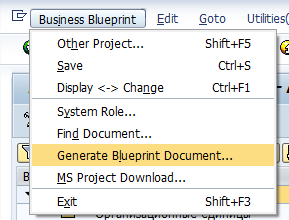
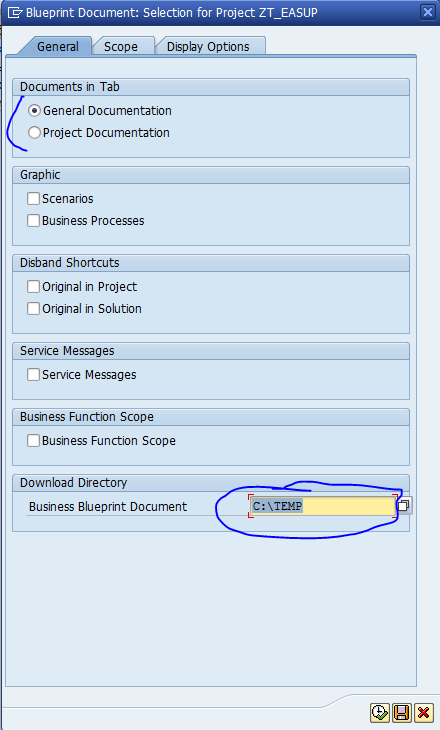
# How to import docs from Solution Manager 7.1 solar01 to Solution Manager 7.2 soldoc. Part 1

Solar01 have method “export blueprint”, but it get you docs in format which you can’t import in soldoc as it is. This way will export only docs with mark “Blueprint-Relevant” 

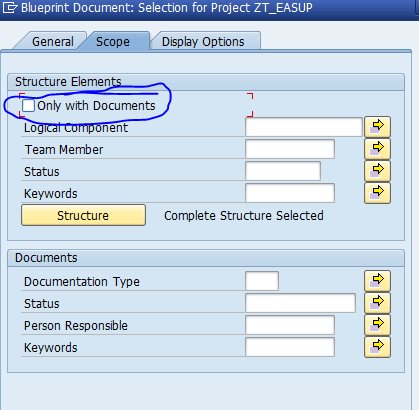
Way to export docs from solar01:

1.

2.choose documents type and select export path



3. deactivate Checkbox “only with documents” – we will need all tree



4. press

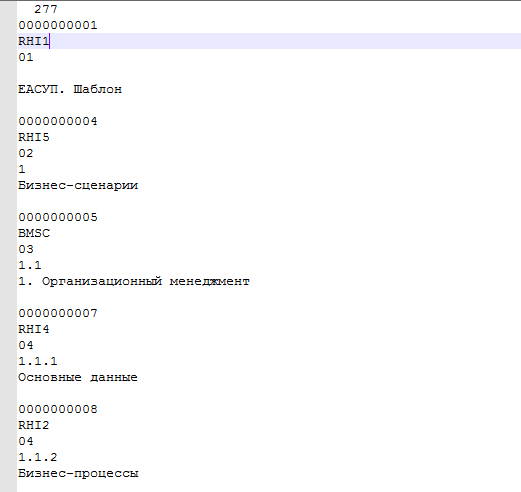
After export in path you have selected you get files from structure with names like “A0D3C106683C1ED8A4AF08124BFC8563.RTF”, file blue.dat, blue.ini, solarblueprint.doc. We will need blue.dat.

solarblueprint.doc have some macros by sap which make specific word file. It don’t give me any help to import and I do not consider it in future.

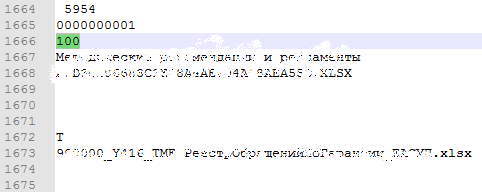
blue.dat – keep all structure solar01. Consider more:

First block of blue.dat – business process tree. It have node id (0000000001, 0000000004 etc),

Node type (RHI\*, BMSC, BMPG, BMPC), node level (01,02,03 etc), node number (1,1.1 etc), node name (NODE\_NAME)



Second block – docs. It begin when you first meet 100 (in my file):



Numbers before blocs – quantity.

Doc have id, type, name, export name, etc.

To parse blue.dat you can use any programming language. I use python 3 (IDE PyCharm Community).

First my script – read\_file. It gets to the entrance export path, which we choose to export from solar01.

