Visit QB365-SLMS - https://www.qb365.in/slms/ Complete Questions and Answers

Answers Key:

5 Marks

1. அளபுருக்கள் என்பது செயற்கூறு வரையறையில் உள்ள மாறிகள் ஆகும்.

அ) தரவு வகை இல்லாத அளபுருக்கள்:

(i) செயற்கூறு வரையறைக்கு ஓர் எடுத்துக்காட்டைக் காணலாம்.

```
(requires: b> = 0)
    (returns: a to the power of b)
    let rec pow a b:=
        if b = 0 then 1
        else a * pow a(b - 1)
```

(ii) மேலேயுள்ள செயற்கூறு வரையறையில் 'b' என்ற மாறி அளபுரு ஆகும். மாறி 'b'க்கு அனுப்பப்படும் மதிப்பானது செயலுருபு ஆகும்.

(iii) செயற்கூறின் முன் நிபந்தனை (requires) மற்றும் பின் நிபந்தனை (return) கொடுக்கப்பட்டுள்ளது.

(iv) சில மொழிகளின் நிரல் பெயர்ப்பி இவ்வகை சிக்கல்களை நிரல் நெறிமுறைப்படி சரி செய்கிறது. ஆனால் சில நிரல் பெயர்ப்பிக்கு தரவு வகையைக் குறிப்பிடுவது கட்டாயமாகும்.

(v) மேலே உள்ள செயற்கூறு வரையறையில், if கோவை, then கிளைக்கு மதிப்பு 1 யைத் திருப்பி அனுப்பினால், தரவு வகை (data type) விதிப்படி if கோவை முழுவதுமே 'int' தரவு வகைக் கொண்டிருக்கும்.

if கோவையின் தரவு வகை int ஆக இருப்பதால், செயற்கூறின் திருப்பி அனுப்பும் மதிப்பும் int ஆக இருக்கும், 'b' யின் மதிப்பு சுழியத்தோடு ஒப்பீடு செய்யப்படுகிறது.

(vi) அதனால் 'b' யின் தரவுவகை 'int' ஆகும். * செயற்குறியுடன் 'a' யின் மதிப்பு மற்றொரு கோவையோடு பெருக்குத் தொகையைக் கணக்கிடுவதால், 'a' யின் வகையும் int ஆகும்.

ஆ) தரவு வகையுடன் கூடிய அளபுருக்கள் :

(i) சில காரணங்களுக்காக, இப்பொழுது நாம் அதே செயற்கூறு வரையறை தரவு வகையுடன் எழுதலாம்.

```
(requires: b > = 0)
(returns: a to the power of b)
let rec pow(a: int) (b: int) : int: =
    if b = 0 then 1
    else a * pow b(a - 1)
```

(ii) 'a' மற்றும் 'b' தரவு வகை குறிப்பு (type annotations) எழுதும் போது, அடைப்புக்குறிக்குள் ( ) அவசியமானது ஆகும்.

(iii) ஏனெனில், நிரல்பெயர்ப்பி இவற்றை அனுமானிப்பது மிகவும் எளிது.

(iv) வரையறை let என்ற சிறப்புச் சொல்லோடு அறிமுகப்படுத்தப்படுகிறது. அதனைத் தொடர்ந்து செயற்கூறின் பெயர் மற்றும் அதனுடைய செயலுருபுகள் பிறகு செயலுருபுவைக் கணக்கீடு செய்யும் வாய்ப்பாடு := குறிக்குப் பிறகு எழுத வேண்டும். தற்சுழற்சி செயற்கூற்றை வரையறுக்க விரும்பினால் let-க்கு பதிலாக let rec என்று பயன்படுத்த வேண்டும்.

