Örneklerle: C/C++ Giriş

Cebirdeki Düşünce Biçimi

- Amaç
  - Geliştirici ne ister ?
  - Geliştirici ne yapar ?
- Sıralı Programlama
  - Telefon defteri uygulaması
- Nesneye Yönelmek
  - Karmaşadaki Sadelik
  - ÇokGen'den türeyen ÜçGen ve DörtGen
- Çizgiden Türeyen Ok
  - Cizgi class'ı ve uygulaması
  - Ok class'ı ve uygulaması
  - Kullanıcı programındaki kullanımı
  - Kullanıcı programında işaretçi (pointer) ile kullanımı
- Kaynak (code), Öbek (heap) ve Yığın (stack)
  - Yazılımların kullandığı hafıza bölgeleri
- Düşünmek Derken
  - Akış ve UML çizelgeleri
- Temel Programlama İfadeleri
  - Döngüler ve şart koşma
  - Rastlantısal sayılar ile π sayısının hesaplanması
  - π sayısı hesabı için FORTRAN ve C uygulaması ile akış çizelgesi
- Kütüphane Derlemek ve Kullanmak
  - Cizgi ve ok class' larını libCizgiVeOk.so kütüphanesine derlemek ve kullanmak
- ROOT' tan Bahis...
  - Kurulum ve “Merhaba dünya !!” alıştırması
- Proje Ödevinden Bahis...
  - Çift RPC (resistive plate chamber) deneyi ile kozmik ışınların yeniden oluşturulması
- Akşam Sefası
  - Ertesi günün sabahına hazırlanması beklenen akşamlık ödevler

Amaç

Geliştirici Ne İster ?

Bir yazılımın verimli çalışmasından daha önemli olan şeyler var mıdır ?***

parçalanabilirlik (modularity)
kullanıcı için kolaylık (user-friendliness)
doğruluk (correctness)
geliştirici mesaii (programmer time)
bakım kolaylığı (maintainability)
sadelik (simplicity)
işlevsellik (functionality)
genişletilebilirlik (extensibility)
sağlamlık (robustness)
güvenilirlik (reliability)
?
?
?

*** *Prof. Erik D. Demaine' in derslerinden yararlanildi*

Temelde

Geliştirici Ne Yapar ?

Ne ?

Sorunu...

anlamak için *BÖL,*
bölünmüş ve küçük olanı *ANLA,*
anladığına *HAKiM OL,*
hakimiyetin altındakileri *BİRLEŞTİR = Çözüm*

Temelde nasıl ?

Seç...

Sıralı programlama (procedural)
Nesne yönelimli programlama (object-oriented)
Bakış açısı yönelimli programlama (aspect-oriented)
...

Temelde

Geliştirici Ne Yapar ?

(sanırım)

Yazmaya başlamadan önce:

Düşünür ve algoritmasını kalemle kağıda çizer (mantık akışı)
Düşünür ve kağıt üstünde çalıştığından emin olur
Düşünür ve sonra yazmaya başlar

(bence)

*Programlama = Düşünmek = Tasarım (**Seçkin**)*
*Kod Yazmak = Ameliye (**Sıradan**)*

***Dilden bağımsız** (Türkçe/Fransızca/C/C++/Perl v.b.)*
***Hataya en az açık** yöntem*
*Yazılım büyüdükçe **üstel artan bakım güçlüğü***

(mutlak doğru)

Hiçbir programlama dili, kütüphane ya da araç her zaman en iyi çözüm değildir

Dogru Kararı Başlangıçta Verin

Büyüdükçe üstel artan bakım güçlüğü derken...

Sıralı Programlama

*Örnek: Telefon Defteri Uygulaması****

- Yükle işlevinin uygulaması:
  - Pointer yarat (*di* ve *al*)
  - Değişken yarat (*tane*, *k* ve *l*)
  - Kütük okunabiliyor mu (*telDef.dat*)
    - Okunabiliyor ise kütüğün sonuna git ve boyunu ölç
  - Ölçülmüş bu boyut, KAYIT isimli struct' ın kaç katı ?
  - Kayıt sayısı kadar dönecek bir öngü ile tüm kayıtları oku
  - Okunan tüm kayıtları dinamik bağlı liste oluşturacak şekilde birbirine bağla (*dugumEkle* işlevi çağırılıyor)
- Ana program:
  - Menüden yapılan seçime göre *kayıt, arama, sıralama, silme, saklama ve yükleme* işlevlerinden birini çağır.
  - Çıkış (case 7=seçenek 7) seçilmiş ise uygulamadan çık (return 0).

**** Benzer bir örnek Rifat Çölkesen' in "işte C programlama" adlı kitabında, dinamik bağlı liste başlığında verilmistir.*

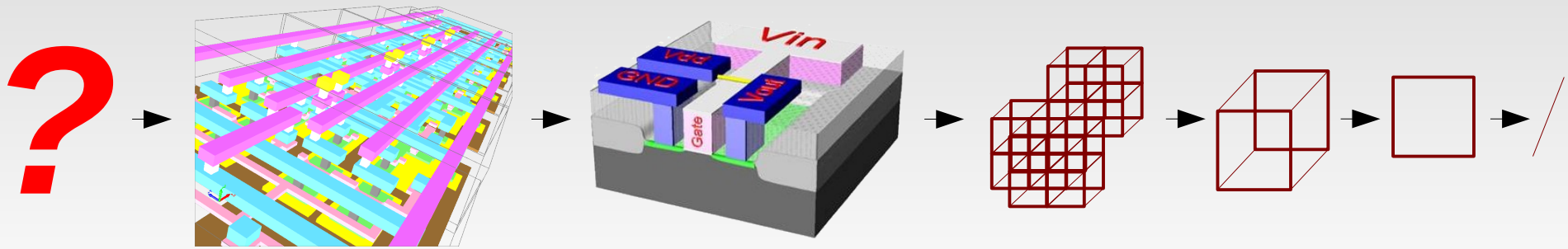
File Edit Search Preferences Shell Macro Windows Help

```
207 // yukle()
208 //_____________________________________________________________
209 void yukle() {
210     FILE *di;
211     KAYIT *al;
212     unsigned int tane;
213     int k, i;
214     if ((di=fopen("telDef.dat","r"))==NULL) {
215         printf("\nHATA : Kütük açilamadi !...\n");
216         return;
217     }
218     fseek(di, 0, 2);
219     tane=ftell(di)/(sizeof(KAYIT)-2*sizeof(al));
220     fseek(di, 0, 0);
221     if (tane<1) {
222         printf("\nKütük bos !..\n");
223         return;
224     }
225     printf("\nKütükten yükleniyor..\n");
226     for (int k=0 ; k<tane ; k++) {
227         al=(KAYIT*)malloc(sizeof(KAYIT));
228         if (!al) {
229             printf("\nHATA : Fiziksel hafiza dolu !..\n");
230             break;
231         }
232         fread(al, sizeof(KAYIT)-2*sizeof(al), 1, di);
233         al->sol=NULL;
234         al->sag=NULL;
235         dugumEkle(kok, al);
236     }
237     fclose(di);
238 }
239
240 // main()
241 //_____________________________________________________________
242 int main() {
243     int secim;
244     while(1) {
245         secim = menudenSecimYap();
246         switch(secim) {
247             case 1 :
248                 kayit();
249                 break;
250             case 2 :
251                 arama();
252                 break;
253             case 3 :
254                 agacListele(kok);
255                 break;
256             case 4 :
257                 silme();
258                 break;
259             case 5 :
260                 sakla();
261                 break;
262             case 6 :
263                 yukle();
264                 break;
265             case 7 :
266                 return 0;
267         }
268     }
269 return 0;
270 }
271
```

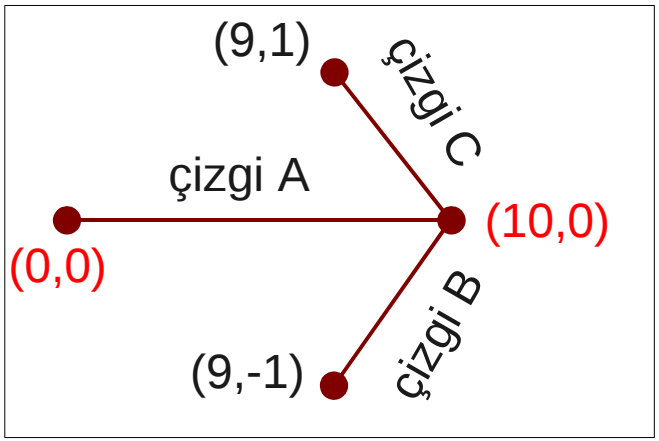
Nesneye Yönelmek

Karmaşadaki sadelik

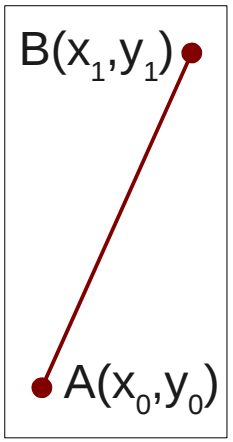
- Karmaşık mimariler ancak nesne yönelimli anlayış kullanılarak oluşturulabilir
- En verimli (*yani başımızı en az ağrıtacak*) yazılım geliştirme şeklidir



- Bir *ok* yapmak istediğimizde, aslında programlamamız gereken tek şey bir *çizgi*dir; *ok*, üç tane *çizgi* ile oluşturulabilecek bir *nesne*dir.