We use

import os  
import codecs  
import shutil

get export path and find blue.dat in this folder.

name\_file={}  
id\_file={}  
path = input()  
file = os.path.join(path,"blue.dat")

get all from export path

obj\_file =[sChild for sChild in os.listdir(path)]

decode blue.dat

with codecs.open(file,"r", "utf\_16\_le") as f:  
 file\_lines = [str(line.replace('\n', "")) for line in f]

get intersection docs in file blue.dat and docs in export path

result = list(set(obj\_file) & set(file\_lines))

make dictionary of file names

for n in result:  
 name\_file[n]=file\_lines[file\_lines.index(n)+5].strip()  
 id\_file[n]=file\_lines[file\_lines.index(n)-3]

in my case some files have special character in name which can’t be save in file name in windows os, that’s why i replace it

match = ["\\","/",":","\*","?","\"","<",">","|"]  
for n in result:  
 for i in match:  
 name\_file[n]=name\_file[n].replace(i,"")

create folder where we copy our files with normal name

path\_rename = os.path.join(path,"RENAME\_FILE")  
if not os.path.exists(path\_rename):  
 os.makedirs(path\_rename)

and copy it

for n in result:  
 shutil.copyfile(os.path.join(path,n), os.path.join(path\_rename, str(name\_file[n] + "." + n.split('.')[1])))

Now we will create file with structure for files in folder:

format = '0000000000'  
with open(os.path.join(path\_rename,"solar01\_structure.txt"), 'w', encoding='utf8') as f:  
 f.truncate()  
 for n in result:  
 p=[]  
 path\_solar=""  
 used\_type=[]  
 for i in reversed(range(1,int(id\_file[n])+1)):  
 node = format[:-len(str(i))] + str(i)  
 if node in file\_lines:  
 if int(file\_lines[file\_lines.index(node) + 2]) <= int(file\_lines[file\_lines.index(id\_file[n]) + 2]) and file\_lines[file\_lines.index(node) + 2] not in used\_type:  
  
 p.append(file\_lines[file\_lines.index(node) + 4])  
 used\_type.append(file\_lines[file\_lines.index(node) + 2])  
  
 path\_solar = "\\".join(reversed(p))  
  
 f.writelines(path\_solar + "\\" + name\_file[n] + "." + n.split('.')[1] + "\n")

id file = id node where it was in solar01, we make format = '0000000000', and create file solar01\_structure.txt in RENAME\_FILE folder (we open it for write, it erase it automatically it’s not necessary to use f.truncate())

to make structure like “NODE\_NAME \Бизнес-сценарии\DOC\_NAME.DOCX”

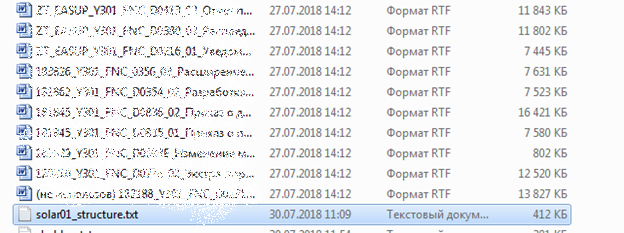
I am take file id and walking through business process tree from file id to 1. But if i just have all node with id < id file I get more excess what’s why I make list used\_type which store type node and if node in range( id file … 1 ) we have used before we skip it.

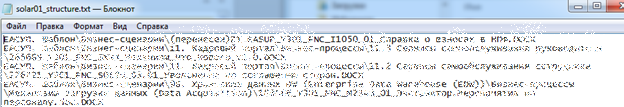
After that we get list p withall nodes for file. We made string of it and write it to file/ and it for all files.

Write “pass” at the end

print("pass")

now we get a folder RENAME\_FILE with normal named files and solar01\_structure.txt





## Full code:

import os  
import codecs  
import shutil  
import re  
  
name\_file={}  
id\_file={}  
path = input()  
file = os.path.join(path,"blue.dat")   
  
obj\_file =[sChild for sChild in os.listdir(path)]  
  
with codecs.open(file,"r", "utf\_16\_le") as f:  
 file\_lines = [str(line.replace('\n', "")) for line in f]  
  
result = list(set(obj\_file) & set(file\_lines))  
  
for n in result:  
 name\_file[n]=file\_lines[file\_lines.index(n)+5].strip()  
 id\_file[n]=file\_lines[file\_lines.index(n)-3]  
  
match = ["\\","/",":","\*","?","\"","<",">","|"]  
for n in result:  
 for i in match:  
 name\_file[n]=name\_file[n].replace(i,"")

path\_rename = os.path.join(path,"RENAME\_FILE")  
if not os.path.exists(path\_rename):  
 os.makedirs(path\_rename)  
  
for n in result:  
 shutil.copyfile(os.path.join(path,n), os.path.join(path\_rename, str(name\_file[n] + "." + n.split('.')[1])))  
  
format = '0000000000'  
with open(os.path.join(path\_rename,"solar01\_structure.txt"), 'w', encoding='utf8') as f:  
 f.truncate()  
 for n in result:  
 p=[]  
 path\_solar=""  
 used\_type=[]  
  
 for i in reversed(range(1,int(id\_file[n])+1)):  
 node = format[:-len(str(i))] + str(i)  
 if node in file\_lines:  
 if int(file\_lines[file\_lines.index(node) + 2]) <= int(file\_lines[file\_lines.index(id\_file[n]) + 2]) and file\_lines[file\_lines.index(node) + 2] not in used\_type:  
   
 p.append(file\_lines[file\_lines.index(node) + 4])  
 used\_type.append(file\_lines[file\_lines.index(node) + 2])  
 path\_solar = "\\".join(reversed(p))  
 f.writelines(path\_solar + "\\" + name\_file[n] + "." + n.split('.')[1] + "\n")  
  
print("pass")

# How to import docs from Solution Manager 7.1 solar01 to Solution Manager 7.2 soldoc. Part 2

Next we want to make SolMan template to load structure.

import os  
import codecs  
path = input()  
path\_rename = os.path.join(path,"RENAME\_FILE")  
file\_docs = os.path.join(path\_rename,"solar01\_structure.txt")  
last\_step = {}

Input parameter the only one – path from first script. We will work with "solar01\_structure.txt", generated in part 1. Dictionary last step will keep couple “step” – “doc”

with open(file\_docs,encoding='utf8') as f:  
 for i in f:  
 i = i.replace("\n","")  
 p = i.split("\\")  
 if p[len(p)-2] not in last\_step.keys():  
 last\_step[p[len(p) - 2]] = ["\*"]  
 b = last\_step.get(p[len(p) - 2])  
 b.append(p[len(p) - 1])  
 last\_step[p[len(p) - 2]] = b  
 else:  
 b = last\_step.get(p[len(p) - 2])  
 b.append(p[len(p) - 1])  
 last\_step[p[len(p) - 2]] = b

Create dictionary “step” – “doc”

file = os.path.join(path,"blue.dat")

file\_lines=[]  
direct=[]  
with codecs.open(file,"r", "utf\_16\_le") as f:  
 file\_lines = [str(line.replace('\n', "")) for line in f]  
  
for i in range(file\_lines.index("100")-3):  
 direct.append(file\_lines[i])

Read file with full structure

nodes\_name = dict(zip(direct[1::6], direct[5::6]))  
nodes\_num = dict(zip(direct[1::6], direct[3::6]))  
nodes\_type = dict(zip(direct[1::6], direct[2::6]))  
scenars=[i for i in nodes\_name if nodes\_type[i]=="BMSC"]  
process=[i for i in nodes\_name if nodes\_type[i]=="BMPG"]  
steps=[i for i in nodes\_name if nodes\_type[i]=="BMPC"]

Create dictionary by node type: scenarios - "BMSC", processes - "BMPG", steps - "BMPC".

Finally, make template:

used=[]  
with open(os.path.join(path\_rename,"new\_shablon.txt"), 'w', encoding='utf8') as f:  
 f.truncate()  
 for i,val in enumerate(scenars):  
 f.writelines("\tCreate a scenario\t1\tSCN\t{0}\t\t\t\n".format(nodes\_name[val]))  
 if nodes\_name[val] in last\_step.keys():  
 used.append(nodes\_name[val])  
 a = last\_step.get(nodes\_name[val])  
 for k in a:  
 if k != "\*":  
 f.writelines(" \t\t2\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(k, os.path.join(path\_rename, k)))  
 for j,valj in enumerate(process):  
 if i != len(scenars)-1:  
 if val <= valj < scenars[i+1]:  
 f.writelines("\tCreate process\t2\tPROC\t{0}\t\t\t\n".format(nodes\_name[valj]))  
 if nodes\_name[valj] in last\_step.keys():  
 used.append(nodes\_name[valj])  
 a = last\_step.get(nodes\_name[valj])  
 for k in a:  
 if k != "\*":  
 f.writelines(" \t\t3\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(k, os.path.join(path\_rename, k)))  
 for m,valm in enumerate(steps):  
 if j != len(steps)-1:  
 if valj <= valm < process[j + 1]:  
 f.writelines("\tCreate step\t3\tREF\_PROCSTEP\t{0}\t\t\t\n".format(nodes\_name[valm]))  
 if nodes\_name[valm] in last\_step.keys():  
 used.append(nodes\_name[valm])  
 a = last\_step.get(nodes\_name[valm])  
 for k in a:  
 if k != "\*":  
 f.writelines(" \t\t4\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(k, os.path.join(path\_rename, k)))

1. We create new file "new\_shablon.txt".
2. Write line "Create a scenario {NAME}".
3. Check last\_steps dictionary to have scenario same files.
4. Check process in this scenario and write line "Create process {NAME}”.
5. Check last\_steps dictionary to have process same files.
6. Check steps in this process and write line "Create step {NAME}”.
7. Check last\_steps dictionary to have step same files.
8. Next in block “else” make the same for boundary note
9. All entering write to used list
10. In the end compare last\_step and used and write to file difference
11. for n in list(set.difference(set(last\_step) - set(used))):  
     #print(n, last\_step.get(n))  
     a = last\_step.get(n)  
     for k in a:  
     if k != "\*":  
     f.writelines(  
     " \t\t1\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(  
     k, os.path.join(path\_rename, k)))

