پیاده سازی کد هافمن در پایتون

در خط 2 ام ماژولی به نام heapq را در برنامه ایمپورت میکنیم که این ماژول نود های(اینجا منظور از نود ها همان حروف هستند) ما را گرفته و آنها را وارد یک هیپ میکند و بعد از مرتب کردن این نود ها بر اساس تعداد تکرارشان آنها را به صورت صعودی مرتب میکند.

در خط 3 ام کلاسی به نام node ایجاد میکنیم که در خط 6 ام تعداد تکرار حروف یا همان نود ها و در خط 8 ام خود نود ها را با نام symbol در خود خواهد داشت به علاوه 3 خط کد 10 و 12 و14 که به ترتیب فرزندان سمت چپ ، فرزندان سمت راست و مقدار صفر و یک بودن(huff) را مشخص میکنند.

حالا در خط 18 ام تابع اصلی خود را با نام printNodes تعریف میکنیم که بعد از گرفتن ورودی(همان کلاس node که قبلا ایجاد کردیم) و قرار دادن آن روی درخت و پیمایش این درخت کد های هافمن هر حرف را ایجاد میکند .   
در خط 20 ام ک متغیر huff خود را به value که از ورودی گرفته اضافه میکند.

در خط 22 ام چک میکند که نود مورد نظر جزو نود های سمت چپ است یا خیر در صورتی که نود چپ باشد یعنی دارای فرزند باشد دوباره تابع اصلی را برای این نود بازخوانی میکند.

در خط 25 همین کار را برای نود های سمت راست انجام میدهد.

در خط 28 اگر نود مورد نظر جزو نود های راست و چپ نباشد (نود بدون فرزند) این نود را به همراه کد هافمن خود چاپ میکند.

در خط 32 یک لیست از کاراکتر ها و در خط 34 فراوانی آنها را تعریف میکنیم.

در خط 36 نیز یک لیست خالی به نام nodes ایجاد میکنیم.

حال در خط 38 این لیست را به کمک ماژول heapq وارد یک هیپ میکنیم.

در خط 41 یک حلقه while تعریف میکنیم که تا زمانی که تعداد node ها بیشتر از 1 باشد آنها را بر اساس فراوانی شان به صورت صعودی مرتب میکند.

در خطوط 46 و 47 مقدار huff هر نود را بر اساس راست یا چپ بودنشان به آنها اضافه میکند.

در خط 49 نیز نود جدیدی ایجاد میکنیم که دو نود قبلی به عنوان فرزندان سمت چپ و راست این نود در نظر گرفته خواهند شد و سپس این نود جدید نیز وارد هیپ میشود(خط 51 ).

این حلقه تا زمانی ادامه پیدا میکند که حداکثر 1 نود باقی بماند.

بالاخره در خط 53 تنها نود باقی مانده در لیست چاپ میشود که با این کار درواقع تمام نود های لیست ما به همراه کد هافمنشان به عنوان خروجی چاپ خواهند شد.

برای مثال ما میخواهیم متنی از حروف را به شکل زیر به عنوان ورودی وارد کنیم:

[‘aaccdbcbbbacddbcbacb’]

[‘a,b,c,d’]

Frequency:[4,7,6,3]

همانطور که میبینید متن ورودی را به حروف تجزیه کردیم و فراوانی هر حرف را بدست آوردیم.

حال با وارد کردن این حروف به علاوه فراوانی شان به عنوان ورودی ، خروجی زیر چاپ خواهد شد:

(به صورت صعودی)

d🡪00

a🡪01

c🡪10

b🡪11