Class ***ok***



Class ***cizgi***

Örnek

ÇokGen' den türeyen ÜçGen ve DörtGen

- Bir çokgen, genişlik ve yükseklik ile tanımlanabilir.
- Dörtgen ve üçgen birer çokgendir.
- *Yükseklik* ve *genişlik* bilgisini tutacak olan *Çokgen* class' ını yaz ve bu bilgiyi miras alacak (inherit) olan *üçgen* ile *dörtgen* class' larını ondan türet (derive).
- Genişlik ve yükseklikleri sırasıyla 4 ve 5 olan bir üçgen ile bir dörtgen yarat

- Üçgen ve dörtgen, DegerAyarla() işlevine sahip değil ancak türedikleri ÇokGen class' ı sahip (base class).
- Alan() işlevine ÇokGen class' ı sahip olmasa da hem Üçgen hem Dörtgen sahip.

File Edit Search Preferences Shell Macro Windows Help

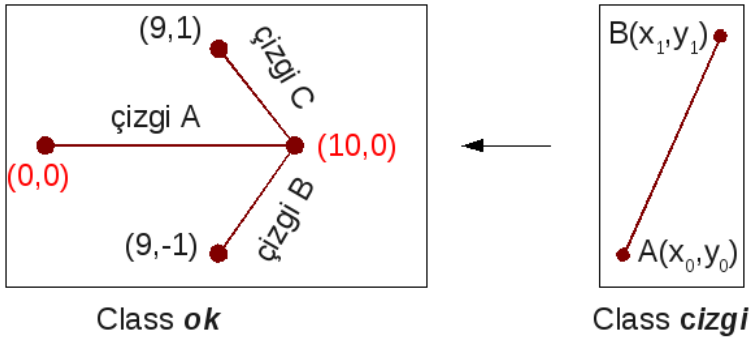
gen.cxx | cikti

```
 1 #include <iostream>
 2 using namespace std;
 3
 4 class CokGen {
 5   protected:
 6     int genislik, yukseklik;
 7   public:
 8     void DegerAyarla (int a, int b)
 9       { genislik=a; yukseklik=b;}
10   };
11
12 class Dortgen: public CokGen {
13   public:
14     int Alan ()
15       { return (genislik * yukseklik); }
16   };
17
18 class Ucgen: public CokGen {
19   public:
20     int Alan ()
21       { return (genislik * yukseklik / 2); }
22   };
23
24 int main () {
25   Dortgen BenimGuzelDortgenim;
26   Ucgen   BenimGuzelUcgenim;
27
28   BenimGuzelDortgenim.DegerAyarla (4,5);
29   BenimGuzelUcgenim.DegerAyarla (4,5);
30
31   cout << BenimGuzelDortgenim.Alan() << endl;
32   cout << BenimGuzelUcgenim.Alan() << endl;
33
34   return 0;
35 }
36
```

Örnek

Ok – cizgi class' ı (header)

- Bir çizgi farklı şekillerde tanımlanabilir:
  - İki nokta
  - Nokta, yön, uzunluk
  - Vektör
  - v.b.
- Yandaki örnekte iki nokta kullanılmıştır



Class ***ok*** Class ***cizgi***

File Edit Search Preferences Shell Macro Windows Help

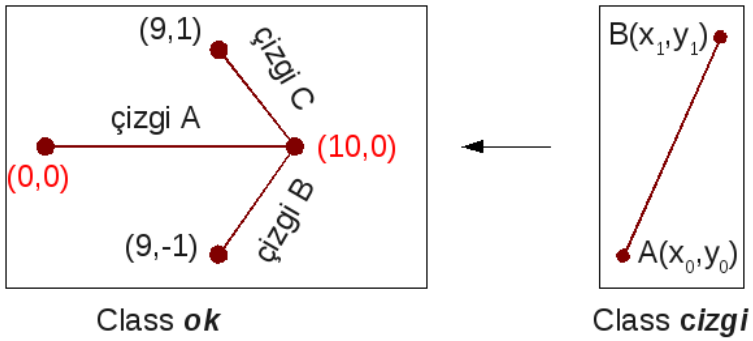
ok.cxx | ok.h | **cizgi.h** | cizgi.cxx | anaProgram.c | cikti

```
 1 #ifndef CIZGI_H
 2 #define CIZGI_H
 3
 4 #include <math.h>
 5 #include <iostream>
 6
 7 //_____________________________________________________________
 8 class cizgi
 9 {
10     private:
11         float x0, y0;      // ilk nokta
12         float x1, y1;      // ikinci nokta
13
14     public:
15         void degerAta(float a,
16                       float b,
17                       float c,
18                       float d);   // noktalari gir
19         virtual void bas();       // noktalari goster
20         virtual float uzunluk();  // uzunlugunu dondurur
21 };
22
23 #endif
24
```

Örnek

Ok – ok class' ı *(header)*

- Bir ok, üç çizgi ile oluşturulabilir
  - Bir ana çizgi
  - Okun ucunu oluşturan 2 çizgi daha
- Okun ucunu oluşturan iki çizginin, ana çizgiden ne kadar uzaklasacağı ve boyunun yine ana çizginin kaçta kacı olacağı okGenis değişkeni (private variable) ile belirlenecek



Class ***ok*** Class ***cizgi***

File Edit Search Preferences Shell Macro Windows Help

ok.cxx | **ok.h** | cizgi.h | cizgi.cxx | anaProgram.c | cikti

```
 1 #ifndef OK_H
 2 #define OK_H
 3
 4 #include "cizgi.h"
 5
 6 //_____________________________________________________________
 7 class ok : public cizgi
 8 {
 9     private:
10         cizgi A;            // Okun ana cizgisi
11         cizgi B, C;         // Ana cizginin iki yanindakiler
12         float okGenis;      // okun genisligi
13
14     public:
15         void degerAta(float a,
16                       float b,
17                       float c,
18                       float d);    // noktalari gir
19         virtual void  bas();       // noktalari goster
20         virtual float uzunluk();   // toplam uzunluk
21 };
22
23 #endif
24
```

Örnek

Ok – cizgi class' ının uygulaması (implementation)

- Tüm üye işlevler (*member function* ya da *method*) class' ın private değişkenlerine erişim hakkına sahiptir:
  - degerAta()
  - uzunluk()
  - bas()

File Edit Search Preferences Shell Macro Windows Help

ok.cxx | ok.h | cizgi.h | **cizgi.cxx** | anaProgram.c | cikti

```
 1 #include "cizgi.h"
 2
 3 //_____________________________________________________________
 4 void cizgi::degerAta(float a, float b, float c, float d)
 5 {
 6 // Duzlemde tanimlanan iki noktanin koordinatlari
 7 // sadece class'a ait islevlerin (method ya da
 8 // member function) erisim hakki bulunan
 9 // degiskenlere ataniyor.
10
11     if (a==c && b==d) {
12         printf("HATA: Bu bir nokta ! \n");
13     } else {
14         x0 = a;                    // 1. noktanin x ordinati
15         y0 = b;                    // 1. noktanin y ordinati
16         x1 = c;                    // 2. noktanin x ordinati
17         y1 = d;                    // 2. noktanin y ordinati
18     }
19 }
20
21 //_____________________________________________________________
22 float cizgi::uzunluk()
23 {
24 // Cizginin uzunlugunu dondurur.
25
26     return sqrt( (y1-y0)*(y1-y0) + (x1-x0)*(x1-x0) );
27 }
28
29 //_____________________________________________________________
30 void cizgi::bas()
31 {
32 // Cizgiyi olusturan noktalarin koordinatlarini dondurur
33
34     printf("1. Nokta = (%f, %f) \n", x0, y0);
35     printf("2. Nokta = (%f, %f) \n", x1, y1);
36 }
37
```

Örnek

Ok – ok class' ının uygulaması (implementation)

- Tekerleği yeniden icat etmeyin !! Zaten varolanları kullanın:
  - Sıfırdan bir *ok* yapmak yerine zaten yapılmış çizgileri kullan (degerAta() işlevi)
  - Tek tek her bir noktayı hesaplamak yerine, oku oluşturan çizgilerin her birine hangi noktalardan oluştuklarını sor (bas() işlevi)
  - Tek tek her bir çizginin uzunluğunu hesaplamak yerine bu işi, oku oluşturan çizgilere yaptır

File Edit Search Preferences Shell Macro Windows Help

ok.cxx | ok.h | cizgi.h | cizgi.cxx | anaProgram.c | cikti

```
 1 #include "ok.h"
 2
 3 //_____________________________________________________________
 4 void ok::degerAta(float a, float b, float c, float d)
 5 {
 6 // Duzlemde tanimlanan iki nokta
 7 // arasinda bir ok cizer.
 8
 9         A.degerAta( a,          b,          c, d );
10         okGenis = A.uzunluk() / 10.0;
11         B.degerAta( c-okGenis, d-okGenis, c, d );
12         C.degerAta( c-okGenis, d+okGenis, c, d );
13 }
14
15 //_____________________________________________________________
16 void ok::bas()
17 {
18 // Oku olusturan noktalarin koordinatlarini dondurur
19
20     printf("Ana cizgi:\n");
21     A.bas();
22     printf("Ust cizgi:\n");
23     B.bas();
24     printf("Alt cizgi:\n");
25     C.bas();
26 }
27
28 //_____________________________________________________________
29 float ok::uzunluk()
30 {
31 // Cizginin uzunlugunu dondurur.
32
33     return (  A.uzunluk() +
34               B.uzunluk() +
35               C.uzunluk()    );
36 }
37
```

Örnek

Ok – Kullanıcı uygulaması

- Class' ların başlık (header, *.h) ve uygulamalarını (implementation, *.cxx) yazdıktan sonra bu class kütüphanesini (library) kendi programımızda kullanacağız:
  - anaProgram.cxx

- (0,5) ve (10,5) noktaları arasında bir ok çizmek, oku oluşturan noktaları okumak ve oku oluşturan çizgilerin toplam uzunluğunu okumak istiyoruz.

