252SE311: ວິສະວະກຳຊອບແວຣ໌

ການສ້າງ, ທົດສອບ ແລະ ບຳລຸງຮັກສາ

ບົດທີ 11

ການຂຽນໂປຣແກຣມ

(Program Writing)

ເນື້ອໃນຫຍໍ້

- ມາດຕະຖານການຂຽນໂປຣແກຣມ
- ຫລັກການໃນການຂຽນໂປຣແກຣມ
- ການເຮັດເອກະສານກ່ຽວກັບໂປຣແກຣມ
- ຂະບວນການຂຽນໂປຣແກຣມ

ມາດຕະຖານການຂຽນໂປຣແກຣມ

- ເພື່ອໃຫ້ການປະກອບຊອບແວຣ໌, ການເຮັດເອກະສານ ແລະ ການບຳລຸງຮັກສາງ່າຍຂຶ້ນຈຳເປັນຈະຕ້ອງມີມາດຕະຖານ ແລະ ຂະບວນການເຮັດວຽກເພື່ອຄວບຄຸມການເຮັດວຽກຂອງໂປຣແກຣມເມີໃຫ້ຊອດຄ່ອງ, ມີລະບຽບ ແລະ ເປັນໄປຕາມທິດທາງດຽວກັນ
- ໂປຣແກຣມເມີທຸກຄົນຈະຕ້ອງຮັບຮູ້ ແລະ ເຂົ້າໃຈໃນມາດຕະຖານ ແລະ ຂະບວນການດັ່ງກ່າວຢ່າງຖືກຕ້ອງ ແລະ ກົງກັນ

ມາດຕະຖານການຂຽນໂປຣແກຣມ

- ປະໂຫຍດຕໍ່ໂປຣແກຣມເມີ
  - ຊ່ວຍໃຫ້ໂປຣແກຣມເມີຈັດລຳດັບຄວາມຄິດໄດ້ຢ່າງເປັນລະບົບ, ຫລີກລ້ຽງຂໍ້ຜິດພາດໃນເບື້ອງຕົ້ນ ເຊັ່ນ: ການກຳໜົດຊື່ຕົວປ່ຽນ, ຊື່ຖານຂໍ້ມູນທີ່ບໍ່ກົງກັບທີມອື່ນ
  - ຊ່ວຍໃຫ້ໂປຣແກຣມເມີເຮັດວຽກໄດ້ງ່າຍຂຶ້ນ
  - ເອກະສານທີ່ເຮັດຂຶ້ນຕາມມາດຕະຖານ ເຮັດໃຫ້ສາມາດກວດສອບຫາຂໍ້ຜິດພາດ ແລະ ການປ່ຽນແປງງ່າຍຂຶ້ນ
- ປະໂຫຍດຕໍ່ບຸກຄົນອື່ນ
  - ເຮັດໃຫ້ການປະກອບ ແລະ ທົດສອບຊອບແວຣ໌ດ້ວຍໂປຣແກຣມເມີຄົນອື່ນງ່າຍຂຶ້ນ

ຫລັກການໃນການຂຽນໂປຣແກຣມ

- ໂຄງສ້າງການຄວບຄຸມການເຮັດວຽກຂອງໂປຣແກຣມ
  - ໃນບົດຜ່ານມາໄດ້ຮູ້ເຖິງການຄວບຄຸມການເຮັດວຽກຂອງສ່ວນປະກອບຍ່ອຍຂອງຊອບແວຣ໌ຫລາຍຮູບແບບເຊັ່ນ: Call and Returen, Event-Based Control ແລະ Broadcast Model
  - ການຄວບຄຸມການເຮັດວຽກຂອງໂປຣແກຣມຍ່ອຍມີຄວາມສຳຄັນຫລາຍ ເນື່ອງຈາກມັນເຮັດໃຫ້ໂປຣແກຣມເຮັດວຽກໄປຕາມທິດທາງທີ່ຕ້ອງການໄດ້
  - ສຳຫລັບວິທີການໃນການຂຽນໂປຣແກຣມທີ່ຊ່ວຍໃຫ້ການອ່ານໂຄດໃຫ້ງ່າຍນັ້ນ ມີ 2 ວິທີການຄື:
    - ຂຽນໂຄດຈາກເທິງລົງລຸ່ມ
    - ແບ່ງໂປຣແກຣມອອກເປັນໂມດຸ່ມ