## Full code:

import os  
import codecs  
path = input()  
path\_rename = os.path.join(path,"RENAME\_FILE")  
file\_docs = os.path.join(path\_rename,"solar01\_structure.txt")  
last\_step = {} #последний шаг - документ  
  
with open(file\_docs,encoding='utf8') as f:  
 for i in f:  
 i = i.replace("\n","")  
 p = i.split("\\")  
 if p[len(p)-2] not in last\_step.keys():  
 last\_step[p[len(p) - 2]] = ["\*"]  
 b = last\_step.get(p[len(p) - 2])  
 b.append(p[len(p) - 1])  
 last\_step[p[len(p) - 2]] = b  
 else:  
 b = last\_step.get(p[len(p) - 2])  
 b.append(p[len(p) - 1])  
 last\_step[p[len(p) - 2]] = b  
#print(last\_step)  
  
file = os.path.join(path,"blue.dat")

file\_lines=[]  
direct=[]  
with codecs.open(file,"r", "utf\_16\_le") as f:  
 file\_lines = [str(line.replace('\n', "")) for line in f]  
  
for i in range(file\_lines.index("100")-3):  
 direct.append(file\_lines[i])  
  
nodes\_name = dict(zip(direct[1::6], direct[5::6]))  
nodes\_num = dict(zip(direct[1::6], direct[3::6]))  
nodes\_type = dict(zip(direct[1::6], direct[2::6]))  
scenars=[i for i in nodes\_name if nodes\_type[i]=="BMSC"]  
process=[i for i in nodes\_name if nodes\_type[i]=="BMPG"]  
steps=[i for i in nodes\_name if nodes\_type[i]=="BMPC"]  
  
#делаем шаблон  
used=[]  
with open(os.path.join(path\_rename,"new\_shablon.txt"), 'w', encoding='utf8') as f:  
 f.truncate()  
 for i,val in enumerate(scenars):  
 f.writelines("\tCreate a scenario\t1\tSCN\t{0}\t\t\t\n".format(nodes\_name[val]))  
 if nodes\_name[val] in last\_step.keys():  
 used.append(nodes\_name[val])  
 a = last\_step.get(nodes\_name[val])  
 for k in a:  
 if k != "\*":  
 f.writelines(" \t\t2\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(k, os.path.join(path\_rename, k)))  
 for j,valj in enumerate(process):  
 if i != len(scenars)-1:  
 if val <= valj < scenars[i+1]:  
 f.writelines("\tCreate process\t2\tPROC\t{0}\t\t\t\n".format(nodes\_name[valj]))  
 if nodes\_name[valj] in last\_step.keys():  
 used.append(nodes\_name[valj])  
 a = last\_step.get(nodes\_name[valj])  
 for k in a:  
 if k != "\*":  
 f.writelines(" \t\t3\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(k, os.path.join(path\_rename, k)))  
 for m,valm in enumerate(steps):  
 if j != len(steps)-1:  
 if valj <= valm < process[j + 1]:  
 f.writelines("\tCreate step\t3\tREF\_PROCSTEP\t{0}\t\t\t\n".format(nodes\_name[valm]))  
 if nodes\_name[valm] in last\_step.keys():  
 used.append(nodes\_name[valm])  
 a = last\_step.get(nodes\_name[valm])  
 for k in a:  
 if k != "\*":  
 f.writelines(" \t\t4\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(k, os.path.join(path\_rename, k)))  
 else:  
 if valj <= valm:  
 f.writelines("\tCreate step\t3\tREF\_PROCSTEP\t{0}\t\t\t\n".format(nodes\_name[valm]))  
 if nodes\_name[valm] in last\_step.keys():  
 used.append(nodes\_name[valm])  
 a = last\_step.get(nodes\_name[valm])  
 for k in a:  
 if k != "\*":  
 f.writelines(  
 " \t\t4\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(k, os.path.join(path\_rename, k)))  
 else:  
 if val <= valj:  
 f.writelines("\tCreate process\t2\tPROC\t{0}\t\t\t\n".format(nodes\_name[valj]))  
 if nodes\_name[valj] in last\_step.keys():  
 used.append(nodes\_name[valj])  
 a = last\_step.get(nodes\_name[valj])  
 for k in a:  
 if k != "\*":  
 f.writelines(  
 " \t\t3\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(k, os.path.join(path\_rename, k)))  
 for m,valm in enumerate(steps):  
 if j != len(steps)-1:  
 if valj <= valm < process[j + 1]:  
 f.writelines("\tCreate step\t3\tREF\_PROCSTEP\t{0}\t\t\t\n".format(nodes\_name[valm]))  
 if nodes\_name[valm] in last\_step.keys():  
 used.append(nodes\_name[valm])  
 a = last\_step.get(nodes\_name[valm])  
 for k in a:  
 if k != "\*":  
 f.writelines(  
 " \t\t4\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(k, os.path.join(path\_rename, k)))  
 else:  
 if valj <= valm:  
 f.writelines("\tCreate step\t3\tREF\_PROCSTEP\t{0}\t\t\t\n".format(nodes\_name[valm]))  
 if nodes\_name[valm] in last\_step.keys():  
 used.append(nodes\_name[valm])  
 a = last\_step.get(nodes\_name[valm])  
 for k in a:  
 if k != "\*":  
 f.writelines(  
 " \t\t4\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(k, os.path.join(path\_rename, k)))  
  