- BenimGuzelOkum nesnesinin üye işlevlerine (member function) nokta (“.”) işlemcisi (ya da operatörü) ile ulaşılıyor

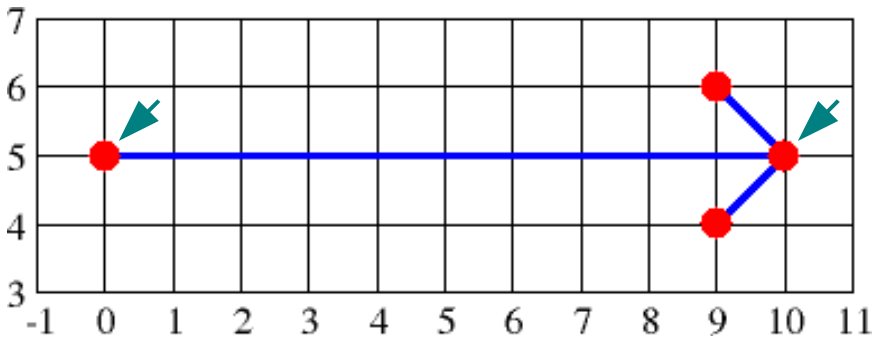
- BenimGuzelOkum nesnesi, ***yığın***' da (stack) oluşturuluyor

File Edit View Search Tools Documents Help

New Open Save Print... Undo Redo Cut Copy

anaProgram.cxx | anaProgram.cxx | cizgi.cxx

```
1  #include "ok.h"
2
3  //_____________________________________________________
4  int main () {
5
6      // Ok nesnesi "BenimGuzelOkum"u yarat
7      ok BenimGuzelOkum;
8
9      // Okun hangi iki nokta arasinda olacagi
10     // Uye islevlere (member function ya da method)
11     // nokta (".") isareti ile ulasiliyor:
12     // nesne.islev(imza) biçiminde.
13     BenimGuzelOkum.degerAta(0.0, 5.0, 10.0, 5.0);
14
15     //Oku olusturan noktalari goster
16     BenimGuzelOkum.bas();
17
18     // Oku olusturan cizgilerin toplam uzunlugunu goster
19     printf("Uzunluk = %f \n", BenimGuzelOkum.uzunluk());
20
21     // Programdan basari ile cik
22     return 0;
23 }
24
```



Ln 12, Col 36 INS

Örnek

Ok – İşaretçi (pointer) ile...

- Aynı kütüphane (*ok* class'ı) ve benzer kullanıcı isteği
- Bu kullanımda "*ok*" nesnesi, "***new***" işlemcisi (ya da operatörü) ile bir işaretçi (pointer) olarak yaratılıyor (14. satır)
- "*BenimGüzelOkum*" artık bir işaretçi olduğu için, üye işlevlerine (member function) ulaşmak için nokta ("**.**") yerine *işaretçi işlemcisi* ("**->**") kullanılıyor (19., 22. ve 25. satırlar)
- Kullanımdan sonra nesne *siliniyor* ve işgal ettiği hafıza özgür bırakılıyor (30. satır); bu işlemin gerçekleşmemesi durumunda *hafıza kaçağı* (memory leak) meydana gelir.
- BenimGüzelOkum nesnesi ***öbek***' te (heap) oluşturuluyor

File Edit View Search Tools Documents Help

New Open Save Print... Undo Redo Cut Copy

anaProgram.cxx | cizgi.cxx | cizgi.h

```
1  #include "ok.h"
2
3  //_____________________________________________________
4  int main () {
5
6      // Bir "ok" nesnesine isaret eden ve ismi
7      // "BenimGuzeOkum" olan bir isaretci (pointer)
8      // yarat. Bu isaretcinin isaret ettigi hafiza
9      // bolgesi, ok class'inda tanimlanan degiskenlerin
10     // toplam buyuklugu kadardir ve bu degisken icin
11     // ozel olarak ayrilmistir; kullanimi sona erdikten
12     // sonra tekrar "serbest" birakilmalidir ya da
13     // yaratilan nesne "yok edilmelidir".
14     ok *BenimGuzelOkum = new ok;
15
16     // Okun hangi iki nokta arasinda olacagi
17     // Nokta (".") yerine isaretci ("->") islemcisi
18     // kullanildigina dikkat edin !!!
19     BenimGuzelOkum->degerAta(0.0, 5.0, 10.0, 5.0);
20
21     // Oku olusturan noktalari goster
22     BenimGuzelOkum->bas();
23
24     // Oku olusturan cizgilerin toplam uzunlugunu goster
25     printf("Uzunluk = %f \n", BenimGuzelOkum->uzunluk());
26
27     // Nesne, kullanimi sonunda yok ediliyor; kullandigi
28     // fiziksel hafiza serbest birakiliyor ya da isletim
29     // sisteminin kullanimina geri donuyor.
30     delete BenimGuzelOkum;
31
32     // Programdan basari ile cik
33     return 0;
34 }
35
```

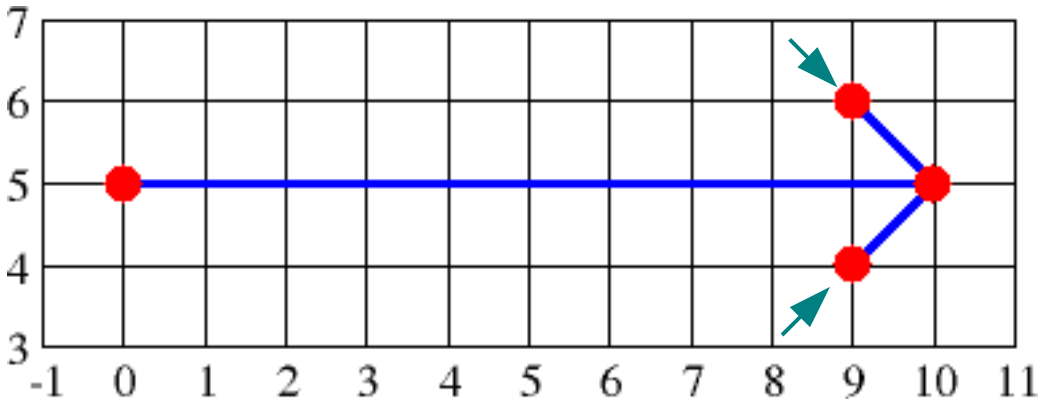
Ln 35, Col 1 INS

Örnek

Ok - Çıktı

- Kullanıcı programını ve iki class' tan oluşan kütüphaneyi:
  `g++ anaProgram.cxx cizgi.cxx ok.cxx -o anaProgram`
  ile derleyip çalıştırdığımızda...

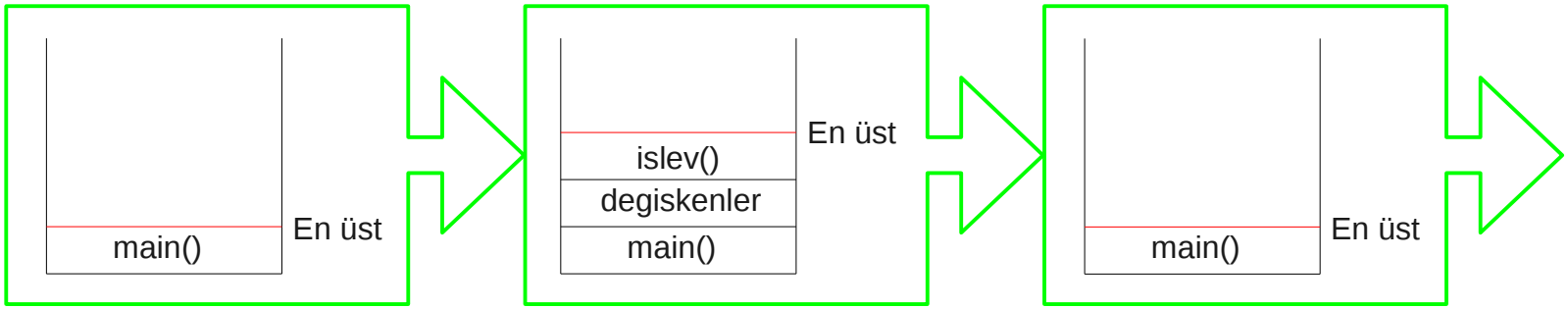
```
oc@olmak2:~/Documents/HEP_Okulu/workDir/ok$ ./anaProgram
Ana cizgi:
1. Nokta = (0.000000, 5.000000)
2. Nokta = (10.000000, 5.000000)
Ust cizgi:
1. Nokta = (9.000000, 4.000000)
2. Nokta = (10.000000, 5.000000)
Alt cizgi:
1. Nokta = (9.000000, 6.000000)
2. Nokta = (10.000000, 5.000000)
Uzunluk = 12.828426
oc@olmak2:~/Documents/HEP_Okulu/workDir/ok$ _
```



Kaynak (Code), Öbek (Heap) ve Yığın (Stack)

