Este es el título del documento

NOMBRE

12 de junio de 2020

1. Codigos

1.1. Insertar codigo desde un archivo

```c
#include <stdio.h>

int main(void)
{
    char *str = "prueba";
    printf("Esto es un codigo de %s", str);

    for (int i = 0; i < 5; i++)
    {
        printf("Contando... %d", i);
        if (i == 3)
            printf("El numero es %d:", 3)
    }

    return 0;
}
```

Código 1: Un codigo hecho en C.

1.2. Pseudocodigo

```
INICIO;
char *str = "prueba";
Imp("Esto es un codigo de $str");
Para (int i = 0; i < 5; i++)
{
    Imp("Contando... $i");
    Si (i == 3)
        Imp("El numero es %d:", 3)
}
FIN;
```

Código 2: Pseudocodigo del programa anterior.

1.3. Codigos en otros lenguajes

```
import sys
import re

import os.path
from os import path

reglas = [
    ["return 0;\n}","\nFIN;"],
    ["return 0;}","\nFIN;"],
    ["return 1;\n}","\nFIN;"],
    ["return 1;}","\nFIN;"],
    ["return 0;","FIN;"],
    ["return 1;","FIN;"],
    ["printf","Imp"],
    ["scanf","Leer"],
    ["for ","Para "],
    ["if (","Si ("],
    ["else","else "],
    ["else ","Sino "],
    ["while ","Mientras "],
    ["do {","Hacer {"],
    ["int argc,char **argv",""],
    ["int main(void)","int main()"],
    ["int main()\n{","INICIO;"],
    ["int main(){","INICIO;"],
    ["Sino  Si ","SinoSi "],
    ['\\' + 'n',""], #quitar \n de printfs y demas
    ["\n    ","\n"]] #quitar 1 tabulacion



stdioSpecifiers = ['c', 'd', 'i', 'e', 'E', 'f', 'g', 'G', 'o', '
    ↪ s', 'u', 'x', 'X', 'p', 'n', '%']
stdioVarEnd = [' ', ',', ')']

class stdioVarPlaceholder():
    def __init__(self, found, text, specifier, startIndex,
        ↪ endIndex):
        self.found = found
        self.text = text
        self.specifier = specifier
        self.startIndex = startIndex
        self.endIndex = endIndex
def getStdioVarPlaceholder(st, startIndex):
    endIndex = 0
    try:
        firstIndex = st.index("%", startIndex)
    except ValueError:
        return(stdioVarPlaceholder(0, "", '', 0, 0))

    lowestendIndex = len(st)
    foundspecifier = ''

    for specifier in stdioSpecifiers:
        try:
```

```
54              endIndex = st.index(specifier, firstIndex + 1)
55              if (endIndex <= lowestendIndex):
56                  lowestendIndex = endIndex
57                  foundspecifier = specifier
58          except ValueError:
59              endIndex = lowestendIndex
60
61      endIndex = lowestendIndex
62
63      text = str(st[firstIndex:endIndex + 1])
64      return(stdioVarPlaceholder(1, text, foundspecifier,
           ↪ firstIndex, endIndex))
65  def getStdioVarReplace(st, startIndex):
66      endIndex = 0
67      try:
68          firstIndex = st.index(" ", startIndex)
69      except ValueError:
70          return(stdioVarPlaceholder(0, "", '', 0, 0))
71
72      lowestendIndex = len(st)
73      foundspecifier = ''
74
75      for specifier in stdioVarEnd:
76          try:
77              endIndex = st.index(specifier, firstIndex + 1)
78              if (endIndex <= lowestendIndex):
79                  lowestendIndex = endIndex
80                  foundspecifier = specifier
81          except ValueError:
82              endIndex = lowestendIndex
83      endIndex = lowestendIndex
84
85      text = str(st[firstIndex + 1:endIndex])
86      return(stdioVarPlaceholder(1, text, foundspecifier,
           ↪ firstIndex, endIndex))
87  def getVarList(st, startIndex):
88      varList = []
89      try:
90          firstVar = getStdioVarPlaceholder(st, startIndex)
91          if (firstVar.found == 0):
92              return(varList)
93          else:
94              lastIndex = firstVar.endIndex + 1
95              varList.append(firstVar)
96
97              while getStdioVarPlaceholder(st, lastIndex).found ==
                   ↪ 1:
98                  varList.append(getStdioVarPlaceholder(st,
                       ↪ lastIndex))
99                  lastIndex = getStdioVarPlaceholder(st, lastIndex)
                       ↪ .endIndex + 1
100     except ValueError:
101         varList = []
102     return(varList)
103
104 def getReplaceVarList(st, startIndex):
105     varList = []
```

```
106     try:
107         firstVar = getStdioVarReplace(st, startIndex)
108         if (firstVar.found == 0):
109             return(varList)
110         else:
111             lastIndex = firstVar.endIndex + 1
112             varList.append(firstVar)