ຫລັກການໃນການຂຽນໂປຣແກຣມ

- ໂຄງສ້າງການຄວບຄຸມການເຮັດວຽກຂອງໂປຣແກຣມ
  - ຂຽນໂຄດຈາກເທິງລົງລຸ່ມ
    - ເປັນການຂຽນໂປຣແກຣມໃນລັກສະນະທີ່ເປັນໄປຕາມຄຳສັ່ງຄວບຄຸມຂອງໂປຣແກຣມ, ບໍ່ໃຫ້ມີລັກສະນະການຂຽນແບບກະໂດດໄປມາ
  - ແບ່ງໂປຣແກຣມອອກເປັນໂມດຸ່ນ
    - ເປັນການແບ່ງການເຮັດວຽກອອກເປັນໂມດຸ່ນ ແລະ ມີໂມດູ່ນຫລັກເຮັດໜ້າທີ່ໃນການເອີ້ນໃຊ້ໂມດູ່ນຍ່ອຍຕ່າງໆ
    - ເຮັດໃຫ້ການອ່ານໂຄດໂປຣແກຣມ, ການທົດສອບ ແລະ ການບຳລຸງຮັກສາງ່າຍຂຶ້ນ

ຫລັກການໃນການຂຽນໂປຣແກຣມ

- ↳ ໂຄງສ້າງການຄວບຄຸມການເຮັດວຽກຂອງໂປຣແກຣມ
  - ○ ຂຽນໂຄດຈາກເທິງລົງລຸ່ມ

```
    dis = normal;
    if (price >= 3000) goto A;
    dis = normal + bonus +1.5;
    dis = 0; goto C;
    if (price >= 5000) goto B;
    if (price >= 10000) goto C;
A:  if (price >= 5000) goto B;
    dis = normal; goto C;
B:  if (price >= 10000) goto C;
    dis = normal + bonus;
C:  next statement;
```

Top-down

```
if (price >= 3000) dis = normal;
elseif (price >= 5000) dis = normal + bonus;
elseif (price >= 10000) dis = normal + bonus + 1.5;
else dis = 0;
```

ສະແດງການປັບໂຄງສ້າງຄວບຄຸມຈາກແບບ ກະໂດດໄປມາເປັນແບບເທິງລົງລຸ່ມ

ຫລັກການໃນການຂຽນໂປຣແກຣມ

- ລຳດັບຂັ້ນຕອນຂອງຄຳສັ່ງ (Algorithm)
  - ເປັນການຈັດລຳດັບຂັ້ນຕອນຂອງຄຳສັ່ງ ຫຼື ລຳດັບຂັ້ນຕອນໃນການແກ້ໄຂບັນຫາໃດໜຶ່ງ
  - ໜ້າທີ່ຂອງໂປຣແກຣມເມີໃນຂັ້ນຕອນນີ້ແມ່ນການເອົາ Algorithm ທີ່ໄດ້ຈາກການອອກແບບມາຂຽນເປັນໂຄດໂປຣແກຣມພາຍໄຕ້ພາສາຂຽນໂປຣແກຣມ, ແພລດຟອມ ແລະ ຮາດແວຣ໌ທີ່ກຳໜົດ
  - ຫລັກການທີ່ຕ້ອງຄຳນຶງເຖິງເມື່ອແປງຈາກ Algorithm ມາເປັນໂຄດກໍ່ຄື: ຕ້ອງຂຽນໂຄດໃຫ້ມີປະສິດທິພາບ ແລະ ປະສິດທິຜົນ

ຫລັກການໃນການຂຽນໂປຣແກຣມ

- ໂຄງສ້າງຂອງຂໍ້ມູນ (Data Structure)
  - ແມ່ນລັກສະນະຂອງຂໍ້ມູນທີ່ຈະຖືກປະມວນຜົນໃນໂປຣແກຣມ
  - ໃຫ້ຂຽນໂປຣແກຣມຕາມໂຄງສ້າງຂໍ້ມູນ ດັ່ງນີ້
    1. ຂຽນໂປຣແກຣມໃຫ້ລຽບງ່າຍ
       - ບໍ່ວ່າໃນຂັ້ນຕອນການອອກແບບຈະກຳນົດຂໍ້ມູນມາໃນລັກສະນະໃດກໍ່ຕາມ ໃຫ້ເອົາມາປັບໃໝ່ໃຫ້ສາມາດຂຽນໂປຣແກຣມໄດ້ງ່າຍຂຶ້ນ
    2. ໃຊ້ໂຄງສ້າງຂໍ້ມູນເປັນຕົວກຳໜົດໂຄງສ້າງຂອງໂປຣແກຣມ
       - ກ່ອນຈະລົງມືຂຽນໂປຣແກຣມຄວນພິຈາລະນາໃຫ້ຮອບຄອບວ່າພາສາໂປຣແກຣມໃດທີ່ເໝາະສົມກັບໂຄງສ້າງຂໍ້ມູນທີ່ມຮຢູ່ສ່ວນໃຫຍ່