 for n in list(set.difference(set(last\_step) - set(used))):  
 #print(n, last\_step.get(n))  
 a = last\_step.get(n)  
 for k in a:  
 if k != "\*":  
 f.writelines(  
 " \t\t1\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n \t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n \t\t\t\t\tsmd\_responsible\tname\n \t\t\t\t\t$FILE\_PATH\t{1}\n".format(  
 k, os.path.join(path\_rename, k)))  
print("pass")

# How to import docs from Solution Manager 7.1 solar01 to Solution Manager 7.2 soldoc. Part 3

Now we want to generate directory of folders with docs from SolMan.

path = input()  
path\_rename = os.path.join(path,"RENAME\_FILE")  
file\_docs = os.path.join(path\_rename,"solar01\_structure.txt")  
last\_step = {} # “step” – “doc”  
z=30 #name length

Input parameter the only one – path from first script. We will work with "solar01\_structure.txt", generated in part 1. Dictionary last step will keep couple “step” – “doc”. Z=30 – name length (folder name and path have limit in Windows)

with open(file\_docs,encoding='utf8') as f:  
 for i in f:  
 i = i.replace("\n","")  
 p = i.split("\\")  
 x = p[len(p) - 2][:z].strip()  
 y = p[len(p) - 1].strip()  
 if x == "Организационные единицы" or x == "Основные данные" or x == "Бизнес-процессы" or x == "Бизнес-сценарии":  
 x = p[len(p) - 3][:z].strip()  
 if x not in last\_step.keys():  
 last\_step[x] = ["\*"]  
 b = last\_step.get(x)  
 b.append(y)  
 last\_step[x] = b  
 else:  
 b = last\_step.get(x)  
 b.append(y)  
 last\_step[x] = b

Make dictionary “last step” – “doc”, have exception for service nodes like:

* "Организационные единицы" – Organizational units
* "Основные данные" – Basic data
* "Бизнес-процессы" – Business processes
* "Бизнес-сценарии" – Business scenarios

file = os.path.join(path,"blue.dat")

file\_lines=[]  
direct=[]  
with codecs.open(file,"r", "utf\_16\_le") as f:  
 file\_lines = [str(line.replace('\n', "")) for line in f]

Read "blue.dat" file.

for i in range(file\_lines.index("100")-3):  
 direct.append(file\_lines[i].strip())  
  
nodes\_name = dict(zip(direct[1::6], direct[5::6]))  
nodes\_num = dict(zip(direct[1::6], direct[3::6]))  
nodes\_type = dict(zip(direct[1::6], direct[2::6]))  
scenars=[i for i in nodes\_name if nodes\_type[i]=="BMSC"]  
process=[i for i in nodes\_name if nodes\_type[i]=="BMPG"]  
steps=[i for i in nodes\_name if nodes\_type[i]=="BMPC"]

Create dictionary by node type: scenarios - "BMSC", processes - "BMPG", steps - "BMPC".

match = ["\\","/",":","\*","?","\"","<",">","|"]  
for n in nodes\_name:  
 for i in match:  
 nodes\_name[n]=nodes\_name[n].replace(i,"")

Delete special symbols in file names.

for n in nodes\_name:  
 if len(nodes\_name[n])>z:  
 nodes\_name[n]=nodes\_name[n][:z].strip()

Cut the name to length “Z”

name = ["\(не исп","\(не акт","\(пер"]  
for key in last\_step.keys():  
 b = last\_step.get(key)  
 c = []  
 for doc in b:  
 for nam in name:  
 search = re.search(nam, doc.lower())  
 if search:  
 c.append(doc)  
 for item in c:  
 b.remove(item)  
 last\_step[key] = b

Delete files we want by name start with["\(не исп","\(не акт","\(пер"], where “(не исп” and another – text you can change.

used=[]  
for i,val in enumerate(scenars):  
 path\_scenario = os.path.join(path\_rename, nodes\_name[val])  
 if not os.path.exists(path\_scenario):  
 os.makedirs(path\_scenario)  
 if nodes\_name[val] in last\_step.keys():  
 used.append(nodes\_name[val])  
 a = last\_step.get(nodes\_name[val])  
 for k in a:  
 if k != "\*":  
 shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_scenario, k))  
 for j,valj in enumerate(process):  
 if i != len(scenars)-1:  
 if val <= valj < scenars[i+1]:  
 path\_process = os.path.join(path\_scenario, nodes\_name[valj])  
 if not os.path.exists(path\_process):  
 os.makedirs(path\_process)  
 if nodes\_name[valj] in last\_step.keys():  
 used.append(nodes\_name[valj])  
 a = last\_step.get(nodes\_name[valj])  
 for k in a:  
 if k != "\*":  
 shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_process, k))  
 for m,valm in enumerate(steps):  
 if j != len(steps)-1:  
 if valj <= valm < process[j + 1]:  
   
 path\_step = os.path.join(path\_process, nodes\_name[valm])  
 if not os.path.exists(path\_step):  
 os.makedirs(path\_step)  
 if nodes\_name[valm] in last\_step.keys():  
 used.append(nodes\_name[valm])  
 a = last\_step.get(nodes\_name[valm])  
 for k in a:  
 if k != "\*":

shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_step, k))