Yazılımların kullandığı hafıza bölgeleri

- Bir program hafızaya yüklendiğinde, üç hafıza bölgesi (segment) kullanılır: text ya da code (***kaynak***), heap (***öbek***) ve stack (***yığın***):
  - Kaynak bölgesi derlenmiş ve çalıştırılabilir olan programın makine ifadesinin işgal ettiği yerdir. Program çalıştığında atılacak adımların bir sıralaması niteliğindedir. Tüm kullanıcı ve sistem işlevlerinin bulunduğu alandır.
  - ***Öbek*** ve ***yığın*** ise derleyici tarafından veri saklamak için kullanılabılecek sistem hafızasıdır ve anlam bakımından farklı görünmeseler de aralarında işlev bakımından büyük farklar vardır:
    - ***Yığın (stack)***, bir sıralı (procedural) programda tanımlanan sıradan değişkenlerin işgal ettiği ve programdan ya da o değişkenin tanımlandığı kapsamdan çıkıldığında salıverilen, son-giren-ilk-çıkar (LIFO) yapısında bir hafızadır. Denetimi derleyiciye bırakılmıştır.
    - ***Öbek (heap)*** ise veri saklamak için daha uygun olan, kullanıcının denetiminde ayrılan ve salıverilen bir hafıza çeşididir. Nesne yönelimli programlamada sıkça kullanılır. Öbek'te ayrılan hafıza salıverilmediğinde, hafıza kaçağı denen olumsuzluğa yol açar.
- ***Bir nesneyi heap'te ya da stack'ta yaratmak*** ile bunların getirileri ve götürüleri, ileriki uygulamalarda daha açık hale gelecektir



Yigin çalisma yapisi

Örneklerle: C/C++ Giriş

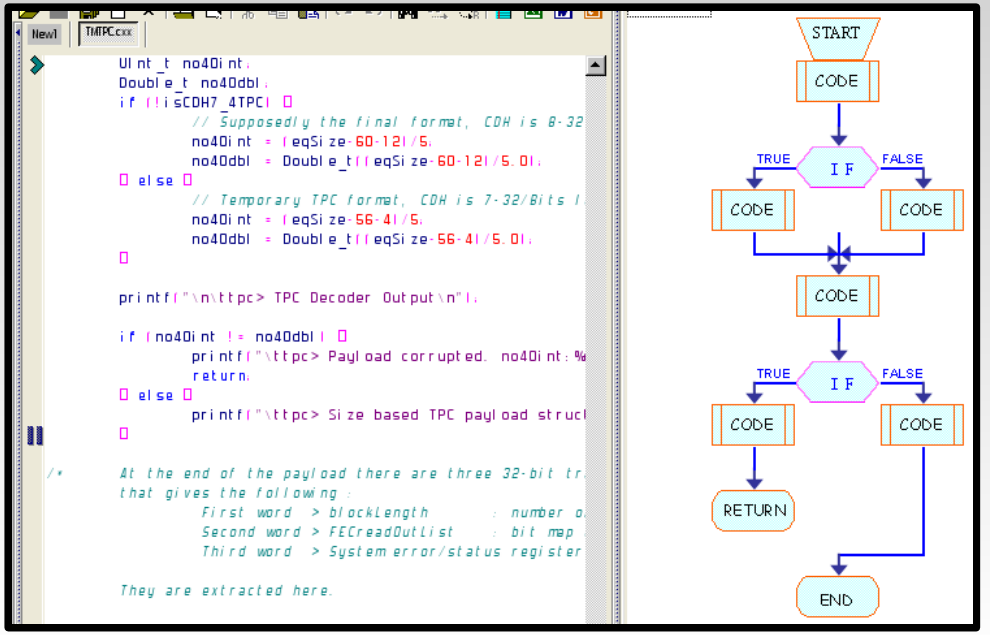
Cebirdeki Düşünce Biçimi

- Amaç
  - Geliştirici ne ister ?
  - Geliştirici ne yapar ?
- Sıralı Programlama
  - Telefon defteri uygulaması
- Nesneye Yönelmek
  - Karmaşadaki Sadelik
  - ÇokGen'den türeyen ÜçGen ve DörtGen
- Çizgiden Türeyen Ok
  - Cizgi class'ı ve uygulaması
  - Ok class'ı ve uygulaması
  - Kullanıcı programındaki kullanımı
  - Kullanıcı programında işaretçi (pointer) ile kullanımı
- Kaynak (code), Öbek (heap) ve Yığın (stack)
  - Yazılımların kullandığı hafıza bölgeleri
- Düşünmek Derken
  - Akış ve UML çizelgeleri
- Temel Programlama İfadeleri
  - Döngüler ve şart koşma
  - Rastlantısal sayılar ile π sayısının hesaplanması
  - π sayısı hesabı için FORTRAN ve C uygulaması ile akış çizelgesi
- Kütüphane Derlemek ve Kullanmak
  - Cizgi ve ok class' larını libCizgiVeOk.so kütüphanesine derlemek ve kullanmak
- ROOT' tan Bahis...
  - Kurulum ve “Merhaba dünya !!” alıştırması
- Proje Ödevinden Bahis...
  - Çift RPC (resistive plate chamber) deneyi ile kozmik ışınların yeniden oluşturulması
- Akşam Sefası
  - Ertesi günün sabahına hazırlanması beklenen akşamlık ödevler

Düşünmek Derken...

Akış ve UML çizelgeleri

- Sıralı programlamada akış çizelgesi
- Nesne yönelimli programlamada UML (unified modelling language) çizelgesi
  - UML içindeki class için akış çizelgesi

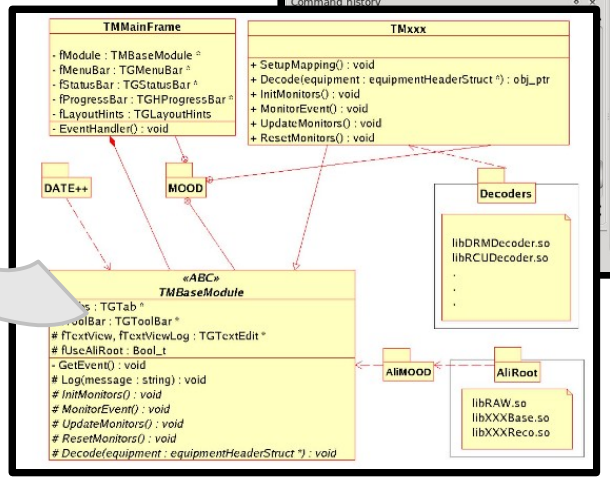


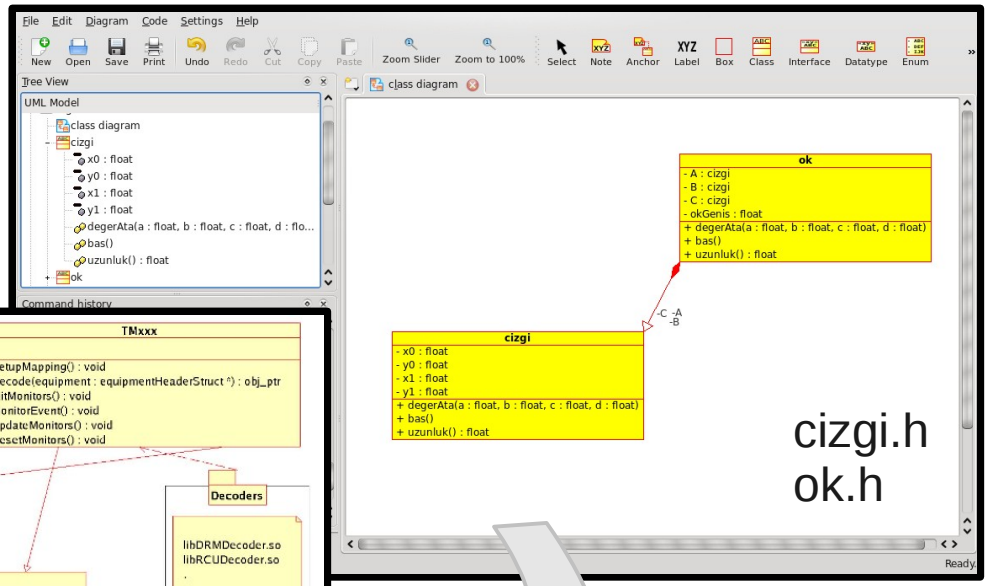
- Bir kere UML çizelgesi oluşturuldu mu bundan class üretimi çok kolaydır (***otomatik***)
- Bir kere class yazıldı mı mimarinin neye benzediği kuş bakışı izlenebilir; class' tan UML çizelgesi üretilebilir (***otomatik***)

```
class Dortgen: public Co
  public:
    int Alan ()
      { return (genislik
  };

class Ucgen: public CokG
  public:
```

UML' den üretilmiş kaynak.





Daha Karışık

Temel İfadeler

Döngüler ve Şart Koşma

- bool trackTable(x_1,y_1,x_2,y_2) olarak tanımlanmış 4 boyutlu bir dizi, iki RPC algılayıcısı arasında, A(x_1,y_1) ve B(x_2,y_2) noktaları arasında parçacık izi (track) olup olmadığını, "evet" veya "hayır" biçiminde aklında tutan bir değişkendir. Yandaki döngü bu değişkeni sıfırlamak için yazılmıştır.