113 
114             while getStdioVarReplace(st, lastIndex).found == 1:
115                 varList.append(getStdioVarReplace(st, lastIndex))
116                 lastIndex = getStdioVarReplace(st, lastIndex).
                        ↪ endIndex + 1
117     except ValueError:
118         varList = []
119     return(varList)
120 def find_between( s, first, last ):
121     try:
122         start = s.index( first ) + len( first )
123         end = s.index( last, start )
124         return s[start:end]
125     except ValueError:
126         return ""
127 def sfind_between( s, first, last ):
128     try:
129         start = s.index( first )
130         end = s.index( last, start ) + len( last )
131         return s[start:end]
132     except ValueError:
133         return ""
134 def del_closedcomment(src):
135     comentario=sfind_between(src,"/*","*/")
136     return(src.replace(comentario, ""))
137 def delete_includes(src): #Elimina includes.
138     print("\nEliminando includes...")
139     src_out = ""
140     for index, line in enumerate(src.splitlines(), start=1):
141         if (not line.startswith('#include')):
142                 src_out += line + '\n'
143         else:
144             print(" - [Linea: " + str(index) + "] " + line)
145     return(src_out)
146 def delete_defines(src): #Elimina defines.
147     print("\nEliminando defines...")
148     src_out = ""
149     for index, line in enumerate(src.splitlines(), start=1):
150         if (not line.startswith('#define')):
151                 src_out += line + '\n'
152         else:
153             print(" - [Linea: " + str(index) + "] " + line)
154     return(src_out)
155 def delete_commentlines(src): #elimina comentarios.
156     print("\nEliminando comentarios de linea...")
157     src_out = ""
158     for i, line in enumerate(src.splitlines(), start=1):
159         try:
160             start = line.index( "//" )
161             comentario = line[start:len(line)]
```

```
162             print(comentario)
163             line = line.replace(comentario, "")
164             src_out += line + "\n"
165         except ValueError:
166             src_out += line + "\n"
167 
168 
169     return(src_out)
170 def delete_commentclosed(src): #elimina comentarios cerrados.
171     print("\nEliminando comentarios cerrados...")
172     src_out = src
173 
174     while src_out.count("/*") > 0:
175         comentario=sfind_between(src_out,"/*","*/")
176         comentariopt=comentario.replace("\n","")
177         src_out = src_out.replace(comentario, "")
178         print(" - " + comentariopt)
179 
180     return(src_out)
181 def delete_blanks(src): #elimina lineas en blanco
182     print("\nEliminando Lineas en blanco...")
183     src_out = ""
184     for index, line in enumerate(src.splitlines(), start=1):
185         if (not line.isspace()):
186             if ((line and line.strip())):
187                 src_out += line + '\n'
188             else:
189                 print(" - [Linea: " + str(index) + "]")
190         else:
191             print(" - [Linea: " + str(index) + "]")
192     return(src_out)
193 def processtdio(src):
194     print("\nCorrigiendo printf's...")
195     src_out = ""
196     for i, line in enumerate(src.splitlines(), start=1):
197         try:
198             printfstart = line.index( "printf" )
199             printf_parentesis_start = line.index("(", printfstart
                ↪ )
200             varList = getVarList(line, printf_parentesis_start)
201             if (len(varList) > 0):
202                 printf_string_end = line.index("\",",
                    ↪ printf_parentesis_start) + 1
203                 printf_end = line.index(");", printf_string_end)
204                 replaceVarList = getReplaceVarList(line,
                    ↪ printf_string_end)
205                 line = line.replace(line[printf_string_end:
                    ↪ printf_end], "")
206                 #reemplazar variables...
207                 replaceTasks = ""
208                 for index, var_s in enumerate(varList, start=0):
209                     destvar = var_s.text
210                     fromvar = "$" + str(replaceVarList[index].
                        ↪ text)
211                     line = line.replace(destvar, fromvar)
212                     replaceTasks += destvar + " -> " + fromvar +
                        ↪ "  "
```