செயற்கூறு வரையறையின் தொடரியல் :

```
let rec fn a 1 a2,,,an := k
```

2. (அ) gcd
(ஆ) let rec gcd a b : =
(இ) a,b
(ஈ) gcd b (a mod b)
(உ) if b < > 0 ல் உள்ள else return a

3. **Pure செயற்கூறுகள் :**

(i) ஒரே மாதிரியான செயலுருபுகளை அனுப்பும் போது, ஒரே மாதிரியான விடையைத் தரும் செயற்கூறு pure செயற்கூறுகள் ஆகும். எடுத்துக்காட்டாக, கணித செயற்கூறு sin(0)-ன் விடை எப்பொழுதுமே 0 ஆகும். இதன் அர்த்தம் என்னவென்றால், அதே அளபுருக்களைக் கொண்டு செயற்கூறினை ஒவ்வொரு முறையும் அழைக்கும் போது, அதே சரியான விடையை எப்பொழுதும் பெறலாம். மாறியின் பண்பை மாற்றக் கூடிய எந்த விதமான வெளிப்புற மாறியும் இல்லாமல் இருந்தால் அந்த செயற்கூறு pure செயற்கூறாகும்.

(எ.கா)

```
let square x: =
return: x * x
```

(ii) மேலேயுள்ள square செயற்கூறு pure செயற்கூறு ஆகும். ஏனென்றால் ஒரே மாதிரியான உள்ளீட்டிற்கு வெவ்வேறு விதமான வெளியீட்டைத் தராது.

(iii) கோட்பாடு சார்ந்த நன்மைகளை pure செயற்கூறுகள் கொண்டுள்ளன. ஒரு நன்மை என்னவென்றால், இதன் pure செயற்கூறாக இருக்கும் பொழுது, அதே அளபுருக்களுடன் செயற்கூறுவை பல தடவைகள் அழைக்கும்போது, உண்மையில் நிரல் பெயர்ப்பி செயற்கூறுவை மீண்டும் ஒரு தடவை அழைக்கும் தேவை மட்டுமே ஏற்படுகிறது.

```
let length s: =
  i: = 0
  if i <strlen (s) then
  -- Do something which doesn't affect s ++ i
```

(iv) இது இயக்கப்படும் போது, ஒவ்வொரு முறையும் strlen (s) அழைக்கப்படுகிறது strlen-க்கு ஒட்டு மொத்தமாக 's' சுழற்சி செய்ய தேவைப்படுகிறது.

(v) strlen என்பது pure செயற்கூறாகும்.

Impure செயற்கூறுகள்:

(i) செயற்கூறுக்கு அளபுருக்களை அனுப்பாதபோதும், செயற்கூறின் உள்ளே உள்ள மாறியானது பக்க விளைவுகளை ஏற்படுத்தும், இந்த

வகையான செயற்கூறை impure செயற்கூறு என்பர்.
(ii) ஒரு செயற்கூறு வரையறை தொகுதியின் வெளியே உள்ள மாறிகள் அல்லது செயற்கூறுகளைச் சார்ந்து இருந்து ஒவ்வொரு முறை அழைக்கும் பொழுதும் செயற்கூறு ஒரே மாதிரியாக இயக்கப்படும் என கூற இயலாது. எடுத்துக்காட்டாக. random() என்கிற கணித செயற்கூறு ஒரே மாதிரியான அழைப்புக்கூற்றுக்கு வெவ்வேறு விதமான வெளியீடுகளைக் கொடுக்கும்.

```
let a : random ( )
if a > 10 then
return:a
else
return:10
```

(iii) இங்கு Random என்பது impure செயற்கூறு ஆகும். ஏனெனில் இதனை அழைக்கும் பொழுது என்ன விடை கிடைக்கும் என நிச்சயமாக கூற இயலாது.

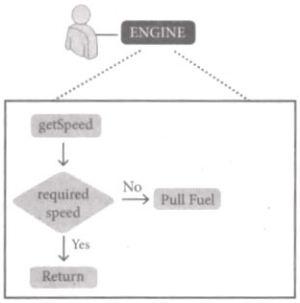
4. **(i) இடைமுகம் :**
ஒரு பொருள் (Object) செய்யக்கூடிய செயல்களின் தொகுப்பு இடைமுகம் ஆகும். எடுத்துக்காட்டாக, மின் விளக்கின் சுவிட்சை அழுத்தும் போது மின் விளக்கு ஒளிர்கிறது. அது எவ்வாறு ஒளிர்கிறது என்பது தெரிந்திருக்க வேண்டியதில்லை. பொருள் நோக்கு நிரலாக்க மொழியில், இடைமுகம் என்பது அனைத்து செயற்கூறுகளின் விவரிப்பு ஆகும். எடுத்துக்காட்டில், மின் விளக்கை போல் செயல்படும் எதுவும் turn_on () மற்றும் turn_off () என்ற செயற்கூறு வரையறையைக் கொண்டிருக்கும். இனக்குழுவின் பண்புக் கூறுகளை செயல்படுத்த இடைமுகம் பயன்படுகிறது.
(ii) இடைமுகம் மற்றும் செயல்படுத்துதல் இரண்டுக்கும் இடையே உள்ள வேறுபாடு என்னவெனில்,

இடைமுகம்	செயல்படுத்துதல்
ஒரு பொருள் செய்யக்கூடிய நடவடிக்கையை வரையறுக்கிறது, ஆனால் அவற்றை உண்மையில் செய்யக்கூடியது இல்லை.	இடைமுகத்தில் வரையறுக்கப்பட்டுள்ள கட்டளைகளை நிறைவேற்றுகிறது.

(iii) பொருள் நோக்கு நிரலாக்க மொழியில் இனக்குழு என்பது இடைமுகம் ஆகும். செயல்படுத்துதல் என்பது பொருள் (object) எவ்வாறு செயல்படுத்தப்பட்டு இயங்குகிறது என்பதை குறிப்பதாகும்.

இடைமுகத்தின் பண்புகள்

(i) ஒரு பொருளை முறையாக உருவாக்கி வழங்குவதற்கும் அதனை செயல்படுத்துவதற்கும் தேவையான இடைமுகத்தை இனக்குழு வார்ப்பு குறிப்பிடுகிறது.
(ii) செயற்கூறுகளைப் பொருளுக்கு அனுப்புவதன் மூலம் பொருளின் பண்புகளையும் பண்புக்கூறுகளையும் கட்டுப்படுத்த முடிகிறது. காரின் வேகத்தை அதிகரிக்கும் எடுத்துக்காட்டை எடுத்துக் கொள்வோம்.



QB 365 Study Material
QUESTION BANK 365

(iii) காரை ஓட்டும் நபர் அந்த காரின் உட்புற செயல்பாடுகள் பற்றி அறிந்திருக்க வேண்டிய அவசியமில்லை. காரின் வேகத்தை அதிகப்படுத்த, அவர் காரின் துரிதப்படுத்தியை (accelerator) அழுத்தி விரும்பிய பண்பை பெறுவார். இங்கு துரிதப்படுத்தி என்பது கார் ஓட்டுநருக்கும் (அழைக்கும் பொருள்) இயந்திரத்துக்கும் (அழைக்கப்படும் பொருள்) இடையேயான இடைமுகம் ஆகும். இதில், அழைக்கும் செயற்கூறு இவ்வாறு இருக்க வேண்டும் speed(70): என இருக்க வேண்டும். இது ஒரு இடைமுகமாகும்.
(iv) உட்புறமாக, காரின் இயந்திரம் அனைத்து வேலைகளையும் செய்கிறது. எரிபொருள், காற்று, அழுத்தம் மற்றும் மின்சாரம் ஆகியவை இங்கு ஒன்றாக சேர்ந்து, ஆற்றலை உருவாக்கி, வாகனத்தை நகர்த்துகிறது. இந்த அனைத்து நடவடிக்கைகளும் கார் ஓட்டுநரிடம் இருந்து பிரித்து வைக்கப்பட்டுள்ளது. அவர் வேகமாக காரை செலுத்துவதில் ஆர்வமாக இருப்பார். இதனால், இடைமுகத்தை செயல்படுத்துதலில் இருந்து பிரித்து வைக்கப்படுகிறது.
(v) ஒரு எளிமையான உதாரணத்தைக் காணலாம். கொடுக்கப்பட்டுள்ள 3 செயலுருபுகளில் குறைந்த மதிப்பைக் காணும் செயற்கூற்றைச் செயல்படுத்துவதைக் கவனி.

```
let min x y z:=
 if x < y then
        if x < z then x else z
else
        if y < z then y else z
```

3 Marks

5. **இடைமுகத்தின் பண்புகள்:**
(i) ஒரு பொருளை முறையாக உருவாக்கி வழங்குவதற்கும் அதனை செயல்படுத்துவதற்கும் தேவையான இடைமுகத்தை இனக்குழு வார்ப்பு குறிப்பிடுகிறது.
(ii) செயற்கூறுகளைப் பொருளுக்கு அனுப்புவதன் மூலம் பொருளின் பண்புகளையும் பண்புக்கூறுகளையும் கட்டுப்படுத்த முடிகிறது.
6. (i) strlen என்பது pure செயற்கூறாகும். ஏனென்றால், செயற்கூறு அளபுருவாக ஒரே ஒரு மாறியை எடுத்துக் கொண்டு அதனுடைய நீளத்தை கணக்கிடுகிறது. இந்த செயற்கூறு வெளி நினைவகத்தில் இருந்து உள்ளீட்டை எடுத்துக் கொள்கிறது.
(ii) ஆனால் மதிப்புகளை மாற்றுவதில்லை திருப்பி அனுப்பும் மதிப்புகள் வெளி நினைவகத்தில் இருந்து பெறப்பட்டதாகும்.
7. (i) செயற்கூறுக்கு அளபுருக்களை அனுப்பாத போதும், செயற்கூறின் உள்ளே உள்ள மாறியானது பக்க விளைவுகளை ஏற்படுத்தும். இந்த வகையான செயற்கூறை impure செயற்கூறு என்பர்.
(ii) ஒரு செயற்கூறு வரையறை தொகுதியின் வெளியே உள்ள மாறிகள் அல்லது செயற்கூறுகளைச் சார்ந்து இருந்து ஒவ்வொரு முறை

அழைக்கும் பொழுதும் செயற்கூறு ஒரே மாதிரியாக இயக்கப்படும் என கூற இயலாது.

(iii) **எடுத்துக்காட்டாக :** random ( ) என்கிற கணித செயற்கூறு ஒரே மாதிரியான அழைப்புக்கூற்றுக்கு வெவ்வேறு விதமான வெளியீடுகளைக் கொடுக்கும்.

```
let a : random ()
  if a > 10 then
return: a
else
  return:10
```

(iv) இங்கு, Random என்பது impure செயற்கூறு ஆகும். ஏனெனில் இதனை அழைக்கும் பொழுது என்ன விடை கிடைக்கும் என நிச்சயமாக கூற இயலாது.

8.

	pure செயற்கூறு	impure செயற்கூறு
1.	pure செயற்கூறுவின் திருப்பி அனுப்பும் மதிப்பு முற்றிலும் செயலுருபுகளை பொறுத்தே அமையும்.	impure செயற்கூறுவின் திருப்பி அனுப்பும் மதிப்பு முற்றிலும் செயலுருபுகளை பொறுத்து அமையாது,
2.	செயற்கூறினை அதே செயலுருபுகளை கொண்டு அழைத்தால் எப்பொழுதும் அதே திருப்பி அனுப்பும் மதிப்பே கிடைக்கும்.	Impure செயற்கூறினை அதே செயலுருபுகளை கொண்டு அழைத்தால் வெவ்வேறான திருப்பி அனுப்பும் மதிப்பு கிடைக்கும்.
3.	இது எந்த பக்கவிளைவுகளையும் கொண்டிருக்காது.	(எ.கா) random () date ()
4.	இந்த செயற்கூறு செயலுருபுகளை மாற்றம் செய்யாது.	இந்த செயற்கூறு செயலுருபுகளை மாற்றம் செய்யலாம்.

9. ஒரு செயற்கூறுக்கு வெளியே ஒரு மாறியை மாற்றினால் சில பக்க விளைவுகளை அடையாளம் காண்பது எளிதானதாகும் மற்றும் சில யோசிக்க தகுந்ததாக இருக்கும்.

எடுத்துக்காட்டு :

```
y:= 0
let inc(x:int):int : =
y:=y + x
return (y)
```

இங்கு y என்பது வெளிப்புற மாறியாகும். இது செயற்கூற்றில் பயன்படுத்தப்பட்டுள்ளது. இந்த வெளிப்புற மாறி ஒவ்வொரு முறை அழைக்கப்படும் போதும் தனது மதிப்பை மாற்றிக் கொள்வதில் inc() செயற்கூறின் திருப்பி அனுப்பும் விடையானது வெளிப்புற மாறியின் பக்கவிளைவினால் ஒரே மதிப்பை கொண்டு அழைத்தாலும் ஒவ்வொரு முறையும் மாறும்.

2 Marks

10. துணை நிரல்கள் கணினி மொழிகளின் அடிப்படைக் கட்டுமான தொகுதியாக விளங்குகின்றன. துணை நிரல்கள் என்பன ஒரு குறிப்பிட்ட செயலை மீண்டும் மீண்டும் செய்யப் பயன்படும் சிறிய நிரல் தொகுதியாகும். நிரலாக்க மொழிகளில் இத்துணை நிரல்கள் செயற்கூறுகள் (Functions) என்று அழைக்கப்படுகின்றன.
11. செயற்கூறு என்பது ஒரு பெரிய நிரலில் வரையறுக்கப்படும் ஒரு குறிமுறை அலகு ஆகும். செயற்கூறுகள், நிலையான வெளியீட்டை வழங்குவதற்கான மாறிகள், கோவைகள் போன்ற பலவகையான உள்ளீடுகளின் மீது செயல்படும் குறிமுறை தொகுதிகளை கொண்டிருக்கும்.
12. (i) x:=78 ஒரு செயற்கூறு வரையறை ஆகும். வரையறைகள், மதிப்புகளைப் பெயருடன் பிணைக்கின்றன.
(ii) இங்கு 78 என்ற மதிப்பு 'X' என்ற பெயருடன் பிணைக்கப்படுகின்றது. வரையறைகள் கோவைகள் அல்ல. அதே நேரத்தில் கோவைகளை வரையறை எனக் கருதக் கூடாது. வரையறைகள் தனித்தன்மையான தொடரியல் தொகுதிகளைக் கொண்டதாகும்.
(iii) வரையறைகள் உள்ளமைவாக உள்ள கோவைகளைக் கொண்டதாகவோ அல்லது நேர்மாறாகவும் இருக்கலாம்.
13.

இடைமுகம்	செயல்படுத்துதல்
ஒரு பொருள் செய்யக்கூடிய நடவடிக்கையை வரையறுக்கிறது, ஆனால் அவற்றை உண்மையில் செய்யக் கூடியது இல்லை.	இடைமுகத்தில் வரையறுக்கப்பட்டுள்ள கட்டளைகளை நிறைவேற்றுகிறது.

14. (i) தற்சுழற்சி செயற்கூறு
(ii) சாதாரண செயற்கூறு
(iii) தற்சுழற்சி செயற்கூறு

Multiple Choice Question

15. (a) துணை நிரல்கள்
16. (b) செயற்கூறு
17. (c) வரையறை
18. (c) அளப்புருக்கள்
19. (a) செயலுருபுகள்
20. (b) ( )
21. (c) இடைமுகம்
22. (c) செயல்படுத்துதல்
23. (d) Pure செயற்கூறு
24. (a) Impure செயற்கூறு