File Edit View Search Tools Documents Help

New Open Save Print... Undo Redo Cut Copy Paste

kisalt.c | telefonDefteri.C | TTRPC.cxx

```
613     logThis("Histograms are reset.");
614     for (int i=0 ; i<16 ; i++) {
615         for (int ii=0 ; ii<16 ; ii++) {
616             for (int iii=0 ; iii<32 ; iii++) {
617                 for (int iiii=0 ; iiii<32 ; iiii++) {
618                     trackTable[i][ii][iii][iiii]=kFALSE;
619                 }
620             }
621         }
622     }
623     logThis("3D track table is reset.");
```

Ln 613, Col 1 INS

File Edit View Search Tools Documents Help

New Open Save Print... Undo Redo Cut Copy Paste Find Replace

binary_vs_thermometer.C | kisalt.c | telefonDefteri.C | TTRPC.cxx

```
588 //_____
589 void TTRPC::Initialize()
590 {
591 // Initializes :)
592
593     logThis("Initializing...");
594     logThis("-------------------------");
595     if (0!=check4data()) {
596         logThis("Method : Initialize - Can not *open* data file !...");
597     } else {
598         if (0!=readDataFile()) {
599             logThis("Method : Initialize - Can not *read* data file !...");
600         } else {
601             logThis("Data file is read succesfuly.");
602         }
603     }
```

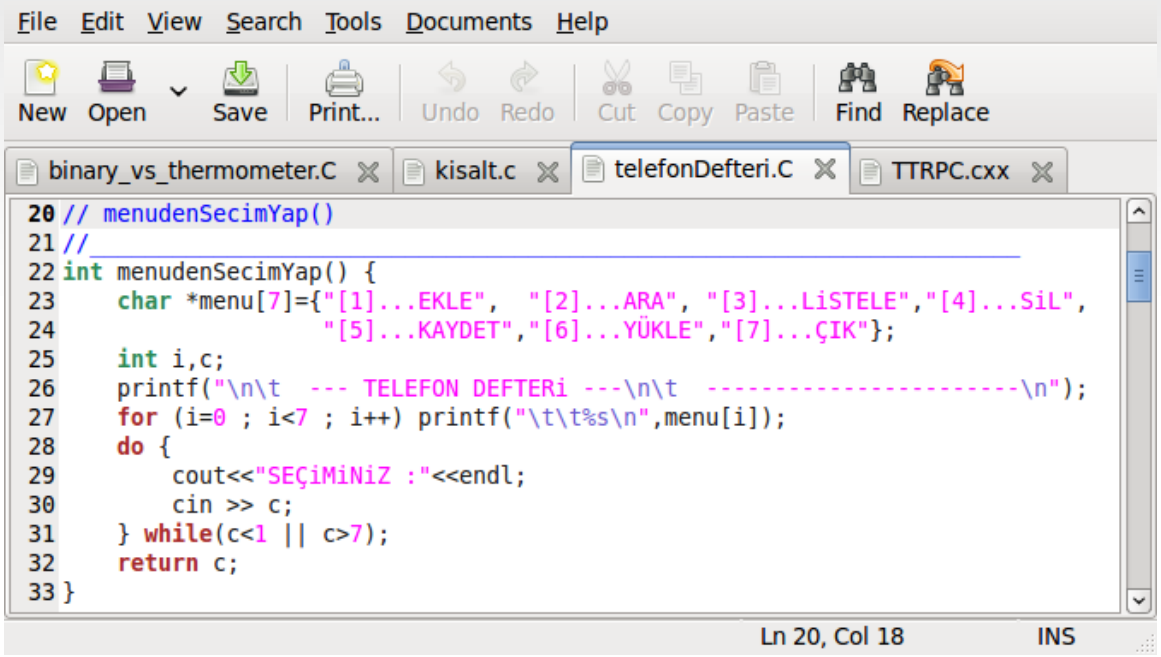
Ln 588, Col 1 INS

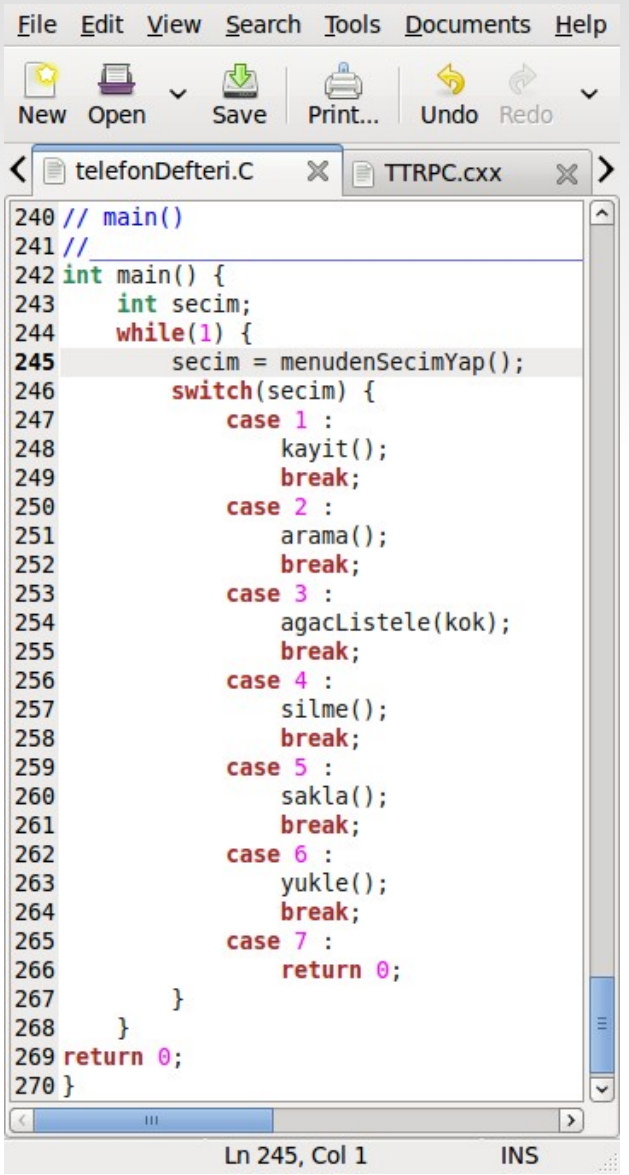
- check4data() işlevi başarı ile tamamlan***ma***mışsa (yani sıfır döndür***me***miş ise) log kütüğüne ilgili hatayı yaz; başarılı ise bu sefer veriyi okumayı dene (readDataFile() işlevi) ve başarılı olup olmadığına göre log kütüğüne ilgili çıktıları yaz.

Temel İfadeler

Döngüler ve Şart Koşma

- Kullanıcıya sonsuza kadar seçim yaptır; while içindeki şartın her zaman sağlandığına dikkat edin (244. satır)
- Tam sayı olarak tanımlanmış "secim" değişkenine şart koş:
  - seçim, 1' e eşitse *kayit()* işlevini çağır (248. satır)
  - seçim, 7 ise programdan çık (266. satır)



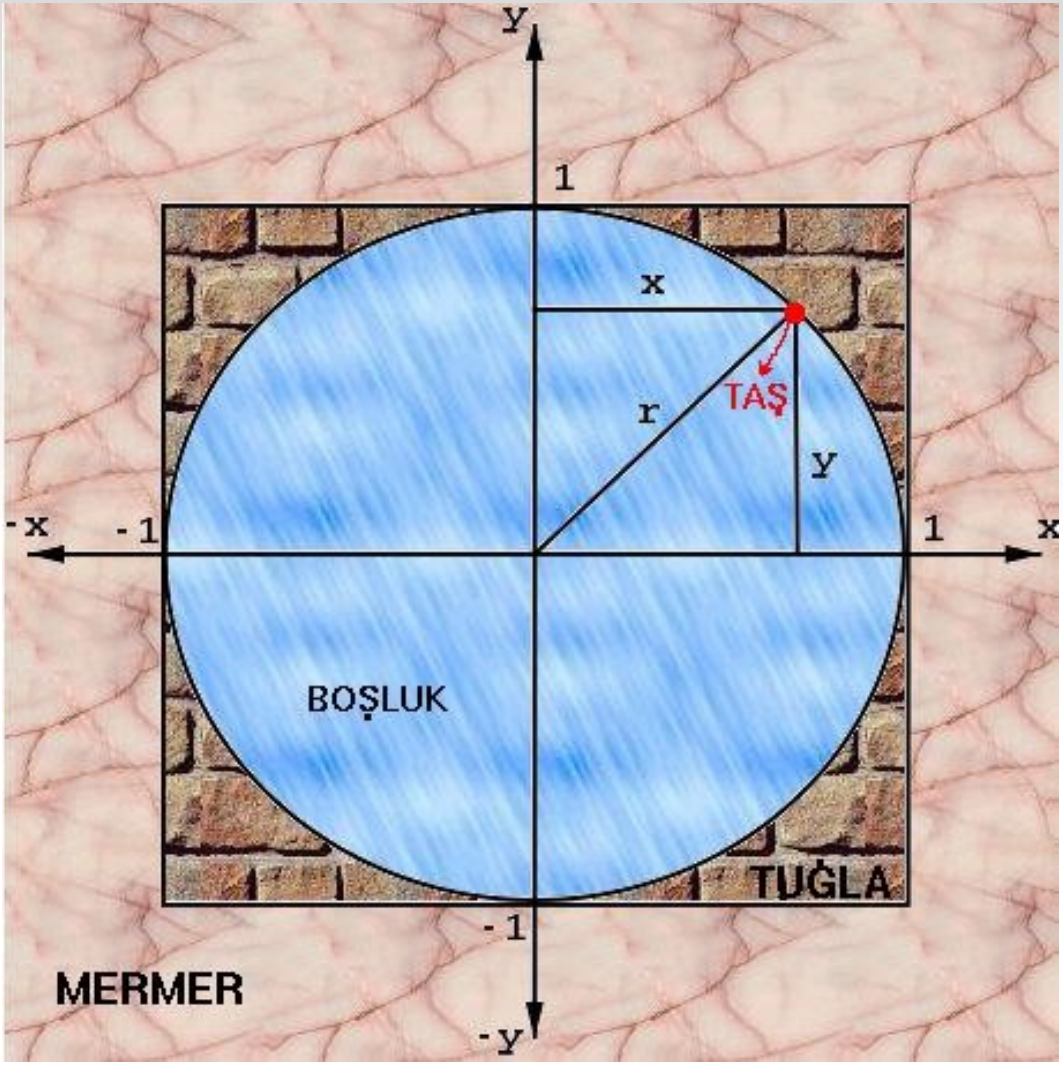


- Telefon defteri uygulamasında kullanıcıya bir komut dizgesi gösteren ve kullanıcının seçimini kendisini çağırana göndüren (return c) işlev.
  - Döngü ile dizge elemanları basılıyor (27. satır)
  - Kullanıcı 1 ile 7 arasında bir seçim girene kadar soru tekrarlanıyor (31. satır)
  - Girdi alındığında seçim çağırana döndürülüyor (32. satır)

