzahllower = 0

for line in wittgenstein:
    for word in re.findall(splitregex,line):
        word = word.strip()
        if(re.search(lowerregex, word)):
            anzahllower = anzahllower + 1

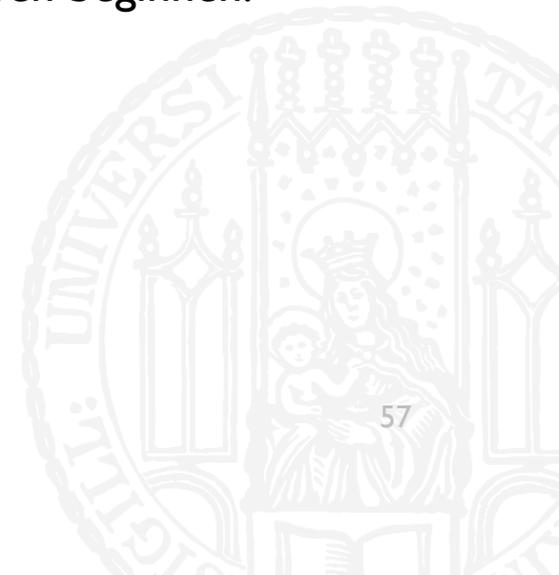
print('In der Datei waren', anzahllower, 'kleingeschriebene Wörter.')
wittgenstein.close()
```



# MUSTERLÖSUNG 8-9

GZZ

Geben Sie alle Zeilen der Datei aus, die mit einem Großbuchstaben beginnen.



# MUSTERLÖSUNG 8-9

GZZ

```
#!/usr/bin/python3
#Aufgabe 8-9
#Autorin: Leonie Weißweiler

import re

großregex = re.compile(r'^[A-ZÄÖÜ]+')
wittgenstein = open('wittgenstein.txt', 'r')

for line in wittgenstein:
    line = line.strip()
    if(re.search(großregex, line)):
        print(line)

wittgenstein.close()
```



# MUSTERLÖSUNG 8-10

GZZ

Geben Sie alle Zeilen der Datei aus, in denen "der", "die" oder "das" vorkommt.



# MUSTERLÖSUNG 8-10

GZZ

```
#!/usr/bin/python3
#Aufgabe 8-10
#Autorin: Leonie Weißweiler

import re

artikelregex = re.compile(r'\b(der|die|das|Der|Die|Das)\b')
wittgenstein = open('wittgenstein.txt', 'r')

for line in wittgenstein:
    line = line.strip()
    if(re.search(artikelregex, line)):
        print(line)

wittgenstein.close()
```



# MUSTERLÖSUNG 8-11

GZZ

Geben Sie alle Wörter der Datei aus, die von einem Satzzeichen gefolgt sind.



# MUSTERLÖSUNG 8-11

GZZ

```
#!/usr/bin/python3
#Aufgabe 8-11
#Autorin: Leonie Weißweiler

import re

splitregex = re.compile(r'\w+[?!.,;:-]')
wittgenstein = open('wittgenstein.txt', 'r')
frequenzliste = {}

for line in wittgenstein:
    for word in re.findall(splitregex,line):
        word = word.strip()
        if (word in frequenzliste):
            frequenzliste[word] = frequenzliste[word] + 1
        else:
            frequenzliste[word] = 1

for key in frequenzliste.keys():
    print(key)

wittgenstein.close()
```



# MUSTERLÖSUNG 8-12

GZZ

Schreiben Sie ein Programm, das ein Wort einliest und nachsieht, ob das von rückwärts gelesene Wortort als Teil eines Wortes in einer großen Datei vorkommt.

[http://www.cis.uni-muenchen.de/kurse/max/einfprog/texte/sz\\\_I\\_Mio.txt.gz](http://www.cis.uni-muenchen.de/kurse/max/einfprog/texte/sz\_I_Mio.txt.gz)

Beispiel: "LOVE" => "EVOL" => Revolution => "R.EVOLution"



# MUSTERLÖSUNG 8-12

GZZ

```
#!/usr/bin/python3
#Aufgabe 8-12
#Autorin: Leonie Weißweiler

import re

splitregex = re.compile(r'\w+')
sz = open('sz_1Mio.txt', 'r')
benutzerwort = input('Geben Sie ein Wort ein.\n')
wortrückwärts = benutzerwort[::-1]

for line in sz:
    for word in re.findall(splitregex,line):
        word = word.strip()
        if(wortrückwärts.lower() in word.lower()):
            print(word)

sz.close()
```