ຫລັກການໃນການຂຽນໂປຣແກຣມ

- ຫລັກປະຕິບັດອື່ນໆ
  - ແຍກ function ຮັບ ແລະ ສະແດງຜົນຂໍ້ມູນອອກຈາກໂຄດສ່ວນອື່ນໆ ເພື່ອໃຫ້ການແກ້ໄຂ ແລະ ປັບປຸງໂຄດໃນສ່ວນດັ່ງກ່າວງ່າຍຂຶ້ນ, ນອກຈາກນັ້ນຍັງເປັນການລຸດຈຳນວນການຂຽນໂຄດເພື່ອຮັບຂໍ້ມູນຊ້ຳໆ ແລະ ເຮັດໃຫ້ການອ່ານໂຄດງ່າຍຂຶ້ນ
  - ໃຊ້ວິທີການຂຽນລະຫັດທຽມເພື່ອທົດລອງອອກແບບ algorithm ຂອງໂປຣແກຣມກ່ອນລົງມືຂຽນໂຄດຕົວຈິງ
  - ຂຽນໂຄດເພື່ອໃຫ້ໂປຣແກຣມເຮັດວຽກໄດ້ແບບພໍ່ຄ່າວໆກ່ອນແລ້ວຄ່ອຍມາເກັບລາຍລະອຽດຕາມຫລັງ
  - ໃຫ້ຕອບສະໜອງຕາມສອງຫລັກການ Reuse: Consumer Reuse ແລະ Producer Reuse

ຫລັກການໃນການຂຽນໂປຣແກຣມ

- ຫລັກປະຕິບັດອື່ນໆ
  - Consumer Reuse
    1. Component ທີ່ກຳລັງພິຈາລະນາມີຂໍ້ມູນຕາມທີ່ຕ້ອງການບໍ່
    2. ຖ້າຕ້ອງມີການດັດແປງ component, ການດັດແປງນັ້ນຈະຄຸ້ມຄ່າຫຼືບໍ່ ປຽບທຽບກັບການສ້າງຂຶ້ນໃໝ່
    3. Component ດັ່ງກ່າວມີເອກະສານປະກອບມາໃຫ້ຢ່າງຄົບຖ້ວນບໍ່
    4. Component ດັ່ງກ່າວມີຂໍ້ມູນຊຸດທົດສອບທີ່ສົມບຸນໃຫ້ມານຳຫຼືບໍ່

ຫລັກການໃນການຂຽນໂປຣແກຣມ

- ຫລັກປະຕິບັດອື່ນໆ
  - Producer Reuse
    1. ສ້າງ component ໃຫ້ເປັນກາງຫລາຍທີ່ສຸດ ໂດຍການກຳໜົດຊື່ຕ໋ວປ່ຽນ ຫຼື ເງື່ອນໄຂໃຫ້ຊອດຄ່ອງກັບລະບົບວຽກທີ່ມັນຮັບຜິດຊອບ
    2. ກຳໜົດ Interface ຂອງ component ໃຫ້ຊັດເຈນ ແລະ ສົມບຸນທີ່ສຸດ
    3. ຄວນບອກຂໍ້ຜິດພາດທີ່ອາດເກີດຂຶ້ນ ແລະ ວິທີແກ້ໄຂໄວ້ດ້ວຍ
    4. ຊື່ຕ໋ວປ່ຽນຄວນສື່ຄວາມໝາຍຊັດເຈນ ແລະ ສາມາດຈຳແນກຄວາມແຕກຕ່າງໄດ້ງ່າຍ
    5. ຄວນເຮັດເອກະສານສະແດງໂ ຄງສ້າງຂອງຂໍ້ມູນ ແລະ algorithm
    6. ຈັດເກັບສ່ວນຕິດຕໍ່ລະຫວ່າງ component ກັບສ່ວນຕອບສະໜອງຕໍ່ຄວາມຜິດພາດໄດ້ຕ່າງຫາກ ເພື່ອການແກ້ໄຂທີ່ງ່າຍຂຶ້ນ