Temel İfadeler

Rastlantısal Sayılar ile yaklaşık π sayısını bulmak

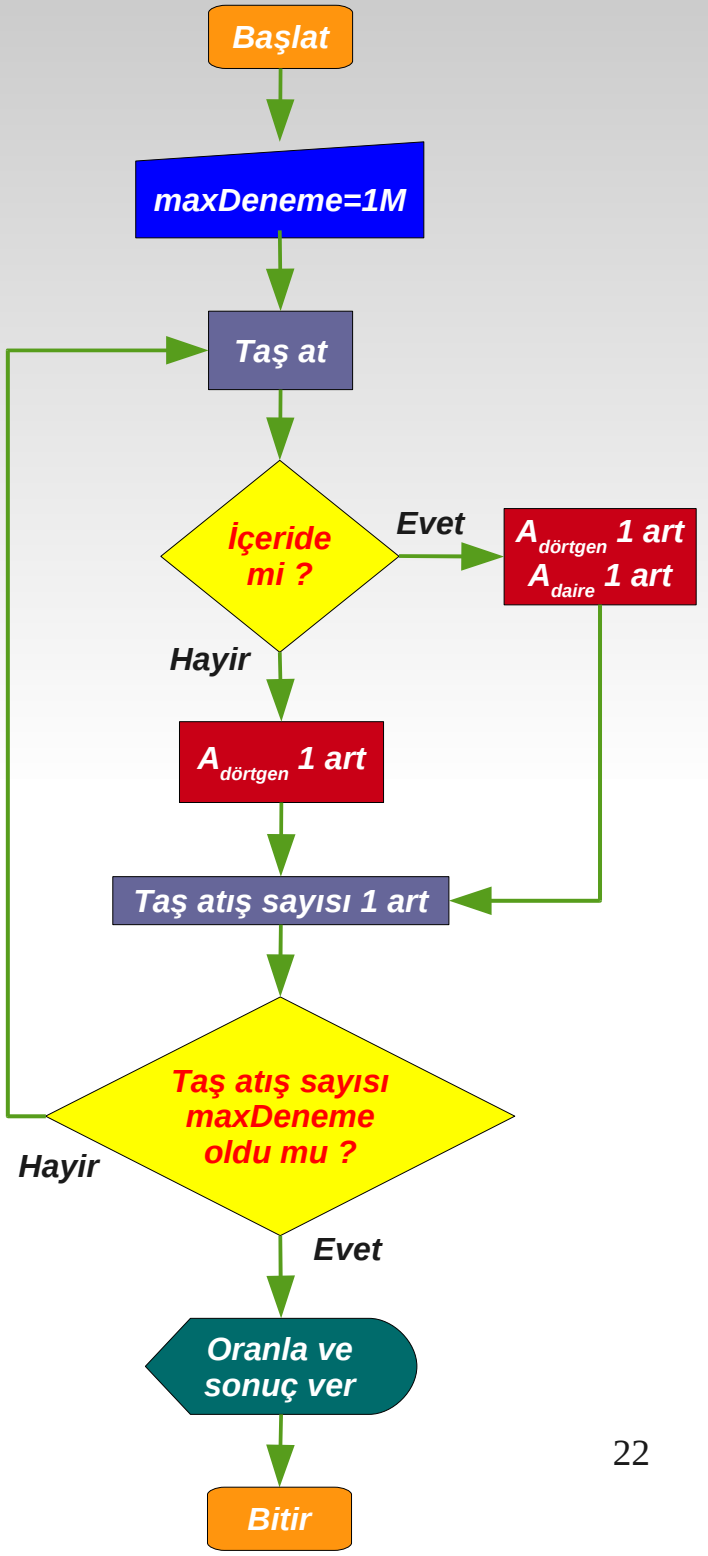
- Üzerinde r yarıçaplı bir delik bulunan bir duvara, kenarları bu delığe değecek şekılde kenar uzunluğu 2r olan bir kare çizilir:
  - $A_{dörtgen} = \pi r^2$
  - $A_{daire} = 4r^2$
  - $A_{dörtgen}/A_{daire} = \pi/4$
- Bu oranı hesaplamak için birim uzunluktaki dörtgen içine rastgele taş atılıp, tasın dairenin içine denk gelip gelmediğine şart koşulur:
  - $r^2 = 1 < x^2+y^2$
- Şartın sağlandığı atış sayısının, toplam atılan taş sayısına oranı aranılan sonucun dörtte birine eşit olacaktır.



Temel İfadeler

Rastlantısal sayılar ile yaklaşık π sayısını bulmak

```
 1 #include <math.h>
 2 #include <stdio.h>
 3 #include <stdlib.h>
 4 #include <time.h>
 5
 6 int main() {
 7     float x, y, r, pi;
 8     int i, maxDeneme=1000000;
 9     int daireAlan, kareAlan;
10
11     // RND sayi uretecini baslatiyoruz
12
13     srand(time(NULL));
14     kareAlan  = 0;
15     daireAlan = 0;
16     pi        = 0;
17
18     // Dortgen icine tas atmaya basliyoruz
19
20     for (i=0 ; i<maxDeneme ; i++) {
21         x=rand()/(1.0*RAND_MAX);
22         y=rand()/(1.0*RAND_MAX);
23         r=sqrt(x*x+y*y);
24         if (r<=1.0) {
25             daireAlan += 1;
26             kareAlan  += 1;
27         } else {
28             kareAlan += 1;
29         }
30     }
31
32     // Programi bitiriyoruz
33
34     printf("Dairenin icine girenler = %d \n", daireAlan);
35     printf("Karenin  icine girenler = %d \n", kareAlan );
36     printf("Hesaplanan pi  = %f \n", 4.0*daireAlan/kareAlan);
37
38     return 0;
39 }
```



Temel İfadeler

Rastlantısal Sayılar ile yaklaşık π sayısını bulmak

File Edit View Search Tools Documents Help

New Open Save Print... Undo Redo Cut Copy Paste

piBul.cxx | piBul.f

```
 1 #include <math.h>
 2 #include <stdio.h>
 3 #include <stdlib.h>
 4 #include <time.h>
 5
 6 int main() {
 7     float x, y, r, pi;
 8     int i, maxDeneme=1000000;
 9     int daireAlan, kareAlan;
10
11     // RND sayi uretecini baslatiyoruz
12
13     srand(time(NULL));
14     kareAlan  = 0;
15     daireAlan = 0;
16     pi        = 0;
17
18     // Dortgen icine tas atmaya basliyoruz
19
20     for (i=0 ; i<maxDeneme ; i++) {
21         x=rand()/(1.0*RAND_MAX);
22         y=rand()/(1.0*RAND_MAX);
23         r=sqrt(x*x+y*y);
24         if (r<=1.0) {
25             daireAlan += 1;
26             kareAlan  += 1;
27         } else {
28             kareAlan += 1;
29         }
30     }
31
32     // Programi bitiriyoruz
33
34     printf("Dairenin icine girenler = %d \n", daireAlan);
35     printf("Karenin  icine girenler = %d \n", kareAlan );
36     printf("Hesaplanan pi  = %f \n", 4.0*daireAlan/kareAlan);
37
38     return 0;
39 }
```

Ln 18, Col 43 INS

File Edit View Search Tools Documents Help

New Open Save Print... Undo Redo Cut Copy Paste

piBul.cxx | piBul.f

```
 1       PROGRAM piBul
 2       IMPLICIT NONE
 3
 4       REAL x, y, r, pi
 5       INTEGER i, maxDeneme
 6       INTEGER daireAlan, kareAlan
 7       PARAMETER (maxDeneme = 1000000)
 8
 9 c Rnd sayi uretecini baslatiyoruz
10
11       r = RAND(TIME())
12       kareAlan  = 0
13       daireAlan = 0
14       pi = 0
15
16 c Dortgen icine tas atmaya basliyoruz
17
18       DO i = 1, maxDeneme
19
20           x = RAND()
21           y = RAND()
22           r = SQRT(x*x+y*y)
23
24           IF (r.LE.1.0) THEN
25               daireAlan = daireAlan + 1
26               kareAlan = kareAlan + 1
27           ELSE IF (r.GT.1.0) THEN
28               kareAlan = kareAlan + 1
29           END IF
30
31       END DO
32
33 c Programi bitiriyoruz
34
35       PRINT*,'Dairenin icine girenler = ', daireAlan
36       PRINT*,'Karenin  icine girenler = ', kareAlan
37       pi = 4.0*daireAlan/kareAlan
38       PRINT*,'Hesaplanan pi = ', pi
39       END
```

Ln 16, Col 38 INS

Kütüphane Derlemek ve Kullanmak

Çizgi ve Ok class' larını libCizgiVeOk.so kütüphanesine derlemek ve kullanmak

- Her class tek başına "**-c**" g++ anahtarı ile derlenir ve o class'a ait object (nesne) kütüğü oluşturulur (*.o)
- Kütüphaneye (*.so kütüğü) konmak istenen nesneler (*.o kütükleri) "**-shared**" g++ anahtarı ile derlenir. Artık **libCizgiVeOk.so** kütüphanesi oluşmuştur.
- Herkesin kullanımına açmak için ***/usr/lib*** gibi ortak erişime açık bir yere yüklenir.
- Kullanıcı programlarında, "**-lCizgiVeOk**" anahtarı ile kullanılır.

```
> g++ -c -fPIC cizgi.cxx
> g++ -c -fPIC ok.cxx
> g++ -shared ok.o cizgi.o -o libCizgiVeOk.so
> sudo cp libCizgiVeOk.so /usr/lib/
> g++ anaProgram.cxx -o anaProgram -lCizgiVeOk
> ./anaProgram
Ana cizgi:
1. Nokta = (0.000000, 5.000000)
2. Nokta = (10.000000, 5.000000)
Ust cizgi:
1. Nokta = (9.000000, 4.000000)
2. Nokta = (10.000000, 5.000000)
Alt cizgi:
1. Nokta = (9.000000, 6.000000)
2. Nokta = (10.000000, 5.000000)
Uzunluk = 12.828426
> _
```