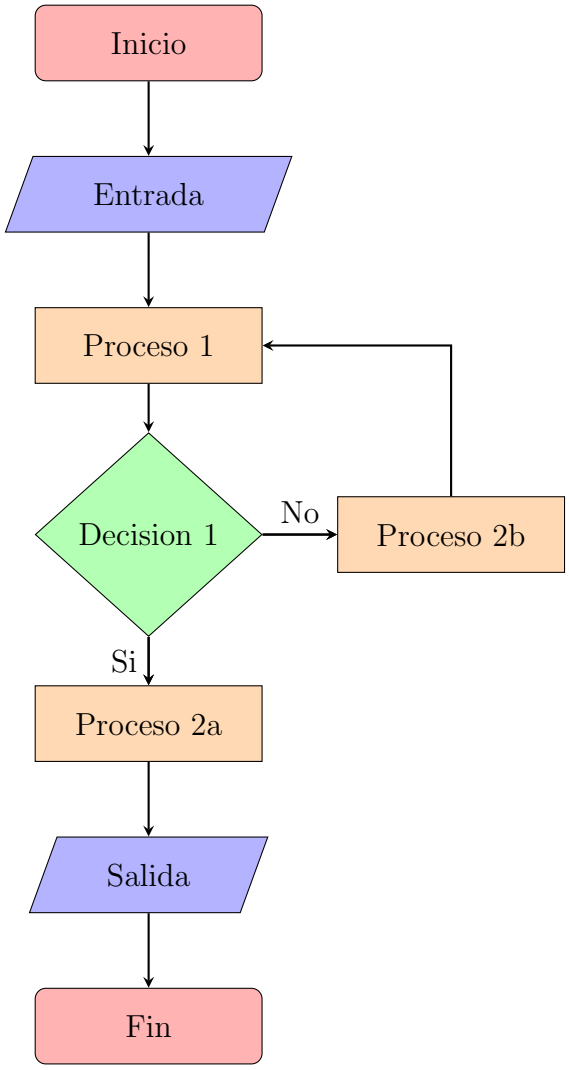
```
213                 print(" - [Linea: " + str(i) + "] " +
                        ↪ replaceTasks)
214         except ValueError:
215             pass
216         src_out += line + "\n"
217 
218     print("\nCorrigiendo scanf's...")
219     src_out_2 = ""
220     for i, line in enumerate(src_out.splitlines(), start=1):
221         try:
222             scanfstart = line.index( "scanf" )
223             scanf_parentesis_start = line.index("(", scanfstart)
224             scanf_string_start = line.index("(\"",
                        ↪ scanf_parentesis_start) + 1
225             scanf_string_end = line.index("\",",
                        ↪ scanf_string_start) + 3
226             line = line.replace(line[scanf_string_start:
                        ↪ scanf_string_end], "")
227             line = line.replace("&", "")
228             print(" - [Linea: " + str(i) + "] " + line[scanfstart
                        ↪ :len(line)])
229         except ValueError:
230             pass
231         src_out_2 += line + "\n"
232     return(src_out_2)
233 
234 #Inicializar el programa
235 if len(sys.argv) != 2:
236     print("Error, especifique archivo de entrada")
237     print("ejemplo: <python3 c-to-pseudo.py test.c>")
238     exit()
239 
240 filename = sys.argv[1]
241 outputfilename = filename.replace(".c", "-pseudo.txt")
242 
243 
244 
245 if not path.exists(filename):
246     print("No se encontro el archivo: <" + filename + ">")
247     exit()
248 
249 #Todo OK!
250 print("Abriendo el archivo: <" + filename + ">")
251 sourcefile = open(filename, "r").read()
252 
253 
254 
255 #eliminar includes, defines, lineas vacias, comentarios, etc.
      ↪ primera pasada
256 source = delete_includes(sourcefile)
257 source = delete_defines(source)
258 source = delete_commentlines(source)
259 source = delete_commentclosed(source)
260 source = delete_blanks(source)
261 
262 #procesar printf y scanf (stdio)
263 source = processtdio(source)
```

```
264 print(source)
265
266 #ejecutar reglas de reemplazo simples.
267 for regla in reglas:
268     source = source.replace(regla[0], regla[1])
269
270 #eliminar includes, defines, lineas vacias, comentarios, etc.
    ↪ segunda pasada
271 source = delete_blanks(source)
272
273
274
275
276
277
278 print("\nEscribiendo el archivo: <" + outputfilename + ">.")
279 output = open(outputfilename, "w+")
280 output.write(source)
281 print("Listo.")
```

Código 3: Un codigo en python muy largo.

2. Diagramas

2.1. Un diagrama de flujos



3. Texto demostrativo

Parrafo 1 Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Parrafo 2 Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Parrafo 3 Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Parrafo 4 Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Parrafo 5 Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor.

Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer.

Parrafo 6 Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio.