Προγραμματισμός I (ECE115)

#5 εκτέλεση υπό συνθήκη / διακλαδώσεις

Χρήστος Αντωνόπουλος
cda@uth.gr

Ροή εκτέλεσης

- Μέχρι στιγμής η ροή εκτέλεσης του κώδικα στα διάφορα παραδείγματα είναι καθαρά **γραμμική**
- Οι εντολές που γράφουμε στον κώδικα μας εκτελούνται μια προς μια και από πάνω προς τα κάτω

- Η ροή εκτέλεσης είναι **προκαθορισμένη**
- Θα εκτελεστούν όλες οι εντολές του προγράμματος με την σειρά που τις έχουμε γράψει
  - κάθε φορά που τρέχουμε το πρόγραμμα

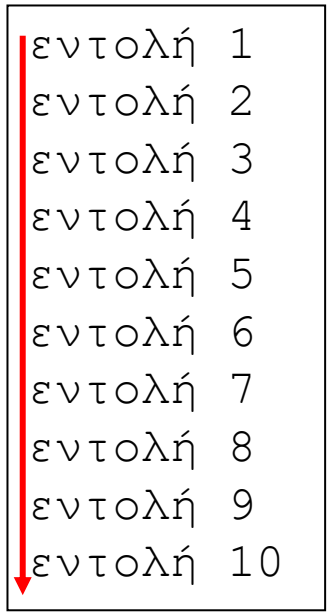
Δομές/εντολές ελέγχου ροής εκτέλεσης

- Μπορούμε να φτιάξουμε προγράμματα στα οποία η ροή εκτέλεσης **δεν** είναι προκαθορισμένη

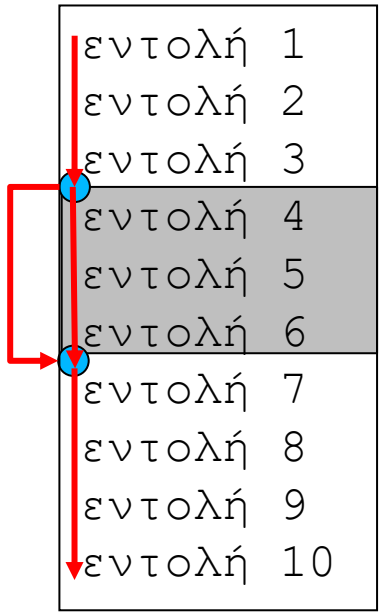
- Αυτό γίνεται με τις **δομές ελέγχου** ροής εκτέλεσης
  - εκτέλεση υπό συνθήκη («διακλαδώσεις» / branches)
  - εκτέλεση σε επανάληψη («βρόγχοι» / loops)

- Το πρόγραμμα παίρνει **«δικές του» αποφάσεις** σχετικά με την εκτέλεση (ή μη) κάποιων εντολών
- Οι αποφάσεις λαμβάνονται **την ώρα της εκτέλεσης**
  - ανάλογα με τις τιμές που έχουν οι μεταβλητές

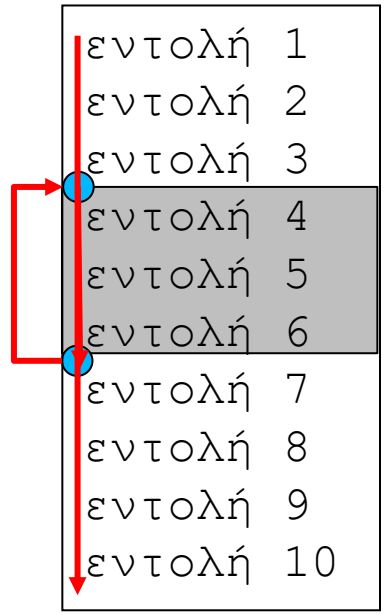
γραμμική εκτέλεση



εκτέλεση με διακλάδωση

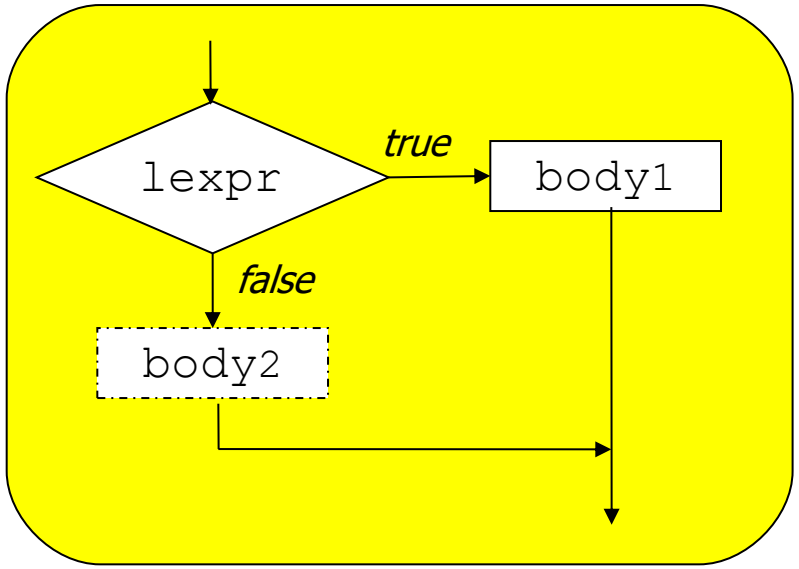


εκτέλεση με επανάληψη



Εκτέλεση υπό συνθήκη: `if-else`

```
if (<lexpr>)
   <body1>
else
   <body2>
```



- Αποτιμάται η λογική συνθήκη `<lexpr>`
  - αν είναι αληθής (`!=0`), η εκτέλεση συνεχίζεται με το `<body1>`
  - διαφορετικά, η εκτέλεση συνεχίζεται με το `<body2>` (αν υπάρχει)
- Το εναλλακτικό σκέλος `else` είναι προαιρετικό
  - αν δεν υπάρχει `else` και η συνθήκη `<lexpr>` αποτιμηθεί σε `0`, απλά παρακάμπτεται το `<body1>` (η εκτέλεση συνεχίζεται μετά το `if`)

```c
/* απόλυτη τιμή ακεραίου */

#include <stdio.h>

int main(int argc, char *argv[]) {
  int val, abs;

  scanf("%d", &val);

  if (val >= 0) {
    abs = val;
  }
  else {
    abs = -val;
  }

  printf("%d\n", abs);

  return(0);
}
```

```c
/* τετραγωνική ρίζα ακεραίου με έλεγχο εισόδου */

#include <stdio.h>
#include <math.h>

int main(int argc, char *argv[]) {
  int x, res;

  printf("Enter non-negative integer: ");
  res = scanf("%d", &x);

  if ((res != 1) || (x < 0)) {
    printf("wrong input\n");
  }
  else {
    printf("sqrt(%d)=%lf\n", x, sqrt(x));
  }

  return(0);
}
```

μετάφραση με `-lm` στο τέλος

```c
/* μέγιστη τιμή τριών ακεραίων */

#include <stdio.h>

int main(int argc, char *argv[]) {
  int val1, val2, val3, max;

  scanf("%d %d %d", &val1, &val2, &val3);

  if (val1 > val2) {
    if (val1 > val3) {
      max = val1;
    }
    else {      /* val1 <= val3 */
      max = val3;
    }
  }
  else {        /* val2 >= val1 */
    if (val2 > val3) {
      max = val2;
    }
    else {      /* val2 <= val3 */
      max = val3;
    }
  }
  printf("%d\n", max);
  return(0);
}
```

```c
/* μέγιστη τιμή τριών ακεραίων */

#include <stdio.h>

int main(int argc, char *argv[]) {
  int val1, val2, val3, max;

  scanf("%d %d %d", &val1, &val2, &val3);

  max = val1;
  if (max < val2) {
    max = val2;
  }
  if (max < val3) {
    max = val3;
  }

  printf("%d\n", max);
  return(0);
}
```

Σώμα εντολών

- Το σώμα μιας εντολής ελέγχου (π.χ. του `if-else`) δίνεται **υποχρεωτικά** ανάμεσα σε `{}` όταν αυτό αποτελείται από πολλές εντολές
- Οι `{}` είναι **προαιρετικές μόνο** όταν το σώμα αποτελείται από μια μοναδική εντολή
  - ένας άπειρος προγραμματιστής καλό είναι να χρησιμοποιεί πάντα `{}`, ακόμα και για σώματα που περιέχουν μια εντολή

- Η χρήση `{}`, μορφοποίηση του κώδικα με λευκούς χαρακτήρες και κατανομή σε ξεχωριστές γραμμές δεν είναι απλά θέμα «γούστου»
  - **επηρεάζει** σημαντικά την αναγνωσιμότητα

```
/* απόλυτη τιμή ακεραίου */

#include <stdio.h>

int main(int argc, char *argv[]) {
  int val, abs;

  scanf("%d", &val);

  if (val >= 0) {
    abs = val;
  }
  else {
    abs = -val;
  }

  printf("%d\n", abs);

  return(0);
}
```

```c
/* απόλυτη τιμή ακεραίου */

#include <stdio.h>

int main(int argc, char *argv[]) {
  int val, abs;

  scanf("%d", &val);

  if (val >= 0)
    abs = val;
  else
    abs = -val;

  printf("%d\n", abs);

  return(0);
}
```

```
/* απόλυτη τιμή ακεραίου */

#include <stdio.h>

int main(int argc, char *argv[]) {
  int val, abs;

  scanf("%d", &val);

  if (val >= 0) abs = val; else abs = -val;

  printf("%d\n", abs);

  return(0);
}
```

δυσανάγνωστο

```c
/* εκτύπωση ημέρας με βάση κωδικό */

#include <stdio.h>

int main(int argc, char *argv[]) {
  int day;

  scanf("%d", &day);

  if      (day == 1) { printf("Mon\n"); }
  else if (day == 2) { printf("Tue\n"); }
  else if (day == 3) { printf("Wed\n"); }
  else if (day == 4) { printf("Thu\n"); }
  else if (day == 5) { printf("Fri\n"); }
  else if (day == 6) { printf("Sat\n"); }
  else if (day == 7) { printf("Sun\n"); }
  else { printf("wrong code %d\n",day); }

  return(0);
}
```

δυσανάγνωστο

```
if      (day == 1) { printf("Mon\n"); }
else if (day == 2) { printf("Tue\n"); }
else if (day == 3) { printf("Wed\n"); }
else if (day == 4) { printf("Thu\n"); }
else if (day == 5) { printf("Fri\n"); }
else if (day == 6) { printf("Sat\n"); }
else if (day == 7) { printf("Sun\n"); }
else { printf("wrong code %d\n",day); }
```

δυσανάγνωστο

```
if      (day == 1) printf("Mon\n");
else if (day == 2) printf("Tue\n");
else if (day == 3) printf("Wed\n");
else if (day == 4) printf("Thu\n");
else if (day == 5) printf("Fri\n");
else if (day == 6) printf("Sat\n");
else if (day == 7) printf("Sun\n");
else printf("wrong code %d\n",day);
```

σαφώς καλύτερα

```
if (day == 1)
  printf("Mon\n");
else if (day == 2)
  printf("Tue\n");
else if (day == 3)
  printf("Wed\n");
else if (day == 4)
  printf("Thu\n");
else if (day == 5)
  printf("Fri\n");
else if (day == 6)
  printf("Sat\n");
else if (day == 7)
  printf("Sun\n");
else
  printf("wrong code %d\n",day);
```

έμφαση στην δομή – αλλά πιο δυσανάγνωστο

```
if (day == 1) printf("Mon\n");
else
  if (day == 2) printf("Tue\n");
  else
    if (day == 3) printf("Wed\n");
    else
      if (day == 4) printf("Thu\n");
      else
        if (day == 5) printf("Fri\n");
        else
          if (day == 6) printf("Sat\n");
          else
            if (day == 7) printf("Sun\n");
            else
              printf("wrong code %d\n",day);
```

```
if (day == 1) {
  printf("Mon\n");
}
else {
  if (day == 2) {
    printf("Tue\n");
  }
  else {
    if (day == 3) {
      printf("Wed\n");
    }
    else {
      if (day == 4) {
        printf("Thu\n");
      }
      else {
        if (day == 5) {
          printf("Fri\n");
        }
        else {
          if (day == 6) {
            printf("Sat\n");
          }
          else {
            if (day == 7) {
              printf("Sun\n");
            }
            else {
              printf("wrong code %d\n",day);
            }
          }
        }
      }
    }
  }
}
```

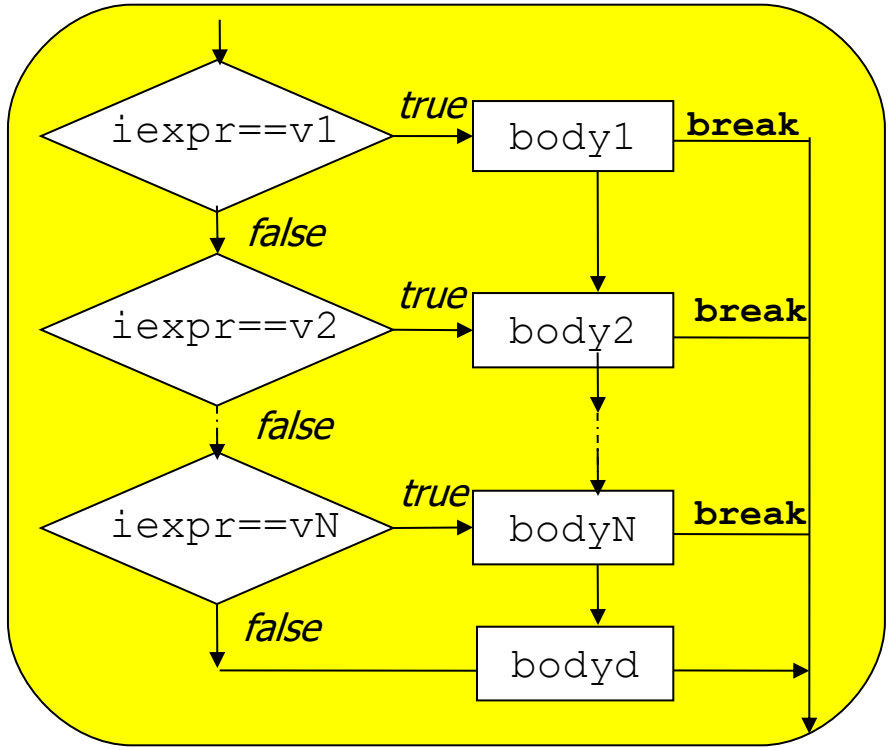
λίγο καλύτερο από το προηγούμενο αλλά πάλι αρκετά δυσανάγνωστο

```
if (day == 1) {
  printf("Mon\n");
}
else if (day == 2) {
  printf("Tue\n");
}
else if (day == 3) {
  printf("Wed\n");
}
else if (day == 4) {
  printf("Thu\n");
}
else if (day == 5) {
  printf("Fri\n");
}
else if (day == 6) {
  printf("Sat\n");
}
else if (day == 7) {
  printf("Sun\n");
}
else {
  printf("wrong code %d\n",day);
}
```

ένας καλός συμβιβασμός ως προς την ανάδειξη της δομής και την αναγνωσιμότητα

Εκτέλεση υπό συνθήκη: `switch-case`

```
switch (<iexpr>) {
  case <v1>: <body1>
  case <v2>: <body2>
  …
  case <vN>: <bodyN>
  default:   <bodyd>
}
```



- Υπολογίζεται η ακέραια αριθμητική έκφραση `iexpr` και επιλέγεται προς εκτέλεση το σώμα της πρώτης περίπτωσης με την ίδια τιμή
- Όσο **δεν** συναντάται `break` η εκτέλεση συνεχίζεται με το επόμενο σώμα
- Το σκέλος default είναι προαιρετικό
  - επιλέγεται αν δεν επιλεγεί άλλη περίπτωση

```c
/* εκτύπωση ημέρας με βάση κωδικό */

#include <stdio.h>

int main(int argc, char *argv[]) {
  int day;

  scanf("%d", &day);

  switch (day) {
    case 1: { printf("Mon\n"); break; }
    case 2: { printf("Tue\n"); break; }
    case 3: { printf("Wed\n"); break; }
    case 4: { printf("Thu\n"); break; }
    case 5: { printf("Fri\n"); break; }
    case 6: { printf("Sat\n"); break; }
    case 7: { printf("Sun\n"); break; }
    default:{ printf("wrong day code %d\n",day); }
  }
  return(0);
}
```

χωρίς την εντολή `break` το πρόγραμμα θα **συνέχιζε** την εκτέλεση με το **σώμα** της επόμενης περίπτωσης

```
switch (day) {
  case 1: {
    printf("Mon\n");
    break;
  }
  case 2: {
    printf("Tue\n");
    break;
  }
  case 3: {
    printf("Wed\n");
    break;
  }
  …
  case 7: {
    printf("Sun\n");
    break;
  }
  default:{
    printf("wrong day code %d\n",day);
  }
}
```

καλύτερη μορφοποίηση (ιδίως αν τα σώματα των περιπτώσεων αποτελούνται από πολλές εντολές)

Τα άγκιστρα ΔΕΝ είναι καν απαραίτητα

```c
/* εκτύπωση όλων των ημερών μετά από μια ημέρα */

#include <stdio.h>

int main(int argc, char *argv[]) {
  int day;

  scanf("%d", &day);

  switch (day) {
    case 1: printf("Mon\n");
    case 2: printf("Tue\n");
    case 3: printf("Wed\n");
    case 4: printf("Thu\n");
    case 5: printf("Fri\n");
    case 6: printf("Sat\n");
    case 7: { printf("Sun\n"); break; }
    default: printf("wrong day code %d\n",day);
  }
  return(0);
}
```

γιατί χρειάζεται αυτό;