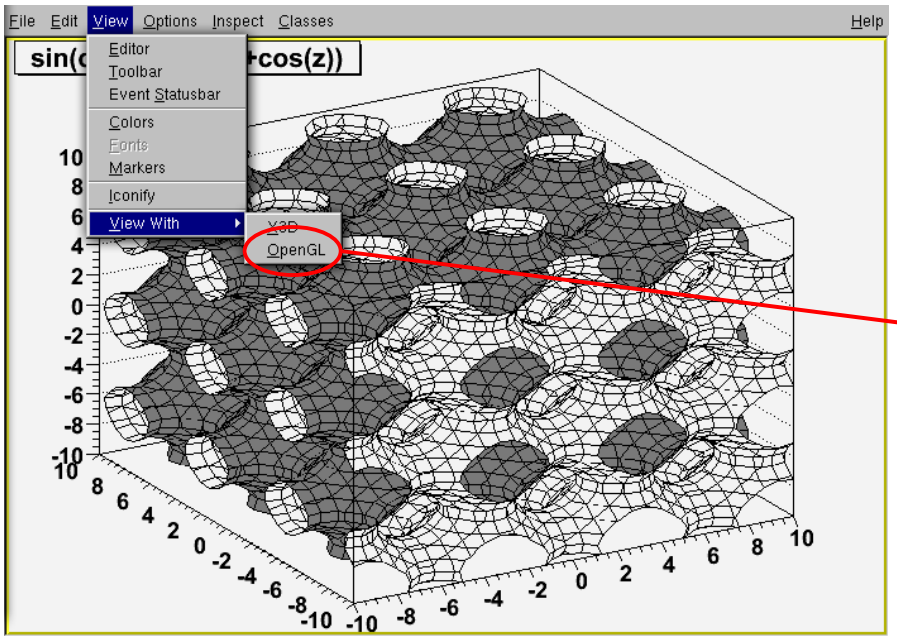
- Kullanıcının kütüphaneyi kullanmak için class kaynaklarına (source code) ihtiyaç duymayacağına ***fakat*** kütüphanede hangi işlevlerin (method) varolduğunu bilmeye ihtiyaç duyacağına dikkat edin.

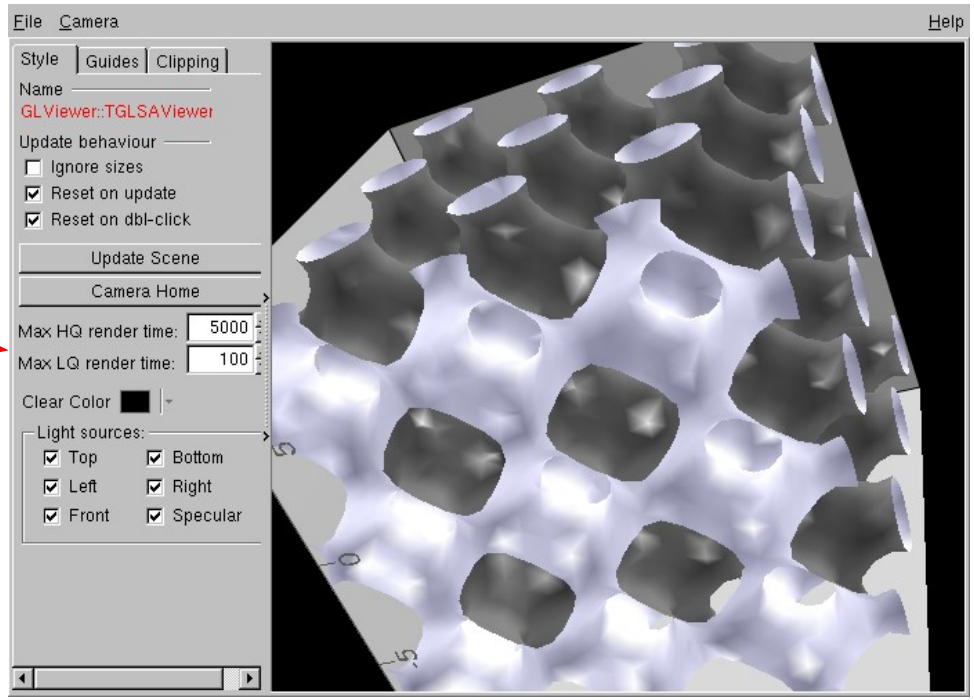
ROOT' tan Bahis

Kısa kurulum ve "merhaba dünya" alıştırması

```
> tar xvfz root_v5.22.00.source.tar.gz
> export ROOTSYS=$HOME/root
> export PATH=$PATH:$ROOTSYS/bin
> export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ROOTSYS/lib
> cd root
> ./configure
> make
> sudo make install
> root -l
root [0] TF3 f1("MerabaDunya","sin(cos(x)+sin(y)+cos(z))",-10,10,-10,10,-10,10)
root [1] f1->Draw()
<TCanvas::MakeDefCanvas>: created default TCanvas with name c1
root [2] _
```

**

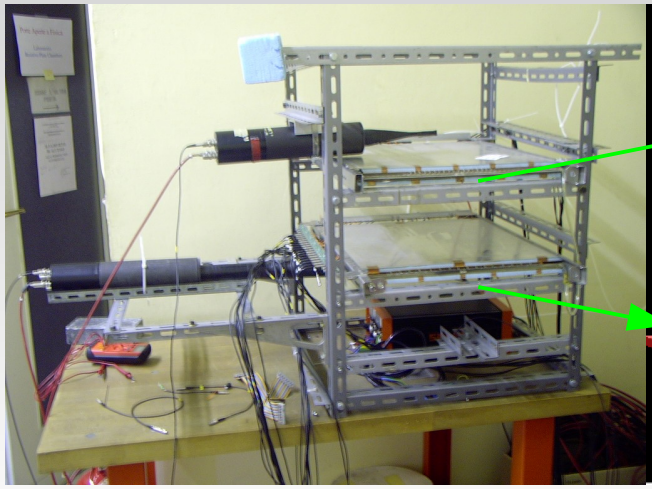


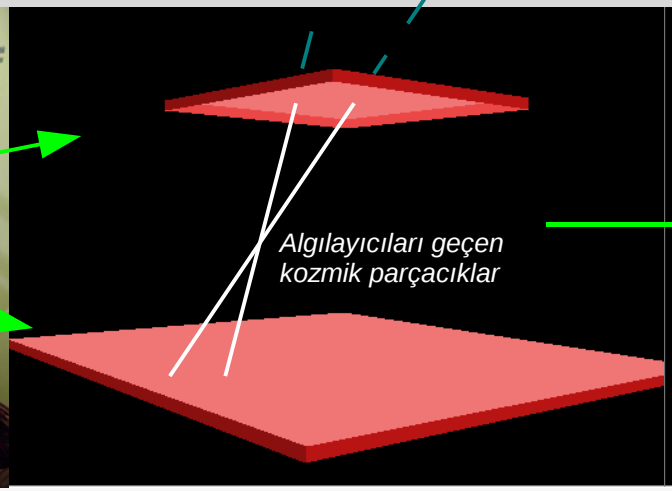


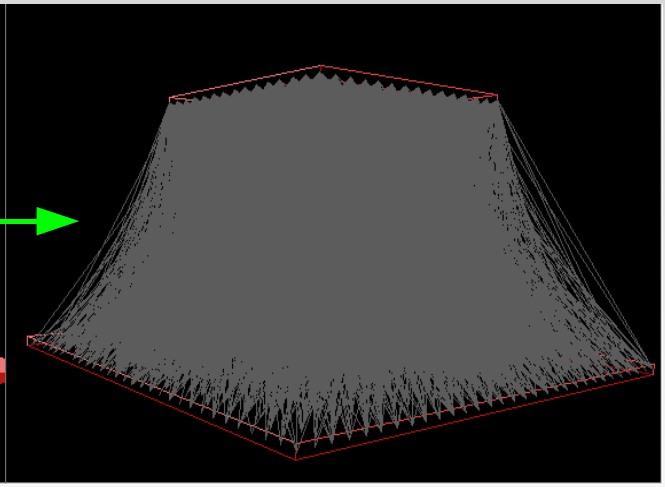
** Bu satırları $HOME/.bashrc kütüğüne yazın ve her defasında tanımlamaktan kurtulun (ilk defa bu işlemi yaptığınızda "source $HOME/.bashrc" komutuyla kütüğün yeni içeriğinin geçerlilik kazanmasını sağlayın)

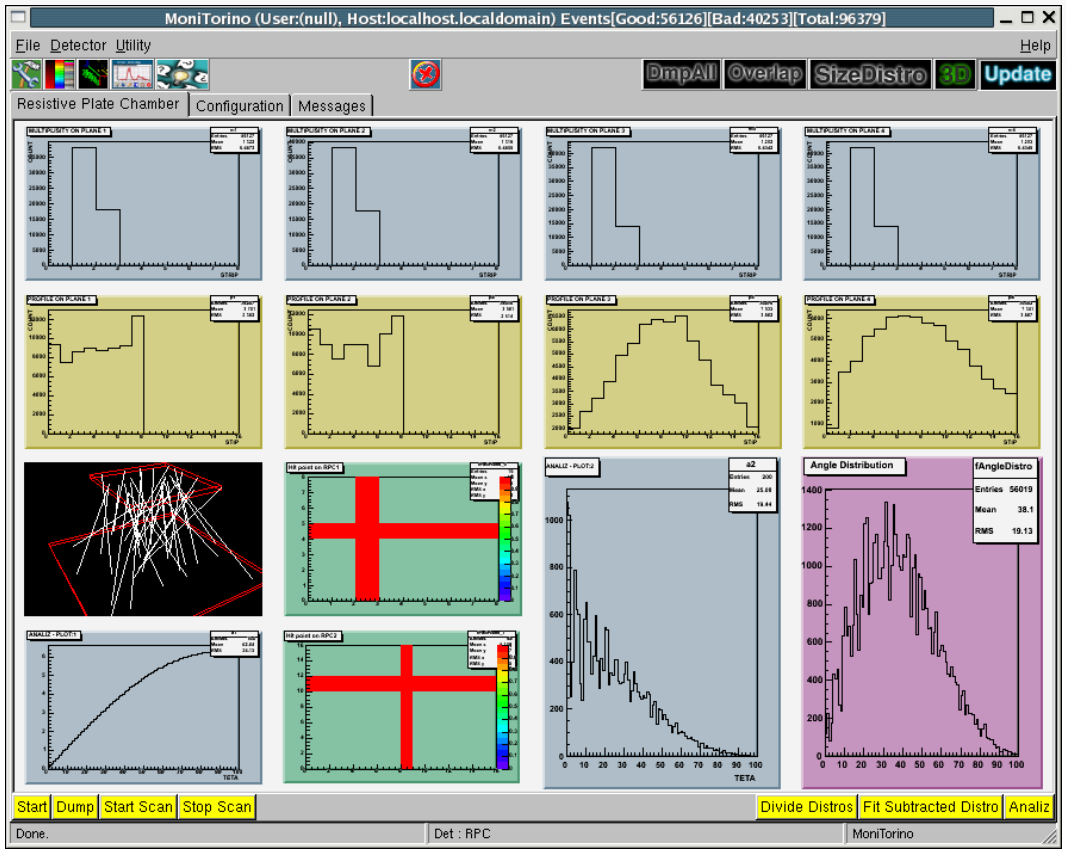
Bitirme Ödevinden Bahis

Çift RPC Deneyi

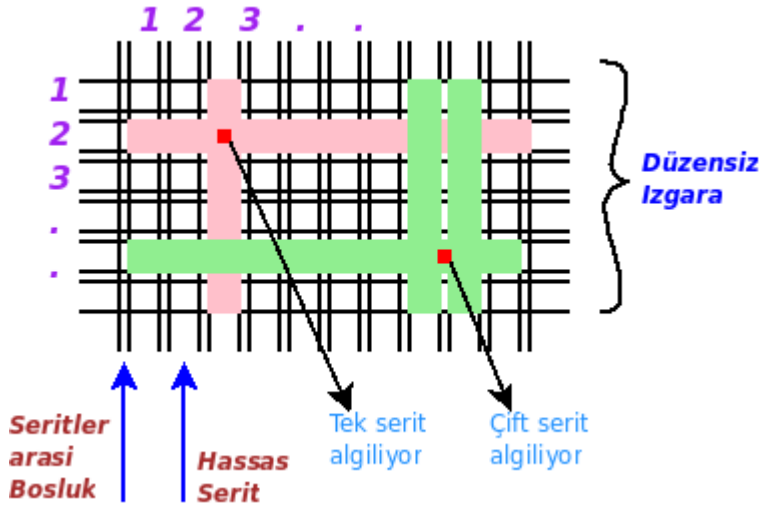








- Alt (32x32 şerit) ve üst (16x16 şerit) RPC
- Her RPC birbirine dik iki ızgaraya sahip
- İzgaraların her bir şeridi (strip) 1 bitlik "evet" ya da "hayır" üretiyor
- Olay = 32-bit' lik üç sayı



Örneklerle: C/C++ Giriş

Cebirdeki Düşünce Biçimi

- Amaç
  - Geliştirici ne ister ?
  - Geliştirici ne yapar ?
- Sıralı Programlama
  - Telefon defteri uygulaması
- Nesneye Yönelmek
  - Karmaşadaki Sadelik
  - ÇokGen'den türeyen ÜçGen ve DörtGen
- Çizgiden Türeyen Ok
  - Cizgi class'ı ve uygulaması
  - Ok class'ı ve uygulaması
  - Kullanıcı programındaki kullanımı
  - Kullanıcı programında işaretçi (pointer) ile kullanımı
- Kaynak (code), Öbek (heap) ve Yığın (stack)
  - Yazılımların kullandığı hafıza bölgeleri
- Düşünmek Derken
  - Akış ve UML çizelgeleri
- Temel Programlama İfadeleri
  - Döngüler ve şart koşma
  - Rastlantısal sayılar ile π sayısının hesaplanması
  - π sayısı hesabı için FORTRAN ve C uygulaması ile akış çizelgesi
- Kütüphane Derlemek ve Kullanmak
  - Cizgi ve ok class' larını libCizgiVeOk.so kütüphanesine derlemek ve kullanmak
- ROOT' tan Bahis...
  - Kurulum ve “Merhaba dünya !!” alıştırması
- Proje Ödevinden Bahis...
  - Çift RPC (resistive plate chamber) deneyi ile kozmik ışınların yeniden oluşturulması
- Akşam Sefası
  - Ertesi günün sabahına hazırlanması beklenen akşamlık ödevler

Örneklerle: C/C++ Giris

Cebirdeki Düsünce Biçimi

Bu kadar C/C++ yeter mi ? Ödevlere geçelim mi ?

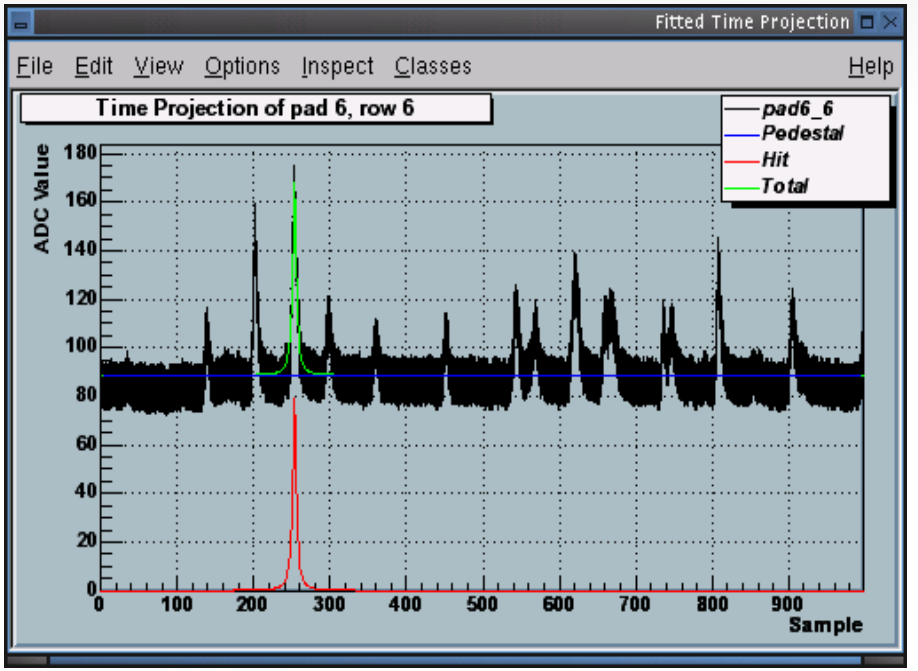
Aksam Sefası

Ertesi günün sabahına hazırlanması beklenen akşamlık ödevler

- Yukarıdaki örneklerde “ok” isimli class' ta okun genişliğini belirleyen değişken “okGenis”, oku oluşturan ana çizginin %10' u olarak belirlenmiştir. Bu class üzerinde gerekli değişiklikleri yaparak, “okGenis” değişkenini kullanıcının denetimine bırakın. Kullanıcı yaratılan yeni nesneler için, bu değişken üzerinde işlem yapmak istemediğinde değeri ön tanımlı olarak %10 kalsın.

- Yukarıdaki örneklerde “ok” isimli class' ta çizilen ok, sağa doğru değil de sola, yukarıya doğru veya açılı çizilseydi sonuç ne olurdu ? Neden ? Bu sonuç kullanıcının istediği bir sey midir ? Olmadığını düşünüyorsanız ok class' ı üzerinde ne gibi bir düzeltmenin gerekli olduğunu belirtiniz. (Seçime bağli kısım: gerekli değişikliği yapınız.)

- **int pad6_6[1000]** biçiminde tanımlanmış bir dizi, “*Sample*” başına “*ADC Value*” bilgisini tutmaktadır. Dizi yan tarafta çizdirilmiştir (siyah). En yüksek “*ADC Value*”sına sahip olan “Sample”i bulan ve bunu kendisini çağırana döndüren “***int bul()***” adlı c/c++ işlevini (fonksiyon) yazınız. Akış çizelgesini çiziniz.



- *Çarpım tablosunu hesaplayan ve basan cift döngüyü yazınız.*
- *Kendisine verilen sayının tek mi çift mi olduğunu anlayan programı yazınız.*
- *Çarpım tablosunda 5' ler ve 8' ler hariç geri kalanı hesaplayan döngüyü yazınız.*
- *100000' e kadar, abcdabcd biçimindeki sayıları tam bölen sayıları bul (47284728 gibi).*
- *Çoban salatası yapma akış çizelgesini hazırlayınız.*