1. We create folder with scenario name.
2. Check last\_steps dictionary to have scenario same files.
3. Check process in this scenario and create folder with process name.
4. Check last\_steps dictionary to have process same files.
5. Check step in this process and create folder with step name.
6. Check last\_steps dictionary to have step same files.
7. Next in block “else” make the same for boundary note
8. All entering write to used list
9. In the end compare last\_step and used and move difference to folder " FOLDER\_NAME " (change)

for n in list(set.difference(set(last\_step) - set(used))):  
 path\_empty = os.path.join(path\_rename, "FOLDER\_NAME")  
 if not os.path.exists(path\_empty):  
 os.makedirs(path\_empty)  
 a = last\_step.get(n)  
 for k in a:  
 if k != "\*":  
 shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_empty, k))

## Full code:

import os  
import codecs  
import shutil  
import re  
  
path = input()  
path\_rename = os.path.join(path,"RENAME\_FILE")  
file\_docs = os.path.join(path\_rename,"solar01\_structure.txt")  
last\_step = {}   
z=30   
  
with open(file\_docs,encoding='utf8') as f:  
 for i in f:  
 i = i.replace("\n","")  
 p = i.split("\\")  
 x = p[len(p) - 2][:z].strip()  
 y = p[len(p) - 1].strip()  
 if x == "Организационные единицы" or x == "Основные данные" or x == "Бизнес-процессы" or x == "Бизнес-сценарии":  
 x = p[len(p) - 3][:z].strip()  
 if x not in last\_step.keys():  
 last\_step[x] = ["\*"]  
 b = last\_step.get(x)  
 b.append(y)  
 last\_step[x] = b  
 else:  
 b = last\_step.get(x)  
 b.append(y)  
 last\_step[x] = b  
  
file = os.path.join(path,"blue.dat")

file\_lines=[]  
direct=[]  
with codecs.open(file,"r", "utf\_16\_le") as f:  
 file\_lines = [str(line.replace('\n', "")) for line in f]  
  
for i in range(file\_lines.index("100")-3):  
 direct.append(file\_lines[i].strip())  
  
nodes\_name = dict(zip(direct[1::6], direct[5::6]))  
nodes\_num = dict(zip(direct[1::6], direct[3::6]))  
nodes\_type = dict(zip(direct[1::6], direct[2::6]))  
scenars=[i for i in nodes\_name if nodes\_type[i]=="BMSC"]  
process=[i for i in nodes\_name if nodes\_type[i]=="BMPG"]  
steps=[i for i in nodes\_name if nodes\_type[i]=="BMPC"]  
  
match = ["\\","/",":","\*","?","\"","<",">","|"]  
for n in nodes\_name:  
 for i in match:  
 nodes\_name[n]=nodes\_name[n].replace(i,"")  
  
for n in nodes\_name:  
 if len(nodes\_name[n])>z:  
 nodes\_name[n]=nodes\_name[n][:z].strip()  
  
name = ["\(не исп","\(не акт","\(пер"]  
for key in last\_step.keys():  
 b = last\_step.get(key)  
 c = []  
 for doc in b:  
 for nam in name:  
 search = re.search(nam, doc.lower())  
 if search:  
 c.append(doc)  
 for item in c:  
 b.remove(item)  
 last\_step[key] = b  
  
used=[]  
for i,val in enumerate(scenars):  
   
 path\_scenario = os.path.join(path\_rename, nodes\_name[val])  
 if not os.path.exists(path\_scenario):  
 os.makedirs(path\_scenario)  
 if nodes\_name[val] in last\_step.keys():  
 used.append(nodes\_name[val])  
 a = last\_step.get(nodes\_name[val])  
 for k in a:  
 if k != "\*":  
   
 shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_scenario, k))  
 for j,valj in enumerate(process):  
 if i != len(scenars)-1:  
 if val <= valj < scenars[i+1]:  
   
 path\_process = os.path.join(path\_scenario, nodes\_name[valj])  
 if not os.path.exists(path\_process):  
 os.makedirs(path\_process)  
 if nodes\_name[valj] in last\_step.keys():  
 used.append(nodes\_name[valj])  
 a = last\_step.get(nodes\_name[valj])  
 for k in a:  
 if k != "\*":  
   
 shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_process, k))  
 for m,valm in enumerate(steps):  
 if j != len(steps)-1:  
 if valj <= valm < process[j + 1]:  
   