ການເຮັດເອກະສານກ່ຽວກັບໂປຣແກຣມ

- ການເຮັດເອກະສານພາຍໃນ
  - ເປັນເອກະສານສະແດງລາຍລະອຽດຂອງໂຄດທັງໝົດຂອງຊອບແວຣ໌ ແລະ ລວມໝາຍເຫດຂອງໂປຣແກຣມນຳ ເຊັ່ນ:
  - Header Comment Block
    - ເປັນການຂຽນໝາຍເຫດໄວ້ໃນສ່ວນຫົວຂອງໂປຣແກຣມ, ເປັນສ່ວນສະແດງທີ່ມາ, ໜ້າທີ່ ແລະ ການໃຊ້ງານໂປຣແກຣມ ຊຶ່ງປະກອບດ້ວຍລາຍລະອຽດດັ່ງນີ້:
      1. ຊື່ໂປຣແກຣມ ຄວນສື່ຄວາມໝາຍຢ່າງຊັດເຈນ
      2. ຊື່ໂປຣແກຣມເມີ ຊຶ່ງລວມທັງເບີໂທລະສັບ, e-mail address ນຳ
      3. ໜ້າທີ່ຂອງໂປຣແກຣມ
      4. ວັນທີຂຽນໂປຣແກຣມແລ້ວ ແລະ ວັນທີປັບປຸງ ລວມທັງຊື່ຫຼືໝາຍເລກລຸ້ນຂອງໂປຣແກຣມດ້ວຍ
      5. ຄວາມສຳຄັນຂອງໂປຣແກຣມ
      6. ວິທີໃຊ້ຂໍ້ມູນ, algorithm ແລະ ວິທີການຄວບຄຸມການເຮັດວຽກ

ການເຮັດເອກະສານກ່ຽວກັບໂປຣແກຣມ

↳ ການເຮັດເອກະສານພາຍໃນ

- Header Comment Block
  - ນອກຈາກນັ້ນໂປຣແກຣມເມີສາມາດອະທິບາຍລາຍລະອຽດເພີ່ມເຕີມໄດ້ດັ່ງນີ້:
    1. ຊື່ແລະຊະນິດຂໍ້ມູນຂອງຕົວປ່ຽນ
    2. ອະທິບາຍ algorithm ແລະ ການຕອບສະໜອງຕໍ່ຄວາມຜິດພາດທີ່ເກີດຂຶ້ນພໍສັງເຂບ
    3. ສະແດງຂໍ້ມູນນຳເຂົ້າແລະຜົນລັບທີ່ຄາດວ່າຈະໄດ້ຮັບ
    4. ວິທີທົດສອບໂປຣແກຣມແລະວິທີການໃຊ້ງານໂປຣແກຣມ
    5. ວິທີຂະຫຍາຍຂີດຄວາມສາມາດຂອງໂປຣແກຣມ

ການເຮັດເອກະສານກ່ຽວກັບໂປຣແກຣມ

- ການເຮັດເອກະສານພາຍໃນ
  - ຕົວຢ່າງ: Header Comment Block

```
PROGRAM SCAN: Program to scan a line of text for a given character.
PROGRAMMER: Tantika Kokpho (02-7896543 / kokpho@hotmail.com)
CALLING SEQUENCE: CALL SCAN(LENGTH, CHAR, NTEXT)
        where LENGTH is the length of the line to be scanned; CHAR is the character
        sought. Line of text is passed as array NTEXT.
VERSION 1: written 10 October 1989 by Tantika Kokpho.
REVISION 1.1: 11 November 2000 by Tantika Kokpho to improve searching algorithm.
PURPOSE: General-purpose scanning module to be used for each new line of text, no
        matter the length. One of several text utilities designed ot add a character, change or
        delete a character.
DATA STRUCTURES: Variable LENGTH - integer
        Variable CHAR - character
        Array NTEXT - character array of length LENGTH
ALGORITHM: Reads array NTEXT one character at atime; if CHAR is found, position in
        NTEXT returned in variable LENGTH; else variable LENGTH set ot 0.
```