 path\_step = os.path.join(path\_process, nodes\_name[valm])  
 if not os.path.exists(path\_step):  
 os.makedirs(path\_step)  
 if nodes\_name[valm] in last\_step.keys():  
 used.append(nodes\_name[valm])  
 a = last\_step.get(nodes\_name[valm])  
 for k in a:  
 if k != "\*":  
 shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_step, k))  
 else:  
 if valj <= valm:  
 path\_step = os.path.join(path\_process, nodes\_name[valm])  
 if not os.path.exists(path\_step):  
 os.makedirs(path\_step)  
 if nodes\_name[valm] in last\_step.keys():  
 used.append(nodes\_name[valm])  
 a = last\_step.get(nodes\_name[valm])  
 for k in a:  
 if k != "\*":  
 shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_step, k))  
 else:  
 if val <= valj:  
   
 path\_process = os.path.join(path\_scenario, nodes\_name[valj])  
 if not os.path.exists(path\_process):  
 os.makedirs(path\_process)  
 if nodes\_name[valj] in last\_step.keys():  
 used.append(nodes\_name[valj])  
 a = last\_step.get(nodes\_name[valj])  
 for k in a:  
 if k != "\*":  
   
 shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_process, k))  
 for m,valm in enumerate(steps):  
 if j != len(steps)-1:  
 if valj <= valm < process[j + 1]:  
   
 path\_step = os.path.join(path\_process, nodes\_name[valm])  
 if not os.path.exists(path\_step):  
 os.makedirs(path\_step)  
 if nodes\_name[valm] in last\_step.keys():  
 used.append(nodes\_name[valm])  
 a = last\_step.get(nodes\_name[valm])  
 for k in a:  
 if k != "\*":  
   
 shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_step, k))  
 else:  
 if valj <= valm:  
   
 path\_step = os.path.join(path\_process, nodes\_name[valm])  
 if not os.path.exists(path\_step):  
 os.makedirs(path\_step)  
 if nodes\_name[valm] in last\_step.keys():  
 used.append(nodes\_name[valm])  
 a = last\_step.get(nodes\_name[valm])  
 for k in a:  
 if k != "\*":  
 shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_step, k))  
  
for n in list(set.difference(set(last\_step) - set(used))):  
 path\_empty = os.path.join(path\_rename, "ЕАСУП. Шаблон")  
 if not os.path.exists(path\_empty):  
 os.makedirs(path\_empty)  
 a = last\_step.get(n)  
 for k in a:  
 if k != "\*":  
  
 shutil.copyfile(os.path.join(path\_rename, k), os.path.join(path\_empty, k))  
print("pass")

# How to import docs from Solution Manager 7.1 solar01 to Solution Manager 7.2 soldoc. Part 4

Create structure in console by “blue.dat”.

"D:\PATH\blue.dat"– path to file.

import codecs  
import os  
  
file = r"D:\PATH\blue.dat"  
file\_lines=[]  
direct=[]  
  
with codecs.open(file,"r", "utf\_16\_le") as f:  
 file\_lines = [str(line.replace('\n', "")) for line in f]  
  
for i in range(file\_lines.index("100")-3):  
 direct.append(file\_lines[i])  
  
nodes\_name = dict(zip(direct[1::6], direct[5::6]))  
nodes\_num = dict(zip(direct[1::6], direct[3::6]))  
nodes\_type = dict(zip(direct[1::6], direct[2::6]))  
scenars=[i for i in nodes\_name if nodes\_type[i]=="BMSC"]  
process=[i for i in nodes\_name if nodes\_type[i]=="BMPG"]  
steps=[i for i in nodes\_name if nodes\_type[i]=="BMPC"]  
  
for i,val in enumerate(scenars):  
 print("\tCreate a scenario\t1\tSCN\t{0}\t\t\t\n".format(nodes\_name[val]))  
 for j,valj in enumerate(process):  
 if i!= len(scenars)-1:  
 if val<=valj<scenars[i+1]:  
 print("\tCreate process\t2\tPROC\t{0}\t\t\t\n".format(nodes\_name[valj]))  
 for m,valm in enumerate(steps):  
 if j != len(steps)-1:  
 if valj <= valm < process[j + 1]:  
 print("\tCreate step\t3\tREF\_PROCSTEP\t{0}\t\t\t\n".format(nodes\_name[valm]))  
 else:  
 if valj <= valm:  
 print("\tCreate step\t3\tREF\_PROCSTEP\t{0}\t\t\t\n".format(nodes\_name[valm]))  
 else:  
 if val <= valj:  
 print("\tCreate process\t2\tPROC\t{0}\t\t\t\n".format(nodes\_name[valj]))  
 for m,valm in enumerate(steps):  
 if j != len(steps)-1:  
 if valj <= valm < process[j + 1]:  
 print("\tCreate step\t3\tREF\_PROCSTEP\t{0}\t\t\t\n".format(nodes\_name[valm]))  
 else:  
 if valj <= valm:  
 print("\tCreate step\t3\tREF\_PROCSTEP\t{0}\t\t\t\n".format(nodes\_name[valm]))

# How to import docs from Solution Manager 7.1 solar01 to Solution Manager 7.2 soldoc. Part 5

Make template only for node with files.

import os  
import codecs  
import shutil  
  
path = input()  
  
path\_rename = os.path.join(path,"RENAME\_FILE")  
file = os.path.join(path\_rename,"solar01\_structure.txt")  
scenario = {} #сценарий - бизнес процесс  
bus\_step = {} #бизнес процесс - шаг  
last\_step = {} #последний шаг - документ  
  
with open(file,encoding='utf8') as f:  
 #solar\_path = [line for line in file]  
 for i in f:  
 i = i.replace("\n","")  
 p = i.split("\\")  
  
 if p[len(p)-2] not in last\_step.keys():  
 last\_step[p[len(p) - 2]] = ["\*"]  
 b = last\_step.get(p[len(p) - 2])  
 b.append(p[len(p) - 1])  
 last\_step[p[len(p) - 2]] = b  
 else:  
 b = last\_step.get(p[len(p) - 2])  
 b.append(p[len(p) - 1])  
 last\_step[p[len(p) - 2]] = b  
 if "Бизнес-сценарии" in i:  
  
 a=len(p)  
 m=int(p.index("Бизнес-сценарии"))+2  
 if int(len(p)) > int(p.index("Бизнес-сценарии"))+2:  
 if p[p.index("Бизнес-сценарии")+1] not in scenario.keys():  
 scenario[p[p.index("Бизнес-сценарии")+1]] = ["\*"]  
 if "Бизнес-процессы" in i:  
 if int(len(p)) > int(p.index("Бизнес-процессы")) + 2:  
  
 x = p.index("Бизнес-процессы")+1  
 z = p[p.index("Бизнес-процессы")+1]  
  
 y = scenario.get(p[p.index("Бизнес-сценарии")+1])  
  
 if p[p.index("Бизнес-процессы")+1] not in y:  
 y.append(p[p.index("Бизнес-процессы") + 1])  
 scenario[p[p.index("Бизнес-сценарии") + 1]] = y  
 if int(len(p)) > int((p.index("Бизнес-процессы") + 3)):   
 if p[p.index("Бизнес-процессы")+1] not in bus\_step.keys():  
 bus\_step[p[p.index("Бизнес-процессы") + 1]] = ["\*"]  
 c = bus\_step.get(p[p.index("Бизнес-процессы") + 1])  
 if p[p.index("Бизнес-процессы") + 2] not in c:  
 c.append(p[p.index("Бизнес-процессы") + 2])  
 bus\_step[p[p.index("Бизнес-процессы") + 1]] = c  
 else:  
 bus\_step["Бизнес-процессы"] = ["\*"]  
 else:  
 if "Бизнес-сценарии" not in scenario.keys():  
 scenario["Бизнес-сценарии"] = ["\*"]  
 else:  
 if p[0] not in scenario.keys():  
 scenario[p[0]] = ["\*"]  
  
  
with open(os.path.join(path\_rename,"shablon.txt"), 'w', encoding='utf8') as f:  
 f.truncate()  
 for scen in scenario.keys():  
 if scen == "": f.writelines("\tCreate a scenario\t1\tSCN\t{0}\t\t\t\n".format("NoName"))  
 else:  
 f.writelines("\tCreate a scenario\t1\tSCN\t{0}\t\t\t\n".format(scen))  
 if scen in last\_step.keys():  
 a = last\_step.get(scen)  
 for i in a:  
 if i!="\*":  
 f.writelines("\t \t2\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n\t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n\t\t\t\t\t$FILE\_PATH\t{1}\n".format(i, os.path.join(path\_rename, i)))  
 for bp in scenario.get(scen):  
   
 if bp != "\*":  
 f.writelines("\tCreate process\t2\tPROC\t{0}\t\t\t\n".format(bp))  
 if bp in last\_step.keys():  
 a = last\_step.get(bp)  
   
 for i in a:  
 if i != "\*":  
 f.writelines("\t \t3\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n\t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n\t\t\t\t\t$FILE\_PATH\t{1}\n".format(i, os.path.join(path\_rename, i)))  
 if bp in bus\_step.keys():  
  
 for step in bus\_step.get(bp):  
 if step != "\*":  
 f.writelines("\tCreate step\t3\tREF\_PROCSTEP\t{0}\t\t\t\n".format(step))  
 if step in last\_step.keys():  
 a = last\_step.get(step)  
 for i in a:  
 if i != "\*":  
 f.writelines("\t \t4\tKWOBJ\t{0}\tsmd\_doctype\tZSB\_0009\t\n\t\t\t\t\tsmd\_state\t0IN\_PROGRESS\n\t\t\t\t\t$FILE\_PATH\t{1}\n".format(i, os.path.join(path\_rename, i)))  
  
print("pass")