ການເຮັດເອກະສານກ່ຽວກັບໂປຣແກຣມ

- ການເຮັດເອກະສານພາຍໃນ
  - ໝາຍເຫດສ່ວນອື່ນໆຂອງໂປຣແກຣມ
    - ຂຽນໝາຍເຫດກຳກັບແຕ່ລະສ່ວນການທຳງານຂອງໂປຣແກຣມໂດຍຕະຕ້ອງຂຽນລາຍລະອຽດເພີ່ມເຕີມທີ່ຈະຊ່ວຍໃຫ້ບຸກຄົນອື່ນເຂົ້າໃນໄດ້ງ່າຍ
    - ບໍ່ຄວນຂຽນຊ ້ຳ ໃນສິ່ງທີ່ມີຢູ່ແລ້ວ

ໝາຍເຫດທີ່ບໍ່ຖືກຕ້ອງ
```
/*Increment j */
J=j+1;
```

ໝາຍເຫດທີ່ຖືກຕ້ອງ
```
/*set counter to read next case */
J=j+1;
```

ການເຮັດເອກະສານກ່ຽວກັບໂປຣແກຣມ

- ການເຮັດເອກະສານພາຍໃນ
  - ຕັ້ງຊື່ຕົວປ່ຽນໃຫ້ສື່ຄວາມໝາຍຊັດເຈນ
    - Label ແລະ ຊື່ຕົວປ່ຽນຄວນສື່ຄວາມໝາຍ, ສາມາດຈຳແໜກຄວາມແຕກຕ່າງ ແລະ ບໍ່ຄວນຕັ້ງຊື່ຊ້ຳ
  - ຈັດຮູບແບບໂຄດ ແລະ ໝາຍເຫດໃຫ້ອ່ານງ່າຍ
    - ການຈັດຮູບແບບແລະຕົກແຕ່ງຂໍ້ຄວາມໂຄດຈະຊ່ວຍໃຫ້ອ່ານໝາຍເຫດໄດ້ງ່າຍ, ບໍ່ປົນກັບປະໂຫຍກຄຳສັ່ງຂອງໂຄດ
  - ເຮັດເອກະສານປະກອບການໃຊ້ງານຂໍ້ມູນ
    - ການກຳໜົດຊະນິດຂອງຂໍ້ມູນໃຫ້ກັບຕົວປ່ຽນ ຫຼື ການຈັດໂຄງສ້າງຂໍ້ມູນ ແລະ ການເອີ້ນໃຊ້ຂໍ້ມູນ ຄວນໃຫ້ລາຍລະອຽດເພີ່ມເຕີ່ມ

ການເຮັດເອກະສານກ່ຽວກັບໂປຣແກຣມ

- ການເຮັດເອກະສານພາຍນອກ
  - ເປັນການອະທິບາຍລາຍລະອຽດຂອງລະບົບ
  - ຄວນມີແຜນພາບ ຫຼືແບບຈຳລອງທີ່ສະແດງໃຫ້ເຫັນໂຄງສ້າງຂອງໂປຣແກຣມ ຫຼື component, ການຮັບ-ສົ່ງຂໍ້ມູນລະຫວ່າງຄອມໂພເນັ້ນ, ຄວາມສຳພັນຂອງຄອມໂພເນັ້ນທັງໝົດ
  - ສ່ວນປະກອບຂອງເອກະສານພາຍນອກມີດັ່ງນີ້:
    - ອະທິບາຍບັນຫາ
    - ອະທິບາຍ algorithm
    - ອະທິບາຍຂໍ້ມູນ

ຂະບວນການຂຽນໂປຣແກຣມ

- ສ້າງຄວາມເຂົ້າໃນຕໍ່ບັນຫາ
  - ແມ່ນການວິເຄາະວຽກທີ່ຈະເຮັດ ໂດຍສ່ວນໃຫຍ່ນັກຂຽນໂປຣແກຣມ ຈະໃຊ້ Flowchart ຫຼືແຜນພາບອື່ນໆ ເປັນເຄື່ອງມືໃນການຈຳລອງ ລຳດັບຂັ້ນຕອນຂອງວຽກ
- ວາງແຜນແກ້ໄຂບັນຫາ
  - ເປັນການອອກແບບວິທີແກ້ໄຂບັນຫາທີ່ຜ່ານການວິເຄາະມາແລ້ວ
- ດຳເນີນການຕາມແຜນ
- ທົບທວນ